NASA / TM-1999-209707

Investigating Interruptions: Implications for Flightdeck Performance

Kara A. Latorella
Langley Research Center, Hampton, Virginia

October 1999

The NASA STI Program Office ... in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results ... even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at *http://www.sti.nasa.gov*

- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA STI Help Desk at (301) 621-0134

- Phone the NASA STI Help Desk at (301) 621-0390

- Write to:
  NASA STI Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076-1320

NASA / TM-1999-209707

Investigating Interruptions: Implications for Flightdeck Performance

Kara A. Latorella
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

October 1999

Available from:

NASA Center for AeroSpace Information (CASI)
7121 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 605-6000

Note

This document is a publication of the dissertation "Investigating Interruptions: Implications for Flightdeck Performance," submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree at the State University of New York at Buffalo's Industrial Engineering program. This research was conducted at NASA Langley in Hampton, Virginia in the Crew/Vehicle Integration branch and sponsored by a NASA Graduate Student Researcher's fellowship (grant number NGT-50992). The research reported in this publication was concluded and first published as the dissertation on November 1, 1996.

Acknowledgments

The opportunity to acknowledge the contributions others make to one's work formally, or perhaps, extensively is rare. I plan to take full advantage of this one. Many people contributed to and facilitated this work. It would indeed have been impossible without the financial, administrative, intellectual, technical, and, most importantly, the spiritual support I received.

I am grateful to the Industrial Engineering Department, and its Chairman, Dr. Rajan Batta, for providing tuition waivers throughout most of this work; to Dr. Valerie Shalin for her help in preparing the GSRP application; to Mr. Doug Arbuckle for accepting my initial GSRP proposal; to Dr. Kathy Abbott for accepting my proposal to work in the Crew/Vehicle Integration branch and for funding this work; to Mr. Paul Schutte for providing funding from his group within Crew/Vehicle Integration branch; and to Dr. Alan Pope for graciously providing supplemental funding on short notice. Though not directly providing financial support, there have been many individuals that have come to the rescue when the wheels of the several bureaucracies involved in this project have threatened to have me in their knurls. In addition to the heroic efforts of Dr. Colin Drury and Mr. Paul Schutte, these people include: the secretaries in the Industrial Engineering Department at the University of Buffalo, particularly Ms. Kathy Dunphy and Mrs. Kris Bovenzi; Mr. John Boehler at the University of Buffalo's Sponsored Programs office; and Mrs. Susan Conry, the Crew/Vehicle Integration branch secretary.

Intellectual contributions to this work also arose from numerous sources. My sincere thanks go to the members of my dissertation committee; Dr. Colin Drury, Dr. Valerie Shalin, and Dr. Joseph Sharit, for their unique contributions to this work, and to my graduate education as a whole. Because this work was conducted primarily at NASA Langley in Virginia, they were required to wade through numerous and lengthy e-mails, and participate in occasional protracted coordination meetings; non-optimal conditions to which they graciously adapted. Dr. Kenneth Funk, III, kindly accepted the task of providing an external review of this work. My thanks to him for his swift and effective accomplishment of this task, encouraging comments and constructive criticisms. Several people at NASA Langley also provided helpful perspectives on this work through casual conversation. In particular, this work benefited from discussions early in the process with Dr. William Rogers; and from sporadic but urgent "reality-checks" with Mr. Paul Schutte throughout its course. Dr. Derrick Sugg, Dr. Marianne

Rudisill, and Dr. Ray Comstock kindly volunteered their time to discuss the statistical treatment of experimental data.

Preparing the simulator and scenario for this experiment was a major undertaking. Arriving at NASA Langley as a complete novice in aviation practice and terminology as well as scenario design, I relied heavily on the technical expertise and enduring goodwill of many people at NASA Langley to help me to develop the experimental conditions and platform. Captain Dave Simmon (ret.) served as my primary information source, guinea pig, sounding board, and instructor in all-things-avionic during this project. He not only provided information, both general and specific to scenario development, but taught me a great deal of invaluable incidentals from his experiences in pilot training, incident and accident analyses, and "pilot-speak". He is a tireless and invaluable resource, but above that, is an extremely patient and generous soul. Captian Richard "Skeet" Gifford's (ret.) additional assistance and good humor are also greatly appreciated. Mr. Frank McGee provided real-time communications during the experiment, endured patiently what seemed like endless sessions recording the static interruption messages, and provided the occasional encouraging "bravo-zulu!" at the most appropriate times. Simulation development relied heavily on the programming mavens, Mrs. Arlene Guenther and Mrs. Wei Anderson, who tolerated my inexperience exceptionally well; Mr. Jerry Karwac's assistance with graphic displays; the TSRV simulator hardware and communications personnel, Mr. Jerry Galla, Mr. Alan Douglas, Ms. Olivia Hunter, Mr. Walter Bowser, Mr. Dale Ashcomb, Mr. Bill Hooper, Mr. Dennis Frasca, and Mr. Wayne Burge, who implemented difficult requirements and even provided a constant and critical supply of caffeine throughout long simulation sessions; audio/visual equipment operators, Mr. Daryl Sacra and Mr. Phil Smith; Mr. Terrence Lawrence, who coordinated use of the VOTAN message recording system; and, for assistance in scheduling and security issues, Mr. Don Horne, Ms. Sonia Herndon, Mr. John Rollins, and Mr. Lem Meetze. Mrs. Regina Johns coordinated subject recruitment and accommodations, often skillfully rearranging and obtaining subjects on short notice. I am also grateful to the Human Engineering Methods group; Mrs. Debbie Rull, Mr. Ed Bogart, Mr. Dan Burdette, Dr. Ray Comstock and headed by Dr. Alan Pope, for their assistance with physiological measures for this experiment. The physiological data are not included in this dissertation but rather will be analyzed separately, however their efforts to continually improve subject preparation, data collection, and their flexibility throughout the process are significantly appreciated. Mr. John "super-user Jake" Barry coded intricate scripts for simulation data reduction, created a useful data visualization program, and patiently endured my attempts to learn UNIX and AWK. I owe a debt of gratitude to the NASA Langley employees who volunteered to pretest the experimental scenarios, Mr. Lee Persons, Mr. Charlie Knox, Mr. Dave Johnson, and Mr. Brad Perry; and to all the pilots who participated as preliminary subjects and actual subjects in this experiment. The technical expertise of Mrs. Lisa Stephens and others at the University of Buffalo's Engineering Computing Services group provided teleconferencing capabilities for my dissertation defense, allowing me to finish this degree without delay; for which I am extremely thankful.

This project would also have been impossible without the tremendous spiritual support and encouragement I have received from my committee, co-workers at NASA Langley, friends, and family. My committee, co-workers, and friends (non-exclusive categories) have all responded to my occasional fits of panic and bewilderment with greatly-appreciated humor and a confidence in me that has always been uplifting. My family has also always been extremely supportive, and tolerant of this endeavor: my brother, Marc, has tolerated many hypocritical big-sister lectures on how to do a thesis; and my extended family has been lighting candles in churches all over the greater Boston metropolitan area for me for years. I am, however, most grateful for my parents. It is difficult to express this most deeply-felt gratitude because their contributions have been both multidimensional and profound. My father has spent countless hours discussing this research, and my career in general, with me in sincere interest and loving determination to ease my way. My mother has always had a faith in me that I don't fully understand, but have always felt. They have always been unflagging in their devotion to me, providing constant encouragement, and the warmth of always-open arms. In addition, they are, perhaps, two of the best proof-readers on the planet. I would be remiss in not acknowledging the comfort provided by the enthusiastic w greetings and quiet presence of Daisy, my dog. She is, as I write this, curled around my feet patiently waiting for me to finish something - as usual.

Okay Dais - Let's go.

I'm done.

Table of Contents

List of Figures

List of Tables

Abbreviations / Glossary

Term	Definition
ACWS	Attitude Control Wheel Steering – ACWS is a reduced mode of the autopilot which provideds constant heading and attitude once flightpath deviations are manually eliminated.
AP	Approach Point – The AP is the waypoint just prior to the touchdown point.
ATC	Air Traffic Control(ler) – A service provided from a control tower for aircraft operating on the movement area and in the vicinity of an airport.
ATIS	Automatic Terminal Information System – ATIS is a continuous broadcast of recorded non-control, routine, but necessary information about a terminal area.
CAA	Auditory Change Altitude intervening task.
CADC	Central Air Data Computer.
CDU	Control Display Unit – The CDU is the interface to the FMS.
COMM	Communication channel.
CSA	Auditory Change Speed intervening task.
CRA	Auditory Change Runway intervening task.
Datalink	Datalink is a technology which provides digital information flow between ground services and flightdecks.
dBA	This unit measures sound pressure level calculated such that frequency ranges are weighted in a manner similar to the human ear’s attenuation.
DME	Distance Measuring Equipment – DMEs measure, in nm, the slant range distance of an arc from a navigational aid to a reference.
EDR	Electro-dermal Response.
EEG	Electro-encephologram.
EHA	Auditory Enter Hold intervening task.

Term	Definition
EKG	Electro-cardiogram.
EPR	Engine Pressure Ratio – EPR is a measure of engine function.
ETA	Estimated Time of Arrival.
FAF	Final Approach Fix procedure.
FMC	Flight Management Computer – The FMC allows pilots to preprogram a desired flightpath and obtain status information, among other control and information functions.
FMS	Flight Management System – The FMS includes the FMC and peripheral devices used to sense and program the aircraft.
FPA	Flight Path Angle – FPA is a parameter equal to the difference between pitch and the angle of attack (see appendix 5.14).
IP	Intervention Position.
IRA	Auditory Initial Runway intervening task.
IRS	Inertial Reference System.
IRV	Visual Initial Runway intervening task.
IT	Interrupting Task / Incidental Task.
KIAS	Knots of Indicated Airspeed.
MAF	Missed Approach Fix – The MAF is the point to which the aircraft should execute a missed approach procedure if the required visual conditions are not adequate to land.
NAV	Navigation channel.
ND	Navigational Display – The ND provides a plan-view of the programmed, actual, and projected flightpath.
nm	Nautical miles.

Term	Definition
NOTAM	Notice to Airmen – A NOTAM contains new information concerning the establishment, condition of, or change in any facility, service, procedure or hazard, the timely knowledge of which is essential to personnel concerned with flight operations.
OT	Ongoing Task.
PFD	Primary Flight Display – The PFD provides attitude, altitude, speed, and track current, target, and trend information.
TD	Touchdown point – The TD is a point located
TOD	Top of Descent point – The TOD point is the last waypoint for which the aircraft is at cruise altitude.
TRANS	Transponder.
TSRV	Transport Systems Research Vehicle – The TSRV is a fixed-base simulation facility at NASA Langley similar to a Boeing 737 flightdeck.
Vref30	Approach Reference Speed for Flaps 30 setting.
Waypoint	A pre-determined geographical position used for route/instrument approach definition, or progress-reporting, and that is defined relative to a navigational aid or in terms of latitude and longitude coordinates.
WYPT	Waypoint.
18K’	18,000 feet-altitude procedure.

Abstract

A fundamental aspect of multiple task management is to attend to new stimuli and integrate associated task requirements into an ongoing task set; that is, to engage in interruption management. Anecdotal evidence and field studies indicate the frequency and consequences of interruptions, however experimental investigations of the mechanisms influencing interruption management are scarce. The commercial flightdeck is a naturally multi-tasking work environment, one in which interruptions are frequent and of various forms. Further, interruptions have been cited as a contributing factor in many aviation incident reports and in at least one major accident. The flightdeck, therefore, provides an appropriate, real-world work environment for investigating interruptions, and one that could obviously benefit from mitigating their effects.

This research grounds an experimental investigation in a stage model of interruption management. The Interruption Management model provides a basis for identifying potential influencing mechanisms and determining appropriate dependent measures. The model also provides an organizational framework for basic research relevant to the study of interruption management. Fourteen airline pilots participated in a flightdeck simulation experiment to investigate the general effects of performing an interrupting task, of performing an interrupted procedure, and the effects of specific task factors: (1) modality; (2) embeddedness, or goal-level, of an interruption; (3) strength of association, or coupling-strength, between interrupted tasks; (4) semantic similarity of the interruption and interrupted task; (5) the level of environmental stress.

General effects of interruptions were extremely robust. All individual task factors significantly affected interruption management, except the similarity factor. Results are interpreted to extend the Interruption Management model, and for their implications to flightdeck performance and intervention strategies for mitigating their effects on the flightdeck.

1. Introduction

Statement of problem

Human operators increasingly supervise and manage multiple tasks in complex, dynamic systems (Sheridan and Johannsen 1976). A fundamental aspect of multiple task management is to attend appropriately to and accommodate new, interrupting stimuli and tasks; that is, to engage in *interruption management* (Adams, Tenney, and Pew 1995; Woods 1995; Cooper and Franks 1993; Abbott and Rogers 1993; Funk 1991, 1996). The effects of interruptions are exacerbated in complex, multi-tasking work environments, but even work environments and tasks not typically considered complex, such as a sales office (*e.g.*, Paquoit, Eyrolle, and Cellier 1986), or multi-tasking, such as database navigation (*e.g.*, Field 1987), suffer the consequences of interruptions. Rapid advances in telecommunications technology have dramatically increased interpersonal access and communication. This increased access and convenience of communication also implies an increased potential for interruptions to a wider range of interrupted tasks and task contexts.

Interruptions often negatively affect human performance. Specifically, most laboratory and applied experiments demonstrate that interruptions increase post-interruption performance times (Detweiler, Hess, and Phelps 1994; Gillie and Broadbent 1989; Field 1987; and Kreifeldt and McCarthey 1981) and error rates (Detweiler, Hess, and Phelps 1994; Cellier and Eyrolle 1992; Gillie and Broadbent 1989; Field 1987; and Kreifeldt and McCarthey 1981), increase perceived workload (Kirmeyer 1988), and motivate compensatory behavior (Cellier and Eyrolle 1992; Paquiot, Eyrolle, and Cellier 1986). Recognizing these significant deleterious effects of interruptions, Kreifeldt and McCarthey (1981) suggest that the ability of a human machine interface to mitigate these effects should be explicitly addressed as a usability issue in design. The deleterious effects of interruptions extend beyond these laboratory experiments and usability studies. Interruptions also contribute to serious incidents and accidents in complex systems; for example, power plant incidents (*e.g.*, Bainbridge 1984; Griffon-Fouco and Ghertman 1984), aviation incidents (*e.g.*, Madhaven and Funk 1993; Chou and Funk 1993; Monan 1979; Turner and Huntley 1991), and aviation accidents (*e.g.* NTSB 1988, 1973).

Research Goals

The ubiquity of interruptions, both within and across many work environments, and the associated performance decrements found in both laboratory and operational settings motivates the study of interruptions. Although the larger issue of multiple task management is widely studied and many basic research perspectives are relevant to the study of interruptions, the study of interruptions, *per se*, has not received commensurate attention. Similarly, research investigating multiple task management on the flightdeck receives a great deal of attention whereas only a few studies addressing the influence of interruptions on flightdeck performance exist. Research of interruptions on the flightdeck predominantly describes interruptions as a causal factor in aviation incidents and accidents. Flightdeck simulation studies have addressed issues relevant to the study of interruptions, however, prior to this research effort, interruptions on the flightdeck have not been explicitly, experimentally investigated in a flightdeck simulation environment. The commercial flightdeck is a multitasking environment in which interruptions naturally occur. In addition, the deleterious effects of interruptions in this environment are well documented. The flightdeck, therefore, provides an appropriate, real-world work environment for investigating interruptions and one that could obviously benefit from mitigating their effects.

The goals of this research are: (1) to provide a conceptual model of interruption management, and (2) to investigate factors hypothesized to influence interruption management, (3) to demonstrate the effects of interruptions in a relatively realistic simulation of a naturally multitasking work environment, the commercial flightdeck. This research presents a stage model of interruption management as a foundation for defining effects of interruptions on ongoing task sets, and relating basic research to interruption management. The present study investigates several factors identified by this model and scant previous research directly focusing on interruption mechanisms. These factors include: ongoing and interrupting task modalities, embeddedness of an interruption in an ongoing procedure, perceived coupling of an interrupted task sequence, semantic similarity of the interrupted and interrupting tasks, and environmentally-imposed stress. To investigate the effects of interruptions on flightdeck performance, I develop a simulation of a commercial flightdeck and flight scenario and expose current, commercial airline pilots to realistic Air Traffic Control (ATC) interruptions.

The motivation for this research is to ultimately alleviate the effects of interruptions on the flightdeck through interface design, intelligent aiding devices, and training systems. This research provides a general theoretical approach and empirical evidence of contextual factors affecting flightdeck interruption management toward the development of these interventions.

2. Literature Review

This section reviews previous literature on interruption management generally and in particular with respect to their occurrence to, and affects on, commercial flightdeck operations. I describe basic research perspectives related to interruption management in the context of the interruption management model in section 3.

Observing and Investigating Interruptions

Previous research takes three approaches to investigating interruptions. First, observations demonstrate the incidence and consequences of interruptions in real work environments. Second, applied research evaluates characteristics in interrupted task scenarios human/machine interface. Third, basic studies use abstract tasks and highly controlled procedures to investigate factors that influence the effects in laboratory settings.

Observations of Interruption Incidence and Consequences

Interruptions increase the uncontrollability and unpredictability of an environment, and as such, increase the stress level of any environment (Cohen 1980; Kirmeyer 1988). The resulting deleterious effects of interruptions are obvious in operational environments. In a telecommunications sales office, phone calls from clients and communication from colleagues interrupt operators while they update written materials (Paquiot, Eyrolle, and Cellier 1986). Seventy-seven percent of these interruptions pre-empted operators' performances of ongoing tasks. These interruptions delay performance times for ongoing tasks but do not significantly increase error rates (Paquiot, Eyrolle, and Cellier 1986). Paquiot, Eyrolle, and Cellier (1986) interpret these observations to indicate that operators strategically expand performance time and choose particular integration strategies to minimize increases in error rates. Kirmeyer (1988) observed that police-dispatching radio operators are also frequently interrupted, and the following effects of interruptions. Radio interruptions occur to almost half, 43%, of dispatcher's ongoing, work-related activities. Frequency of interruptions is directly and significantly associated with dispatchers' appraisals of workload and with the number of self-reported coping activities. Self-appraisals of overload and number of coping actions are significantly associated with the frequency of radio interruptions managed in parallel with ongoing routine tasks, but not with preemptive interruptions. Regrettably, neither the relative frequency, nor the conditions under which dispatchers employ these two interruption management strategies are reported. Interruptions are also a causal factor in power plant incidents. Griffon-Fouco and Ghertman (1984) find that interruptions of primary tasks account for more than 25% of the shut-down incidents they surveyed (Paquiot, Eyrolle, and Cellier 1986). Bainbridge's (1984) survey also found interruptions to be a major source of human error in nuclear power plant operations.

Applied Research: Interruption and Human/Machine Interfaces

Given that interruptions naturally occur in the environments in which consumer products are used, and given that interruptions typically degrade performance, it follows that products should be evaluated in, and designed for, realistic contexts, *i.e.*, those containing interruptions. Evaluations in these more realistic task contexts detect differences in interfaces that do not appear in unrealistically stable circumstances (Kreifeldt and McCarthey 1981). Kreifeldt and McCarthey (1981) propose this methodology most explicitly and call for *interruption resistance* as a human/interface design specification. This methodology is used to evaluate reverse-Polish notation (RPN) and algebraic notation (AN) calculators (Kreifeldt and McCarthey 1981), and to evaluate database search techniques (Field 1987) and hypertext structure and search capabilities (McDonald and Stevenson 1996). Although the intent of this line of research is to evaluate interfaces, results suggest factors that influence how operators handle interruptions.

Calculator Design

Kreifeldt and McCarthey (1981) find some similar effects of interruptions on subjects using both the AN and RPN calculators; *i.e.*, similar resumption times, significantly longer performance times on interrupted tasks compared to uninterrupted tasks, shorter resumption times than initial onset times, and no difference between error rates for interrupted and uninterrupted tasks. Kreifeldt and McCarthey's (1981) other results distinguish between these devices. Interruptions cause much slower, over twice as long, interrupted task performance times for the AN calculator than for the RPN calculator. Two factors are confounded in these calculators, the underlying logic system and the display/control interface. These authors suggest that negligible differences between initial and resuming key presses, and between uninterrupted solution times indicate that users could adapt to either logic system. They focus, then, on display differences in the calculator interfaces. The RPN calculator displays user entries differently than resultants, indicating not only interim calculations but also displaying previous operator actions. By externally displaying elements of the problem representation and previous actions, and thereby decreasing the user's internal memory load, these authors suggest that the RPN calculator's interface may facilitate performance following interruption and allow subjects to perform the total ongoing task faster.

Searching Information Systems

Field (1987) considers the efficacy of a selective retreat search facility (which provides the user with a sequential trace of items visited) in comparison to a more restricted retreat search facility (which allows users only to return to the previous screen) in a database application for simple and complex information acquisition tasks. Field's (1987) results are summarized as follows. Interrupted performance significantly differs for search type and task type conditions, indicated by the number of retreats, and the number of screens required to access the target after interruption. Interruptions do not differentially affect performance on different search types or task types as indicated by resumption time or the time to access the target following the interruption. Subjects retreat less and visit fewer screens prior to target acquisition when using the selective retreat facility. There are two possible explanations for this result: (1) The selective retreat facility externalizes more of the prior sequence and, thereby, affords more memory prompting than the restricted retreat facility. (2) The selective retreat facility may help users develop a 'cognitive map' of the system and the resulting improved contextual system knowledge facilitates post-interrupt performance. In addition, Field (1987) demonstrates that interruptions to complex tasks result in significantly more post-interruption retreats and more screen visitations than interruptions to simple tasks. He suggests that the lack of significance in other measures (*i.e.*, resumption time, post-interrupt target acquisition time), that would more conclusively support this point, may be due to insufficiently-different complexity manipulations.

In a similar study, McDonald and Stevenson (1996) investigate hypertext information structures and associated search facilities. Specifically, they compare three structures of the same information; a linear structure, in which nodes appear in sequence and users can only move forward and backward; a hierarchical structure, in which nodes compose a parent-child tree and provides guided exploration and backtracking; and a non-linear structure, that links related nodes as a network and allows users unrestricted navigation and backtracking. Following an interrupting task, users of the linear system located target cards significantly faster and accessed fewer non-target cards than users of either the hierarchical or non-linear hypertext systems. Users of the hierarchical system access fewer cards than users of the non-linear system. McDonald and Stevenson (1996) attribute linear and hierarchical system users' superior performance, *i.e.*, their relative lack of interruption-induced disorientation, to the supposition that users receive better spatial representation of text location, contextual system knowledge, with these systems than with the unrestricted system. Further, they suggest that linear and hierarchical constraints on navigation facilitate post-interruption reorientation by minimizing the number of choices available, and thereby decreasing memory load.

Basic Research on Interruption Mechanisms

Observational and applied research suggests task characteristics that influence interruption management performance, but it does not explicitly manipulate these factors, and as such, causal relationships are tenuous. In fact, very few investigations address this issue directly. This review begins with a historical perspective on investigating interruptions and continues by describing experimentally-identified effects of task, environment, and operator characteristics on interruption management.

Recall and Resumption of Interrupted Tasks

The most extensive line of interruption research stems from the motivational psychology tradition. This research focuses on demonstrating the relationship between interruption and memory, specifically as evidenced by recall for completed vs. interrupted tasks, and for the tendency to resume interrupted tasks. Early experimentation found that subjects are more likely to recollect interrupted tasks than completed tasks (Zeigarnick 1927), and that, even when told it was unnecessary to do so, subjects spontaneously resume interrupted tasks (Ovsiankina 1928). These studies do not, nor do the many studies that attempted to replicate or extend these results, address the degree of effect caused by an interruption (Gillie and Broadbent 1989). Many of the extensions to this work attempt to relate subjects' propensity for recalling or resuming interrupted tasks to psychological traits or instructions that indicate different motivational states[1]. The value of this research is its establishment of the heightened recall and resumption of interrupted tasks, a phenomenon referred to as the "Zeigarnick effect". One motivational psychology theory suggests that working memory load explains heightened interrupted-task recall and resumption effects (Miller, Galanter, and Pribram 1960).

Task Characteristics Influencing Interruption Management

Several studies further investigate the effects of working memory load on interrupted task performance. In a series of experiments, Gillie and Broadbent (1989) attempt to converge on characteristics of interrupted task scenarios that degrade performance. After finding no performance degradation with both short (30 seconds) and long (2.75 minutes) interruption intervals, under conditions that afford rehearsal prior to performing the interruption task, Gillie and Broadbent (1989) conclude that the length of the interruption interval does not influence an interruption's propensity for causing performance degradation. Based on the presence of disruptive effects when memory load at the interruption point is minimal (Kreifeldt and McCarthey 1981), and the lack of deleterious effects in their more memory-loading ongoing task, Gillie and Broadbent (1989) suggest that the memory load associated with the ongoing task's interruption position does not influence the performance effects of an interruption. They further dismiss evidence of a memory load effect in two of their four experiments as an artifact of the experimental task and procedures. In contrast, other researchers find striking evidence that the memory load associated with the interruption position in the ongoing task does significantly influence an interruption's deleterious effects on performance (Hess and Detweiler 1994; Detweiler, Hess, and Phelps 1994). If the ongoing task is a nested equation, it can be represented as a goal structure in which interruptions can be embedded at different levels of memory-loading (Detweiler, Hess, and Phelps 1994). Detweiler, Hess, and Phelps (1994) manipulate memory load by interrupting at two levels of an equation's goal structure, thereby ensuring one (corresponding to low memory load) or two (corresponding to high memory load) intermittent results in memory at the time of the interruption. Interruptions at higher memory load, more embedded, positions can result in less accurate response on the main task and on the interruptions (Hess and Detweiler 1994; Detweiler, Hess, and Phelps 1994); and, if the ongoing task is presented in varied-sequence, resumption delays (Detweiler, Hess, and Phelps 1994). However, other research does not succeed in demonstrating the distracting effects of interruptions as a function of ongoing task goal-level in a more dynamic hierarchical task (Lorch 1987).

Whereas Gillie and Broadbent's (1989) simple, processing-intensive interruptions that allow rehearsal do not degrade performance; a memory-intensive, free-recall interruption task that does not allow rehearsal does degrade performance. This contrast indicates that either the competition for similar resources between the interruption and the ongoing task (both memory-intensive tasks), or the inability to rehearse the interruption point in the ongoing task causes degraded performance. Performance also degrades when interruptions

---

[1] See Van Bergen (1968) for a review of this literature.

are short, complex (*i.e.*, a decoding and arithmetic task) and do allow rehearsal. Given that both this type of interruption and the free-recall interruption cause performance degradation, it is difficult to conclusively determine one source of performance degradation. Rather, it seems that, although resource competition introduced by similar interruptions and ongoing tasks degrades performance, since a dissimilar interruption produces degradation even when rehearsal is provided, a larger phenomenon is at work. Gillie and Broadbent (1989) suggest that rather than similarity or rehearsal, the operative factor may be task complexity, or the amount of information processing required by the interruption. Other evidence suggests that interruption complexity does reduce accuracy (Cellier and Eyrolle 1992).

The similarity of interruption and ongoing task can be defined by either, the resources utilized to perform the tasks, the form of information to be processed (Hess and Detweiler 1994), or the semantic content of the material. Resources associated with architectural components of human information processing are *base resources* (Cellier and Eyrolle 1992). In contrast, *constructed resources* are associated with semantic knowledge (Cellier and Eyrolle 1992). Gillie and Broadbent (1989) interpret their results in terms of similarity of processing resources. Interruptions similar to the ongoing task in terms of both processing resources and information form, produce less accurate performance (Hess and Detweiler 1994; Detweiler, Hess, and Phelps 1994), and result in longer resumption delays (Detweiler, Hess, and Phelps 1994) than interruptions only similar in terms of processing resources. This effect holds for interruptions with both relatively high and low memory capacity requirements (Detweiler, Hess, and Phelps 1994), suggesting that similarity of material type significantly influences the degree to which an interruption degrades performance. Semantic similarity does not influence response times or accuracy when similarity manipulations rely on distinctions among the sets of; even and odd numbers, numbers over 50, letters, and vowels and consonants (Cellier and Eyrolle 1992).

Although the direction of the base-resource similarity effects are fairly robust, other factors influence whether these effects are significant. Specifically, the effects of base-resource similarity are most evident when interruptions occur at interruption positions that induce a high memory load and when rehearsal is restricted (Hess and Detweiler 1994; Detweiler, Hess, and Phelps 1994). Although rehearsal may mitigate the influences of some interruptions, allowing rehearsal does not necessarily guarantee that interruptions do not degrade performance (Gillie and Broadbent 1989). Whereas Gillie and Broadbent's (1989) experiments manipulate rehearsal by using different interrupting tasks, Detweiler and colleagues (Hess and Detweiler 1994; Detweiler, Hess, and Phelps 1994) explicitly prompt rehearsal by providing warnings to subjects and instructing them to remember their place in the ongoing task. This difference in experimental conditions may explain seemingly contradictory observations on the protective powers of rehearsal.

Other factors that influence the degree of performance degradation induced by an interruption correspond to the ease with which one performs the ongoing task. Detweiler, Hess, and Phelps (1994) compare consistent-sequence presentation to a varied-sequence presentation of the information required to perform the ongoing task. Commensurate with previous research on learning with consistent and varied mappings, interruptions do not degrade performance on ongoing tasks with consistent-sequence information presentation as much as they do to ongoing tasks with information presented in a varied sequence. Further, significant differences in resumption time attributed to the memory load of the interruption position are only evident in varied-sequence conditions. By presenting ongoing task information in a consistent sequence, performance on the ongoing task becomes more efficient, less memory-demanding, and thereby makes available more resources for interruption management (Detweiler, Hess, and Phelps 1994). Even with extended practice on a consistent-sequence task, interruptions can still degrade performance (Hess and Detweiler 1994). However, if allowed the same extent of practice on the ongoing task with intervening interruptions, subjects' performance becomes resistant to the deleterious effects of interrupts (Hess and Detweiler 1994). Performance is not dependent on whether the ongoing task set has a random, or free, order; a fixed and logical order; or a fixed but arbitrary order (Gillie and Broadbent 1989).

Environmental Characteristics Influencing Interruption Management

Time constraints on task performance also affect interruption management. Abrupt interruptions to an ongoing task with high time constraints appear to actually speed performance on the interrupted task, but also impair accuracy of its performance more than interruptions to tasks with more relaxed time constraints (Cellier and Eyrolle 1992). This result might indicate that additional resources are activated under greater time constraints, speeding performance, and, due to a speed/ accuracy trade-off, result in decreased accuracy (Cellier and Eyrolle 1992). Alternatively, subjects may strategically assume that, in low time constraint conditions, the timeliness of performance is assured and therefore the goal is to improve accuracy; whereas, under higher time-constraint conditions, the primary goal is to assure timely performance (Cellier and Eyrolle 1992). Recall that sales personnel also seem to strategically alter performance in one dimension, they extend overall performance time, to achieve another performance goal, reduction of errors (Paquiot, Eyrolle, and Cellier 1986).

Operator Characteristics Influencing Interruption Management

Operator characteristics also have the potential to affect interruption management. In addition, to individual strategy development as previously implied, both personality type and cognitive style influence performance in interrupted situations. An individual's Type-A / Type-B personality classification can predict how they respond in work situations with many interruptions (Kirmeyer 1988). Type-A personalities are characterized by hostility-aggression, impatience or time-urgency, and striving for competitive achievement; whereas, Type-B personalities are more patient, easygoing, and noncompetitive. Kirmeyer (1988) classifies 72 police radio dispatchers as either Type-A or Type-B personalities and associates this personality characteristic with dispatchers' self-appraisals of work overload and the number of coping actions taken in response to with interruption rates. Type-A personalities are more likely to appraise work level as overload and report that their controlling actions increase with the incidence of interruptions. Although no evidence exists that Type-A personalities are interrupted more frequently, the data collected in this experiment is insufficient to rule out the possibility that differences in perceived work overload are not simply reflective of actual differences in work levels or work responsibilities (Kirmeyer 1988).

Jolly and Reardon (1985) associate an aspect of cognitive style, field-dependency, with interrupted task performance. Field-dependency refers to the ability to rapidly reorient assignments of stimuli to different cognitive or mental processes (Braune and Wickens 1986). Field-dependent individuals appear to be more disadvantaged by interruptions in over-learned, automatic procedures (Jolly and Reardon 1988). Field-independent individuals distinguish between task-relevant and task-irrelevant materials more assuredly and use task-relevant materials to reorient to the primary task following interruption.

Interruptions on the Flightdeck

The flightdeck is a complex and dynamic multitasking environment in which pilots increasingly supervise and manage higher-level automated processes rather than continuously monitor and control individual flight parameters. External and aircraft events, as well as interactions with other operators, compete for pilots' attention and require pilots to integrate performance requirements associated with these unexpected prompts with ongoing flightdeck tasks. Interruptions, therefore, are a recognized facet of multiple task management on the flightdeck. Several incident and accident investigations implicate interruptions as a contributing factor. Although the significant incidence and, potentially, severe consequences of interruptions are obvious, experimental research directly investigating interruptions on the flightdeck is sparse. The following sections describe the role of interruption management in the context of multiple task management on the flightdeck, incident and accident investigations implicating interruptions as a causal factor, and experimental research related to investigating interruptions on the flightdeck.

Interruptions in the Flightdeck Context

Task management is one of four flightdeck functions, on *par* with flightdeck management, communications management, and systems management (Abbott and Rogers 1993). While equal in consideration to the other critical functional categories, task management functions are, by definition, interstitial to these other categories. In this definition, task management activities both supervise and support flight management, communications management, and systems management functions, and provide the underlying mechanism for coordinating their requirements. Further, task management on the flightdeck requires monitoring, scheduling, and resource allocation. The scheduling sub-function determines the task sequence to be executed based on task priority, resource availability, and temporal constraints. The scheduling sub-function also includes dynamic alterations of task sequence in response to external cues that trigger the onset of a context-dependent task, interruption of a new task, or resumption of a pending task. This conceptualization of flightdeck functions explicitly indicates the role of interruption handling in task management and the role of task management in the context of other flightdeck functions.

Funk (1996) explicitly extends an earlier conceptualization of cockpit task management (Funk 1991) to include management of not only tasks performed by human operators, but functions and goals of all *actors* on the flightdeck. This extension defines an actor as any entity capable of goal-directed activity, including monitoring and controlling mechanisms such as autopilots, flightpath management systems, and automated caution and warning systems. In a decompositional normative model of flightdeck task management, pilots actively manage an *agenda*, a set of goals, functions, actor assignments, and resource allocations (Funk 1996). Major components of Funk's model include maintaining situation awareness, managing goals (recognizing, inferring, and prioritizing), managing functions (activating, assessing status, and prioritizing), assigning actors (goal-directed entities) to functions, and allocating resources (*e.g.*, displays and controls) to functions. According to this normative model, interruptions are managed by rational consideration of resource availability and relative task priorities.

Observations in Aviation Incidents and Accidents

Interruptions pose a significant problem on the flightdeck. This section describes evidence from both incident reports and accident investigations that indicate the incidence and consequences of interruptions on the commercial flightdeck. Surveys of aviation incidents are based on voluntary, anonymous pilot reports to the Aviation Safety Reporting System (ASRS). Accident investigations are conducted by an independent source, the National Transportation Safety Board (NTSB).

One hundred and sixty-nine, almost 7%, of the 2500 ASRS reports collected to 1979, referred to an interruption as a significant cause of the reported incident[2] (Monan 1979). These cases include two categories of interruptions; *non-operational* interruptions, *i.e.*, tasks not required for flight operations and *operational* interruptions, *i.e.*, outcomes of routine flightdeck tasks that, when performed at inappropriate times, result in excessive workload. The causes of non-operational interruptions and the number of cases attributed to these factors are as follows; performing paperwork (7), using the public announcement system (12), crew member conversation (9), flight attendant conversation (11), and company radio contact (16). Causes of operational interruptions and the number of cases attributed to these factors are; checklist performance (22), malfunctions (19), watching for traffic (16), ATC communications (6), radar monitoring (12), referencing approach chart (14), looking for the airport (3), monitoring new first officer (10), fatigue (10), and miscellaneous interrupts (2). Although some of these interruptions are internally-induced, and therefore do not exactly reflect the type of interruptions examined by the current research, the incidence of externally-induced interruptions and their consequences of interruption are clear. In these incidences, interruptions cause several operationally-significant errors; altitude excursions, lack of cross-check of crew actions, landing without clearance, mistakenly taking a clearance intended for another aircraft, misinterpretation of a clearance, unauthorized entry into an active runway, failure to adequately take see-and-avoid actions, deviations from route, penetration of restricted airspace, failure to reset altimeter, non-stabilized approach, and an approach to a wrong airport (Monan 1979).

---

[2] Monan (1979) describes this phenomenon as distraction, however the nature of the phenomenon he investigated included not only momentary attentional deflections, but also implied an associated task. For this reason, this research is considered evidence of the effects of interruptions.

Results from Monan's (1979) survey instigated the FAA's *Sterile Cockpit* rule in 1981 which reads as follows: "No flight crew member may engage in, nor may any pilot in command permit, any activity during a critical phase of flight that could distract any flight crew member from the performance of his or her duties or that could interfere in any way with the proper conduct of those duties [FAR 121.542 (b) and FAR 135.100 (b)] (Barnes and Monan 1990)." These rules also identify various non-essential flightdeck functions and define critical flight phases as "all ground operations, including taxi, takeoff and landing, and all other flight operations conducted below 10,000 feet, except cruise flight (Barnes and Monan 1990)." Barnes and Monan (1990) verify that not only is the Sterile Cockpit Rule occasionally broken, but even well after this rule's installation, pilots continue to cite interruptions as a causal factor in aviation incidents. They cite partially completed analyses indicating that 65% of interruptions are due to events central to safe flight, 35% are due to events peripheral to safe flight, and 5% to social or personal matters.

Turner and Huntley (1991) analyzed 195 ASRS aviation incident reports in an investigation of checklist usage. Fifty-eight percent of these reports cite interruptions as a causal factor. Of this 58%, approximately half are due to interruptions of checklist performance (*e.g.*, ATC calls), and half due to the performance of a checklist interrupting an operational task (*e.g.*, maintaining position in a departure queue). These interruptions result in the following operational performance effects; exceeding altitude by several thousand feet, failure to reset the altimeter, and almost departing without retracting a spoiler (Turner and Huntley 1991). Degani and Wiener (1990) observed commercial pilots using checklists in normal operations and also found deleterious effects of interruptions to checklists; specifically, elimination of the vital cross-checking function of one crew-member, disruption of the checklist's sequence, and increased memory load associated with remembering the interruption position. Interruptions to checklists are so bothersome that pilots adapt methods for visually representing the resumption point on a checklist; *e.g.*, by placing their thumb at the interrupted position, by writing down the number of the interrupted item, or by checking off items as they are performed (Degani and Wiener 1990). Subjects' development of these adaptive behaviors suggests that pilots perceive interruptions to checklists as opportunities for performance degradation.

Flightdeck task management errors include: (1) task initiation, early, late, incorrect, lack thereof ; (2) task monitoring, excessive, lack thereof; (3) task prioritization, high, low; (4) resource allocation, high, low; (5) task termination, early, late, incorrect, lack thereof; (6) task interruption, incorrect; and (7) task resumption, lack thereof (Chou and Funk 1990, 1993). Chou and Funk (1993) find 98 cockpit task management errors in 77 accident reports[3]. The relative percentages of these errors attributable to their cockpit task management error categories are: task initiation (37.8%), task monitoring (22.4%), task termination (21.4%), resource allocation (8.2%), task interruption (5.1%), task prioritization

---

[3] Chou and Funk (1993) examined a previously-defined set of 324 NTSB reports from years 1960 - 1989 and eliminated from consideration those reports that were unrelated to the study, for example, those caused by obvious weather and catastrophic equipment failure.

(4.1%), and task resumption (1.0%). Although the percentage of errors attributable to the task interruption category seems less significant than other forms of human errors in flightdeck task management, it is important to consider the strict definition of this category in Chou and Funk's (1993) scheme. Cockpit task management errors attributed to task interruption in this scheme are only those that include an inappropriate interruption of an ongoing event. In the larger context, the effects of an interruption might also instigate errors of task resumption, task initiation, task termination, and task prioritization. Madhaven and Funk (1993) collapse the task interruption error category into the task prioritization category. This modification assumes that task prioritization decisions determine interruptions, and assumes that an inappropriate interruption results from faulty prioritization. An analysis of 20 ASRS incident reports[4] according to the modified task management error taxonomy revealed 19 task initiation errors, 18 task monitoring errors, 8 task prioritization errors, and 8 task termination errors (Madhaven and Funk 1993).

Summary reports of aviation incidences indicate interruption sources and performance effects associated of interruptions on the flightdeck, however they do not convey the potentially catastrophic nature of such effects. A Northwest Airlines aircraft in Detroit Metropolitan Airport crashed almost immediately after takeoff due to improper configuration, the trailing edge flaps and leading edge slats were fully retracted (NTSB 1988). One contributing factor in this accident appears to be interruption by ATC communication during the taxi checklist, which contains an item for flap setting. Only one of the 155 persons on board this flight survived. If one considers a system failure a form of interruption to ongoing tasks, the accidents attributed to this problem are even more pronounced. Several accidents are attributed to crews poorly integrating performance requirements for handling an interrupting system alert and compensatory actions with other aviation tasks. For example, on an Eastern Airlines flight, the crew became so engaged in diagnosing a suspected landing gear malfunction, that they failed to monitor instruments and did not detect a rapid descent in time to prevent impact (NTSB 1973). Ninety-nine of the 176 passengers did not survive this accident (NTSB 1973).

Empirical Investigations on Flightdeck Task Management

Although several studies address the more general problem of instrument scanning and multiple task management on the flightdeck, and many aircraft simulation studies could be interpreted, *post hoc*, for effects due to interruptions in the scenarios, only a few experimental investigations address the effects of interruptions *per se*, and none have yet explicitly manipulated characteristics of interruptions with this intent. Studies of multiple task management on the flightdeck indicate, albeit indirectly, the significance of interruptions and some factors that may affect interruption management on the flightdeck. Scenarios with interruptions and multiple tasks induce deeper planning in flightdeck crews than scenarios without these complications (Johannsen and Rouse 1984). Also, task

---

[4] Madhaven and Funk (1993) selected these 20 ASRS reports from a previous compilation of 206 Controlled-Flight-Towards-Terrain reports and 99 In-Flight-Engine Emergency reports. Reports were selected which gave evidence of more than one cockpit task management error type.

prioritization errors increase with the number of concurrent tasks and flightpath complexity (Chou and Funk 1993).

Wickens and colleagues (Raby, Wickens, and Marsh 1990; Raby and Wickens 1990; Raby and Wickens 1991) assert that pilots shed tasks in high workload conditions according to priority assessments as a means of strategically managing workload. Failures in accurately assessing task priorities may result in inappropriate task interruption or resumption (*cf.* Madhaven and Funk 1993). In addition to task priority, Segal and Wickens (1991) propose six factors that they hypothesize might affect the probability that a pilot irrationally pre-empts an ongoing task for another task. These factors include: (1) task modality, auditory tasks are more likely to pre-empt ongoing tasks than visual tasks (Kramer *et al.* 1991; Wickens and Liu 1988); (2) task salience, tasks whose triggering events are loud, bright, or dynamic will be more likely to pre-empt, (3) task difficulty, easier tasks may be more likely to pre-empt ongoing activity than more difficult ones; (4) task performance time, tasks that can be performed rapidly may be more likely to pre-empt than those anticipated to take longer; (5) task arrival-time, recently-arrived tasks may be more likely to pre-empt ongoing tasks (Segal and Wickens 1991). The context created by the set of tasks serves as the foundation for determining relative levels of salience and difficulty (Segal and Wickens 1991). Although these factors are suggested to influence pilots' propensity for switching among a set of already ongoing tasks, these may be extended to influence the probability of switching from an ongoing task to an interrupting task. These factors, suggested as hypothetical influences on multiple task management, are not systematically experimentally tested.

In summary, Wickens and his colleagues' work indicates the relative difficulty of flightdeck operations during multitasking scenarios and indicates specific factors affecting task management behavior. These factors include; operator characteristics, *i.e.*, current and projected workload levels, and assessment of task and environmental characteristics; task characteristics, *i.e.*, priority, modality, salience, difficulty, performance time; and environmental characteristics, *i.e.*, predictability and temporal constraints. These studies demonstrate or propose factors pertinent to flightdeck multiple task management in general. They are presented here as potentially influential factors for predicting interruption management in particular.

Empirical Research on Interruptions to Flightdeck Tasks

Despite the potential consequences and incidence of flightdeck interruptions, it is surprising that only two studies directly address their effects experimentally. One study addresses the effects of interruptions on checklist usage (*i.e.*, Linde and Goguen 1987) and another investigates the effects of datalink interruptions to FMS/CDU tasks (*i.e.*, Williams 1995). The purpose of these studies is to evaluate how pilots perform procedures and use equipment when interrupted. However, neither of these studies focus on the characteristics of interruptions or their relationship to the interrupted task context, nor do they experimentally manipulate interruption conditions. This section reviews these studies for evidence of factors that influence interrupted task performance on the flightdeck.

Interruptions and Checklist Usage

Airline training programs typically suggest that a checklist should not be initiated until it can be executed without interruption from other activities (Linde and Goguen 1987; Degani and Wiener 1990). If a radio transmission occurs during checklist performance, the crew is to ignore it until the checklist is done. If a checklist must be interrupted, an explicit hold should be placed in the checklist by saying "Hold it at (name of checklist item)." When the captain says "Continue the checklist," checklist performance resumes at the point of interruption. Whereas Turner and Huntley (1991) and Degani and Wiener (1990) demonstrate specifically the deleterious effects of interruptions to checklists, Linde and Goguen (1987) evaluate whether expert-ratings of crew quality, that is safe performance, are associated with interrupted checklist performance. They use a subset of flight simulation data from a separate experiment (Murphy *et al.* 1984) in which 16 crews flew a full mission scenario, including weather and equipment problems. Expert pilots rated 14 of these crews on overall safety of performance. Linde and Goguen (1987) determine if the most safe 7 crews could be distinguished from the least safe 7 crews by their performance on linguistically-defined variables of checklist performance.

Linde and Goguen (1987) demonstrate the following results. Although crews are trained to ignore interruptions until a checklist is complete, pilots actually pre-empt, on average, 28% of interrupted checklists. High continuity ratios (the number of checklist speech acts divided by the total speech acts during checklist span) are desirable, and are demonstrably associated with the safer crews. The total number of interruptions per checklist does not distinguish between safe and less-safe crews. Effective cockpit resource management (CRM) dictates that the pilot who is flying should call to resume interrupted checklists. However, pilots responsible for flying resume interrupted checklists with roughly the same frequency in both groups of crews. Flight engineers resume most, 63%, of the checklists in both safe and less-safe crews. Explicit holds are rarely used, but the only two crews who did use explicit holds were two of the three best crews. Crew quality is associated with the length of the interrupt, but neither the definition of this measure, nor the magnitude or direction of effect are obvious.

Linde and Goguen's (1987) conducted this research to identify the potential of linguistic measures to more sensitively evaluate checklist performance and indicate overall crew safety. Although not the central focus of the present study, their results provide evidence of the effects of interruptions on the flightdeck. That pilots sometimes respond to interruptions counter to their checklist and CRM training suggests that intrinsic characteristics of the interruption or interruption position make some interruptions more destructive to flightdeck performance than others. Linde and Goguen (1987) discuss the limitations of training to mitigate effects of checklist interruptions in light of the fact that, for some interrupting conditions, none of the observed crews adhered to the procedure to not interrupt checklist performance.

Datalink Usage and Interruptions

Datalink technology provides a means of communicating between air traffic control (ATC) and flightdecks beyond the current radio/telephone medium. While the concept of datalink communication is not new, increasing radio frequency congestion and technological advancements have spurred increased development of datalink in recent years. Datalink allows digital communication between these two system elements, and therefore provides the opportunity to present visually what is currently aurally-presented information to the flightdeck. As many ATC communications interrupt ongoing flightdeck activities, comparisons of datalink communication and radio communication suggest the importance of interruption modality in interruption management performance.

Most of these comparisons are based on measures of response-time to messages, total time spent communicating, number of communication transactions, and subjective measures of workload and operational acceptability (Kerns 1990). A synthesis of 15 datalink simulation studies, using a variety of interface implementations, finds that; on average, pilots require approximately 10 seconds to read and acknowledge a datalink message and that pilots more rapidly acknowledge datalink messages than radio calls (Kerns 1990). These studies also demonstrate that datalink qualitatively changes pilot / ATC communication and, although no overall workload difference is universally observed, it significantly alters the distribution of workload compared to radio communication. These are only general results; pilot performance is likely affected by the different datalink interface implementations and scenario conditions used in these studies. For example, two studies found that mean response times appear to decrease with altitude and distance to runway (Diehl 1975; Waller and Lohr 1989). It is therefore, difficult to directly ascertain the effects of interruption modality on interruption acknowledgment times or workload effect from these studies.

This previous research, however, does not consider ATC messages as interruptions to ongoing flightdeck activities and therefore does not consider the larger question of how differences in datalink and radio communication might influence not only interruption acknowledgment time, but measures associated with integrating this interruption and propagation effects of an interruption so induced. One comparison (Williams 1995) of a display-shared datalink system and radio communication differs from other datalink investigations by recognizing ATC messages as interruptions to ongoing tasks and considering resumption time as a dependent measure. The datalink system shares the control/display unit (CDU) with that used by the flight management system (FMS). This investigation compares performance of other routine tasks requiring the FMS when ATC clearances are issued visually, on the FMS/CDU datalink, to performance when ATC clearances are issued aurally, by radio. Ten crews perform a full mission scenario that includes a diversion to an alternate airport due to equipment malfunction, and therefore many opportunities for ATC communications. Pilots' performance with datalink and radio communications were characterized by measures of; total number of FMS/CDU button pushes for normal and non-normal flight operations, communication procedure changes, differences between pilot-flying and pilot-not-flying, the number of interruptions occurring to FMS/CDU tasks, and the time to resume after an interruption. The incidence of interruptions and the resumption time after an interruption were determined from videotapes of the scenarios. Observed interruptions were classified according to: (1) the type of task they interrupted (briefings, normal FMS/CDU operations, checklists, other communications, and miscellaneous), and (2) the crew member interrupted (pilot-flying, pilot-not-flying, both).

The modality of ATC clearances does not affect the number of FMS/CDU button pushes associated with normal or non-normal operations, nor does it affect the propensity for interruption. Modality does affect, however, the resumption time from interruptions; resumption after a datalink interruption takes longer than after a radio interruption. The propensity for interruption is also significantly associated with crew member and task type. Results suggest that pilots adhere to cockpit resource management (CRM) strategy to protect the pilot who is flying from interruptions but if both pilots become engaged in the interruption, contradicting CRM training, resumption times are significantly longer. Resumption times are particularly extended if both crew members are engaged in a datalink interruption. Although interrupted task type significantly predicts propensity for interruption, no causal effect is clear since interruptions were not experimentally controlled to interrupt certain task types. This factor is included in recognition that the characteristics of interrupted tasks might be significant, but does not explain differences in propensity for interruption among the task types or include interrupted task type in analysis of resumption times (Williams 1995).

The goal of the above experiment is to evaluate performance effects of competing interfaces in a relatively realistic scenario and consider, in particular, effects on FMS/CDU usage. Toward this end, these results provide mixed evidence, for the viability of a FMS/CDU implementation of datalink and suggests further research is required, specifically to determine the consequences of increased pilot-flying interaction and resumption delays imposed by the datalink implementation. These results provide more generally-useful evidence for understanding interruption management. Interruption modality significantly affects interruption resumption time and some interruptions can cause crew members to depart from CRM practices (Williams 1995).

Summary

Prior to the present study, Linde and Goguen's (1987) and William's (1995) work defines the status of research experimentally addressing the effects of and factors influencing interruptions on the flightdeck. Their work, in conjunction with observations of flightdeck interruption consequences and incidence, indicates the necessity for expanding this line of research to a more controlled, intentionally-manipulated experiment of hypothesized influential factors on flightdeck interruption management. This research experimentally investigates several specific hypothesized effects of interruptions on a commercial flightdeck in a simulated environment.

More generally, the current state of investigation of interruption management suffers from three fundamental problems. First, few studies exist that explicitly attempt to identify the degree to which, task, environment, and operator characteristics degrade performance, particularly in operational environments. Second, reviewing the handful of studies that directly relate to interruption management, makes obvious that there is no common perspective on what interruption management is; what processes it involves, what forms it might take, and how interruptions may affect ongoing tasks. Third, although there are few studies that specifically address this phenomenon *per se,* much research contributes useful perspectives on this phenomenon. However, these separate perspectives have not been identified and interpreted in terms of interruption management. I present a theoretical approach of interruption management as an initial contribution towards eliminating these deficiencies.

3. A Theoretical Approach to Interruption Management

I propose a theoretical model of interruption management based on basic research and previous research on interruption management. With the provision of this theoretical foundation, future investigations of interruption management, such as the empirical investigation herein, may better address the first issue noted above.

A Model of Interruption Management

This theoretical human information processing model formalizes interruption management behavior. This formalization enables definition of specific interruption management behaviors and their effects on ongoing tasks. Further, the model provides a structure for organizing basic research theory and empirical results for the purpose of better understanding the nature and effects of interruptions. Prior to presenting the model, I discuss the interrupted task paradigm for which the model was developed, present the information processing constructs employed by the model, and describe constraints of the model.

Interrupted Task Paradigm

The proposed model assumes certain ongoing and interrupting task and environmental characteristics. These assumed task and environmental characteristics are also incorporated into the experimental scenarios of the empirical investigation. Specification of ongoing and interrupting task characteristics affords a more specific model, but also limits its generalizability to a subset of realistic interruption situations.

Characteristics of the Ongoing Task Set.

The ongoing task set is a finite series of familiar, discrete tasks, heretofore referred to as the *ongoing procedure*. The ongoing procedure can be characterized as a goal-hierarchy and includes strict sequential constraints on constituent task performance. Tasks are said to be composed of activities, which are at the keystroke level. The ongoing procedure requires controlled processing for execution and therefore, this model does not apply to interruption of automated ongoing task sets (*e.g.*, Schneider and Shiffrin 1977; Shiffrin and Schneider 1977). Nor is it applicable to continuous control or monitoring processes, or simple, repetitive tasks with unspecified terminating conditions, because the definitive interruptability of these processes is questionable (*cf.* Adams, Tenney, and Pew 1991; Lewin 1926, 1951; Miller, Galanter, and Pribram 1960). Once interrupted, ongoing procedures are assumed to be resumable from the interruption position.

Characteristics of the Interruption

Interruptions are familiar and, although not incongruous with general expectations of a scenario, are not necessarily expected and are temporally non-deterministic. Interruptions comprise an annunciation stimulus and an associated interrupting task that must eventually be performed. The annunciation stimulus performs two functions. It serves as a *sign* of environmental change to the operator, and *signals* the associated interrupting task performance requirements (Rasmussen 1986). As interruptions are familiar, annunciation stimuli are readily interpreted to identify the interrupting task and associated performance requirements, obviating the need for complex diagnosis and response planning. The interrupting task is at the same level of tasks of the ongoing procedure and also requires controlled processing. The occurrence of an interruption to the ongoing procedure does not affect the performance requirements of the procedure. Interruptions are not concurrent. While multiple and concomitant interruptions might be conceived of as overlaying depicted processes, this circumstance is not explicitly considered for purposes of clarity.

Ensemble Task Set Characteristics

This interrupted task paradigm assumes that operators intend to perform all tasks in the ongoing procedure and the interrupting task. The complete set of performance requirements includes both performance requirements of the ongoing procedure and the performance requirements of the interrupting task. In total, I refer to these performance requirements as the *ensemble task set.* Finally, ensemble tasks exist in an environment that requires regular situation monitoring and assessment and that may impose stress on ensemble task performance. Specifically, if a deadline condition exists for the ongoing procedure, interrupting tasks to that procedure must also be performed within that deadline.

Form of the Model

Most basically, interruption management entails, detecting the annunciation stimulus, interpreting the stimulus in terms of the interrupting task performance requirements, and integrating the interrupting task and the ongoing procedure tasks for performance. The model further embellishes on this simple behavioral description by presenting familiar abstractions of mental processes involved in interruption management. These abstractions are: perceptual processors; sensory, working, and long-term memory stores; plans and intentions; mental operators; and attentional resources.

These simplified definitions suffice for the purpose of introducing the processing stages of this interruption management model. *Perceptual processors* filter the overly abundant environmental sensory array to transfer salient stimuli to a volatile, *sensory memory* that veridically represents the stimulus. These processes and initial storage do not require attention resources. *Working memory* contains information actively used at the moment. It can contain either attended sensory memory information or retrieved information from long-term memory. Working-memory is code-specific and requires attention resources to maintain. *Long-term memory* contains abstract representations of declarative and episodic knowledge. Transfer to and retrieval from long-term memory requires working memory and attention resources. These three memory stores can alternatively be described, not as bins, but memory that is “activated” to lesser degrees (*e.g.*, Cowan 1993; Anderson 1983) by attention resources. According to this description, information in the current attention focus is the most activated subset of working memory, working memory is the most activated subset of long-term memory, and activation level depends on recency of and relevancy for use. A *plan* is taken to be a memory-resident decompositional goal hierarchy, from a most abstract goal to action specifications that guides behavior for the ongoing procedure in this model's structured task environment. *Intentions* are an abstract notion implying the goal-directed nature of cognition and can be conceived of as a motivational force for completing a plan, or, alternatively, as a working memory representation of plan progress. *Mental operations* serve, conceptually, as an interface among processors for the purpose of problem-solving and decision-making, *e.g.*, choice selection and response planning. Problem-solving and decision-making are attention and working-memory intensive. *Attention* is an abstract notion of a limited, and, to some degree, differentiated, divisible, and directable resource required in varying amounts for intentional environmental sampling, controlling goal-directed behavior, maintaining, translating, and accessing memory representations, executing controlled response plans, and conducting mental operations. Proposed mechanisms underlying these human information processing features are presented more fully in the context of reviewing basic theory and research supporting the interruption management model.

Constraints of the Model

The interrupted task paradigm constrains, to some degree, the application of the interruption management model. Therefore, some naturally occurring interruption situations may not generalize directly from the interruption management model presented here. The proposed model is also limited in that it does not describe a validated psychological process. Rather, the purpose of this interruption management model is to provide a parsimonious description of information processing stages involved in interruption management, to describe interruption management behaviors and effects on ongoing task performance, and to offer insight into factors that might influence interruption management performance. This intent constrains usage of the model, and the situations to which it generalizes.

In order to structure the discussion of relevant basic research, the model casts interruption management as a high-level information processing stage model with attention resources (Massaro and Cowan 1993). It assumes certain components of a cognitive architecture as a means for discussing generally-accepted characteristics of human information processing. It does not suggest that the mechanisms described are the singular or *de facto* , preferred means of explaining observed behavior. Nor does it presuppose any particular representation of these processes[5]. Therefore, it claims not to identify underlying mechanisms of mental processes but rather considers these as intervening variables that are useful for describing potentially important distinctions in interruption management behaviors (Van der Heijden and Stebbins 1990). This model depicts the flow of an interruption from its occurrence to re-stabilization of ongoing task performance. It depicts interruption management as sequential stage processing. More likely this is a more continuous process (*e.g.*, Eriksen and Shultz 1978; McClelland 1979) and includes feedback and feedforward mechanisms (*e.g.*, Loftus and Mackworth 1978).

---

[5] For a cogent discussion of the implications for representing interruptability in symbolic, connectionist, and hybrid computational models of cognition, see Cooper and Franks (1993).

Formalizing Interruption Management

A stage model formalizes the process of interruption management (Figure 3.1). I first describe the processing stages of the interruption management stage model. Formalization of the interruption management process identifies distinct effects an interruption may have on the ongoing task. I then define four general effects interruptions may have on the ongoing procedure and describe these effects in terms of their loci in the model.

Interruption Management Processing Stages

The stages of interruption management include: interruption *detection*, interruption *interpretation*, interruption *integration*; and terminate with continued ongoing task performance.

Interruption Detection

Operators are engaged in an ongoing procedure prior to the arrival of annunciation of an interruption. Initial conditions of the model propose that activated memory contains representations associated with the ongoing procedure, and, in particular, those associated with the current task. At the first stage of the model, an annunciation stimulus heralds the interruption. If this stimulus is salient enough to overcome sensory thresholds, it is stored in short-term sensory stores for further processing. This processing stage is *detection* of the annunciation stimulus.

Interruption Interpretation

Successful detection directs attention to the annunciation stimulus for further processing. By mapping the annunciation stimulus to representations in memory, the operator translates the annunciation stimulus to a working memory representation of the interrupting task in terms of its performance requirements. This translation is defined as the *interpretation* of the interruption annunciation. Working memory now supports both representations associated with the ongoing procedure, specifically the interrupted task, and the interruption.

Interruption Integration

Given that the annunciation stimulus is correctly interpreted in terms of the interrupting task's performance requirements, the next stage requires *integration* of these additional performance requirements with those previously defined by the ongoing procedure. Integration includes sub-stages of ongoing task *preemption*, interruption *performance/scheduling*, and ongoing task *resumption*. *Preemption* may occur spontaneously or may result from a deliberate weighing of performance benefits associated with performing the interruption against costs of continuing the interrupted task. To a lesser degree, this deliberate consideration is a preemption of sorts, as it draws attention and computational resources. Interruption *performance* may occur as a direct result of

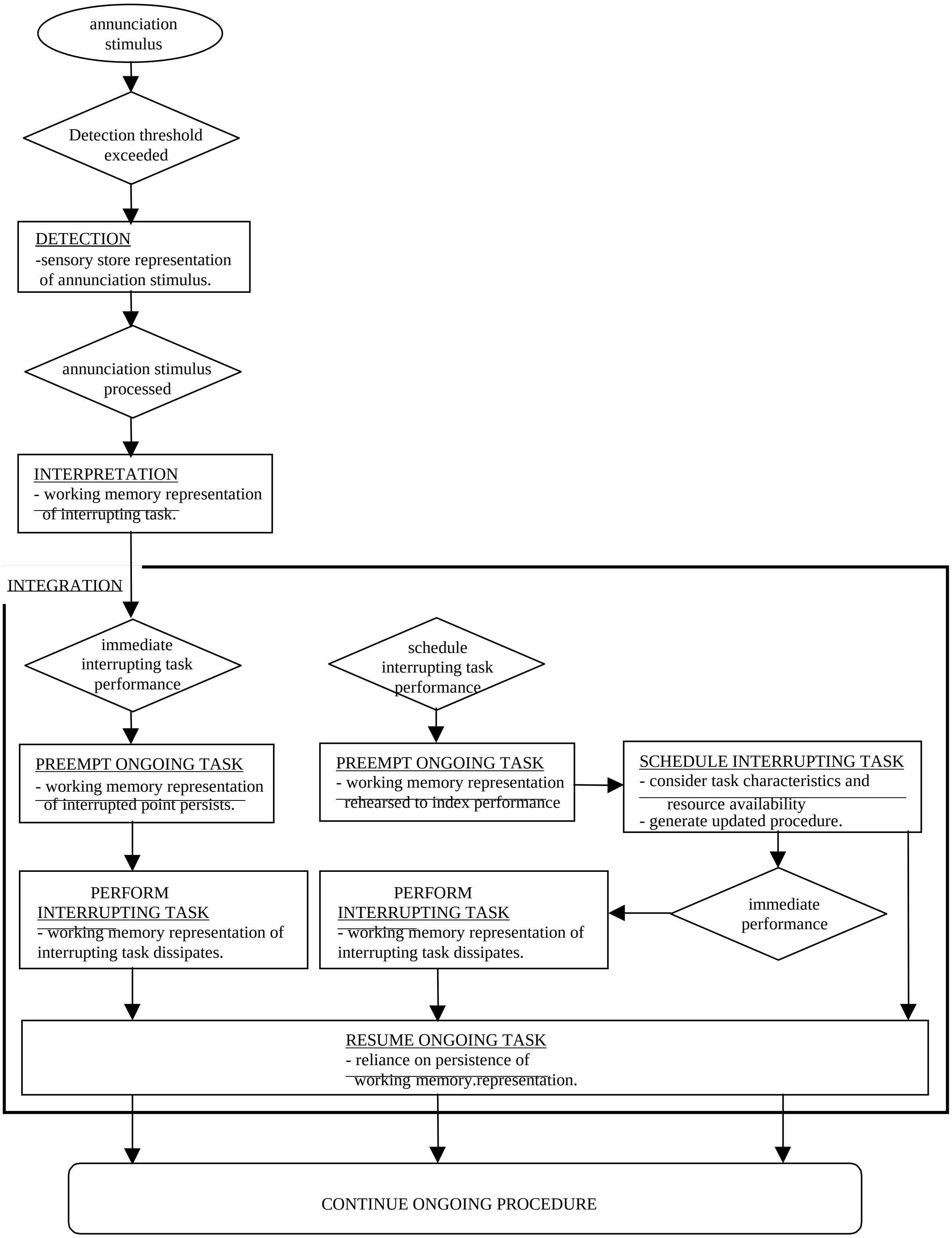


Figure 3.1. Proposed Stage Model of Interruption Management

preemption, or may be deliberately delayed and *scheduled* into the future task requirements. After either performing the interruption or actively scheduling its performance, the operator identifies the *resumption* point in the interrupted task and continues its performance.

Effects of Interruptions on the Ongoing Task

The model identifies four general effects of interruptions; *diversion*, *distraction*, *disturbance*, and *disruption* (Figure 3.2). This section describes these effects in terms of their loci in the interruption management model. Definition of these effects leads directly to dependent measures of interruption management, which are presented here conceptually and defined operationally in terms of this experiment in section 5.6.

After detection of a stimulus, the operator is *diverted* from the ongoing procedure. Detection of the annunciation stimulus implies that attention is directed away from its current focus, and sensory apparatus may also be redirected. Additionally, less attention is available for previously ongoing processes. If the operator's attention remains directed to the annunciation, it is interpreted; that is, translated into the associated interrupting task performance requirements, and the operator is said to be, additionally, *distracted* from the ongoing task. Interpretation requires attention resources to retrieve, or activate, long-term memory representations of the interrupting task; requires representation in working memory; and requires attention resources to maintain this working-memory representation. Capacity limitations and differentiation of these resources may result in deleterious effects. These effects are here defined as effects induced by the interruption. If the operator integrates the interruption, progress on the ongoing procedure is *disturbed*. Integration imposes additional attention and working memory requirements associated with preemption and resumption of the interrupted position. The execution of interruption response plans, and the process of scheduling when the interruption will be performed require attention and working-memory. Disturbance effects refer to those localized to preemption of the ongoing procedure, performance or scheduling of the interrupting task, and resumption of the ongoing procedure. Interruptions may also propagate to *disrupt* future performance on the ongoing procedure. Disruptions are deleterious effects due to previous diversion, distraction, and disturbance effects.

While the terminology for interruption effects on the ongoing procedure; *i.e.*, diversion, distraction, disturbance, disruption, have negative connotations in general parlance, the model does not imply a value judgment for attending to an interruption rather than to the ongoing task; that assessment is incumbent upon the operator following annunciation interpretation. The relative costs of these effects must be balanced with benefits of processing and performing the interrupting task.

Measuring Effects of Interruptions

Diversion indicates only that the operator has oriented perceptual mechanisms to the annunciation stimulus, has determined that that facet of the sensory environment is deserving of further processing. Diversion, therefore, may be indicated by such measures as EEG excitation and eye movement latencies. Distraction, a momentary deflection of attention from ongoing activities to interpret the interruption annunciation, may be indicated by measuring reaction time to comprehending the annunciation stimulus' task requirements. Disturbance effects are due to the efforts imposed by immediately performing or determining future performance of the interrupting task. Measures that indicate the degree to which an interruption disturbs the ongoing procedure at the point of interruption include time latencies to begin the interrupting task and to resume the procedure following interrupting task performance, errors in performing the interrupting task, and unnecessary compensatory actions prior to resuming the ongoing task (*cf.* Kirmeyer 1988). Finally, interruptions potentially disrupt the ongoing procedure as a result of the propagating effects of diversion, distraction, and disturbance. Measures that address these effects on the ongoing procedure as a whole may include the time to perform the procedural and interrupting task requirements, errors in the interrupted procedure, and unnecessary compensatory behaviors (*cf.* Kirmeyer 1988) during the interrupted procedural interval.

A Framework for Relevant Research Perspectives

The proposed stage model of interruption management is useful for defining the effects of interruptions. This model also identifies basic research perspectives relevant to the study of interruptions. These research perspectives can suggest factors that may influence interruption management. In the following sections I describe research perspectives and their association with the model stages. Although a complete review of these perspectives is beyond the scope of this project, this section identifies these perspectives and describes some of their theoretical and empirical implications for interruption management, specifically focusing on factors that are experimentally investigated in the following simulation study.

Detection and Sensory Information Processing

Initially, unexpected interruptions must be detected, or attended to, to begin the interruption management process. Attention resources can be directed to environmental defined elements either involuntarily or intentionally (*e.g.* Muller and Rabbitt 1989; Remington *et al.* 1992; Folk *et al.* 1992). Attention may be captured by external stimuli, or intentionally directed to elements of the perceptual array in response to statistical regularities in the environment (*e.g.*, Moray ,1986; Bohnen and Leermakers 1991). These two mechanisms for obtaining environmental information are also know as, exogenous and endogenous attention control (Posner 1980). These two mechanisms are also known as bottom-up, or stimulus-directed; and top-down, or goal-directed, respectively (Yantis 1993). Alternatively, these mechanisms may be considered as failures in focused attention, and selective attention switching, respectively (*cf.* Wickens 1984).

Pre-attentive processes define locations and/or objects in the perceptual array (*e.g.*, Treisman and Gormican 1988) to which they exogenously direct attention for more

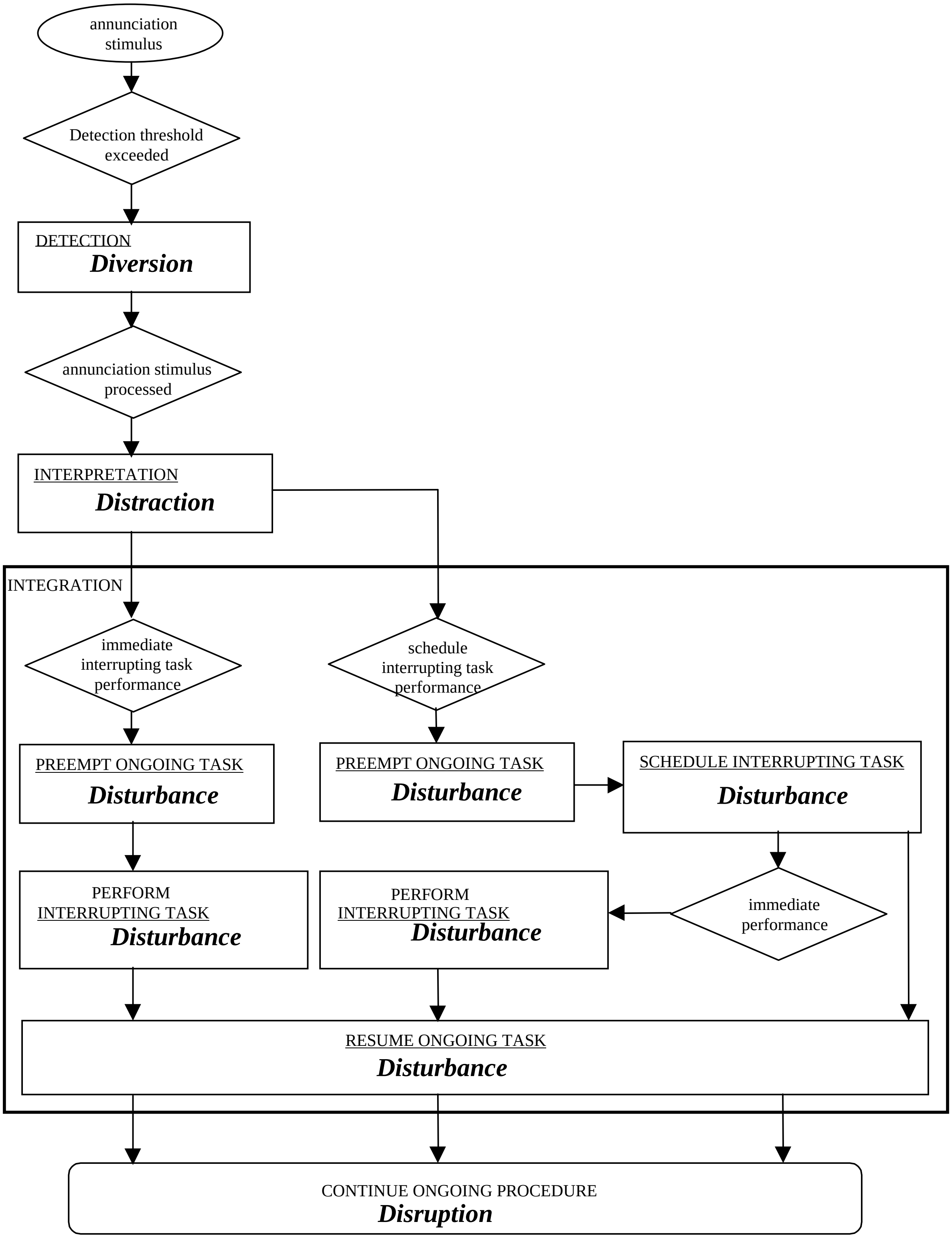


Figure 3.2 Effects of Interruptions on an Ongoing Procedure.

complex, interpretive processing (*e.g.*, Folk *et al.* 1992). Factors that induce exogenous attention control determine the ability of the annunciation stimulus to divert attention from ongoing processes. Signal detection theory (SDT) can model this process as the ability of these pre-attentive processes to distinguish an important stimulus, here the annunciation stimulus, in a surrounding stimulus environment of noise. Signal detection theory emphasizes that the probability that an operator will detect an annunciation stimulus is determined not only by physical characteristics of the annunciation stimulus, but its salience relative to the surrounding sensory context, and characteristics of the operator. Stimulus characteristics found to exogenously, or involuntarily, redirect focused attention include; use of the auditory rather than visual modality (*e.g.,* Nissen 1974; Posner *et al.* 1976; Stanton 1992); abrupt changes in stimulus attributes, specifically changes in luminance (*e.g*. Muller and Rabbit 1989; Posner 1980); proximity to previous attentional focus (*e.g.* Posner, Snyder, and Davidson 1980). Characteristics of the operator related to the ability of a stimulus to exogenously capture attention include; individual-specific thresholds for stimulus attributes, *e.g.*, intenstity, duration, wavelength (Posner 1980); operator functional visual field (*e.g.*, Balota and Rayner 1991); operator perceptual style, *i.e.*, field-dependence (*e.g.*, Braune and Wickens 1986); the operator's active inhibition of external stimuli (*e.g.,* Fox 1994); and resource-priming (Wickens 1984). Arousal theory suggests that environmental stressors increase arousal, effectively reducing attention resources for attending to external stimuli (*e.g.*, Hamilton and Warburton 1979; Sheridan 1981). Thus, physical properties of an annunciation stimulus and characteristics of the operator influences the probability that the annunciation stimulus succeeds in exogenously capturing an operators attention and permits further interruption processing.

Working Memory Manipulations

In terms of the previously-described interrupting task paradigm, an annunciation stimulus occurs while the operator performs an ongoing procedure. Prior to an interruption, active memory contains those knowledge structures relevant to this procedure, and to a greater extent those relevant to the current task. Interpretation of an annunciation stimulus requires that knowledge structures associated with it are active, or resident, in working memory. The process of retrieving, or activating, the interruption's knowledge structures requires attention. Four characteristics relate to the demands imposed by an interrupting task's working memory representation. First, working memory is capacity-limited. Second, working memory representations are not self-sustaining. Third, working memory representations are code-specific. Fourth, the attention required to access knowledge structures is inversely related to the degree to which they are already resident in working memory, or activated in memory. These characteristics and their implications for interruption management are described below.

Capacity Limitations

Capacity limitations of working memory can be discussed by considering working memory as a storage bin with a limited number of slots. Miller (1956) originally defined the limitations of working memory span as 7 (+/- 2) chunks of information, given full attention resources (Wickens 1984). Chunks of information are defined by associations in long-term memory and refer to the level of abstraction at which the information is meaningful (*cf.* Chase and Ericsson 1981). In contrast to the storage bin analogy, cognitive network models of memory represent working memory capacity limitations as a limitation of activating resources (Just and Carpenter 1992). Representation of an interrupting task in working memory competes for "space" or "activation" with representations related to the ongoing procedure. Interruptions associated with different knowledge than that required by the interrupted task may displace their associations, increasing the likelihood of forgetting a critical element of ongoing procedure performance (Adams, Tenney, and Pew 1995; Detweiler, Hess, and Phelps 1994).

Volatility

Numerous studies indicate that working memory contents decay in the absence of attention rehearsal. Retention for items represented in working memory is essentially non-existent after 20 seconds without rehearsal (Brown 1959; Peterson and Peterson 1959), and little information is available beyond 10-15 seconds (*e.g.*, Moray 1980). Retention interval length is inversely related to the number of items in working memory (Melton 1963). Cognitive network models of memory describe the volatility of working memory as a loss of activation (Detweiler, Hess, and Phelps 1994). Thus, the number of pre-existing items in working memory has implications for the retention interval of interruption-related information if no rehearsal is possible. Similarly, the addition of the interruption reduces the retention intervals of pre-existing items related to the ongoing procedure. Finally, if interruption performance or scheduling requires longer than 20 seconds and does not permit attention rehearsal of working memory contents, representations associated with the interruption position in the ongoing procedure may decay, making procedure resumption more difficult (Detweiler, Hess, and Phelps 1994).

Interference

The implications of working memory capacity-limitations and volatility apply irrespective of the form or semantic content of the information represented. Retention intervals decrease if newly added representations, *interpolated* material, are similar to the pre-existing working memory representations, or *pre-load*. Interference effects result when interpolated and pre-load materials are similar in terms of memory codes (*i.e.*, phonetic, visual, semantic) (Wickens 1984). *Retroactive interference* results if a similar representation intervenes between encoding the pre-load representations and retrieving them for use (*e.g.*, Underwood 1957). The effect of *proactive interference* accumulates when similar items are presented serially without adequate separation, and interpolated materials interfere with encoding of pre-load materials (Wickens 1984). Results of interference studies form the basis of limited-capacity, differentiated resource models of attention and memory, *e.g.*, Multiple Resource Theory (Wickens 1984) and suggest that tasks are better timeshared when they require different memory codes. For example, pairs of targets presented in two different sensory modalities are better detected than targets presented either both visually or both aurally (Treisman and Davies 1973; Rollins and Hendricks 1980). Network architectures of cognition characterize interference as the result of a redistribution of activation strengths and therefore degraded representations (Detweiler, Hess, and Phelps 1994). If an interruption activates knowledge representations incompatible with those previously in active memory, the representations may combine in such a way to result in increased processing time, and / or confusion (Adams, Tenney, and Pew 1995).

To the extent that an interruption engages the same resource codes utilized by the interrupted ongoing task, the interruption will degrade performance of that task (Liu and Wickens 1988). Interruptions that require coding resources similar to those already entertained in working memory will interfere more, and cause shorter retention times than interruptions requiring different coding resources. As an example, a visually-presented interruption should be less interfering to an ongoing auditory task than an auditory interruption.

Memory Retrieval

Interference effects derive from code similarity among items represented in working memory and suggest that similar representations degrade retention of working memory items. Content similarity, however, facilitates memory access. Adams, Tenney, and Pew (1995) describe this effect and implications for interruption management in terms of Sanford and Garrod's (1981) theory of text comprehension.

Sanford and Garrod (1981) describe two types of memory, *active* and *latent*. Active memory is that portion of the operator's long-term memory that is primed for use in the current situation. Latent memory is the remainder of the operator's long-term memory. Active memory contains two bins; memory that is in explicit focus (EF), and memory that is in implicit focus (IF). The contents of explicit focus can be considered working memory. Explicit focus has the following properties. It operates as a fixed-capacity queue (*cf*. Miller 1956) containing pointers to knowledge structures in long-term memory. Attention is required to maintain EF. Further, maintenance of any EF pointer is a function of its relevancy for the task at hand, and the recency of activation (Adams, Tenney, and Pew 1991). Implicit focus encompasses the full representation of the situation that is partially represented in EF. Access to information in IF is slower and must be more directly-addressed than to that in EF (Adams, Tenney, and Pew 1991). Sanford and Garrod (1981) describe latent memory as composed of two bins: long-term episodic memory and long-term semantic memory. With respect to multiple-task management, long-term episodic memory contains a complete record of the knowledge structures that have been constructed or accessed in the course of the current mission (Adams, Tenney, and Pew 1991). Long-term semantic memory contains the lifetime accumulation of knowledge in general. Knowledge structures residing in latent memory can be accessed only given considerable effort or strong external cueing, however episodic memory is more easily activated than more cognitively-remote semantic memory (Adams, Tenney, and Pew 1991).

Based on this model of memory, Adams, Tenney, and Pew (1991, 1995) predict characteristics that influence interruption management, related to the interpretation of annunciation stimuli: (1) Interrupting events are most easily assimilated that directly map to the knowledge resident in explicit focus for the ongoing task. (2) Events related to the ongoing task, but not to that aspect of it in process, are also handled relatively easily because they refer to knowledge that is active in implicit focus. (3) If an interruption is not related to those knowledge structures primed by the ongoing task, and requires additional long-term memory addressing, the probability and effort associated with proper processing depends on factors such as, the saliency of significance and the time available for interpreting significance. In summary, Adams, Tenney, and Pew's (1991, 1995) points generally propose that interruption management is facilitated to the degree that the interrupting and interrupted (ongoing) task are conceptually similar, that is, refer to and rely on the same knowledge structures. This supposition is consistent with spreading activation theory for network models of memory (Anderson 1983) and with empirical evidence (*e.g.*, Meyer and Schvaneveldt 1971).

Intentions and Working Memory

Two theories from motivational psychology suggest conditions under which an interruption is most easily integrated with an ongoing procedure and mechanisms underlying task preemption and resumption; Lewin's field theory (1926, 1951) and Miller, Galanter, and Pribram's (1960) cognitive theory of intentions. These theories are based on a vast collection of empirical work initially established by Zeigarnick (1927) and Ovsiankina (1928) (see section 2.1.3). This section briefly presents and contrasts the above theories. Adams, Tenney, and Pew (1991, 1995) assimilate these theories into their cognitive framework of multiple task management. Their framework, as does the proposed model of interruption management, assumes goal-directed behavior that can be represented in as a hierarchical plan, although it also assumes that operators are reactive to their environment. The implications Adams *et al.* (1991, 1995) derive for interrupted task management are discussed.

Lewin's (1926, 1951) field theory of task tension presumes no cognitive mechanism. Rather, it proposes that organisms tend toward a state of equilibrium, of homeostasis, at the lowest level of tension. Once a task is begun, the requirements to perform the task are considered *quasi-needs*, and the set of these are considered a *tension system*. Lewin (1926, 1951) proposes that as long as a task remains unfinished, it represents a system under tension, tension that can only be dispersed upon task completion. If activities required for task completion are not permitted, the quasi-needs are not fulfilled, and the system remains under tension. It is this tension, then, that compels recollection of uncompleted tasks and the intention to resume interrupted tasks. Miller, Galanter, and Pribram (1960) impose a human information processing model on Lewin's theory to clarify the concept of an *intention*. These authors first define the set of ongoing tasks as a *plan*, "any hierarchical process in the organism that can control the order in which a sequence of operations is to be performed (p. 16) (italics omitted)." To execute a plan, it must be brought into active memory (Miller, Galanter, and Pribram 1960). If interrupted during execution, the representation of an index to remaining activities remains resident in active memory. This activated pointer in working memory then motivates improved recollection and the desire to resume an interrupted task.

Miller, Galanter, and Pribram's (1960) cognitive theory provides an information processing interpretation of Lewin's (1926, 1951) motivational theory. Although these two interpretations are consistent in their prediction of behavior in most cases, conflicts arise when Lewinian theory would assume the establishment of a tension system, when an external representation of plan progress obviates the need for an internal representation to index progress. Both theories predict that interruptions in simple, repetitive, continuous tasks (*e.g.*, stringing beads) would not compel recollection, or resumption following an interruption (Adams, Tenney, and Pew 1991). Lewinian theory suggests that this is because continuous tasks do not establish a tension system and that because there is no distinguishable structure to, or endpoint of, a continuous task; interruption is simply a halt to ongoing performance. According to Miller, Galanter, and Pribram (1960), continuous tasks do not require a plan, and therefore are not hierarchically represented in active memory. Thus, there is no residual intention to complete or recall the interruption. When such a task has an endpoint, however (*e.g.*, to string a certain number of beads), Miller, Galanter, and Pribram (1960) suggest that a representation of the uncompleted task compels recollection and resumption. Lewin's (1926, 1951) theory also predicts this outcome, because since the task is now interruptable, a task-tension system develops. However, empirical results do not always indicate this effect (Zeigarnick 1927). Zeigarnick (1927) explains these results, based on Lewinian theory, by suggesting that because the interruption point in this type of task is arbitrary, a tension system does not develop and subjects are unlikely to recall the interruption point or resume the task. When a task has an externally-obvious endpoint, rather than one internally maintained (*e.g.*, to string all the beads provided) these theories predict different results. As in the previous case, Lewinian theory assumes construction of a tension system that compels recollection and resumption of the interrupted task. In contrast, Miller, Galanter, and Pribram (1960) suggest that, because requirements for task completion are represented externally, no internal representation exists and thus recall and resumption of the interrupting task are not likely. Ovsiankina's (1928) results support the Lewinian position and Bechtel's (1965) results support the cognitive theory's position (Van Bergen 1968).

Adams, Tenney, and Pew (1991, 1995) extend these theories of intentions and motivation in their cognitive framework of multiple task management. First, consistent with Miller, Galanter, and Pribram (1960), they assume that an active memory representation exists to guide performance. However, rather than assuming that only index information is represented; as in Miller, Galanter, and Pribram (1960); Adams, Tenney, and Pew (1991, 1995) assume that the entire mission of a multiple task management situation is in activated memory. Further, they assume that the particular activity engaging the operator resides in explicit focus, and goal-related knowledge, less closely-related to the immediate task resides in implicit focus (Adams, Tenney, and Pew 1991). This is consistent with Lewinian theory which assumes that mission initiation raises activation of all information related to its performance (Adams, Tenney, and Pew 1991).

Both Adams, Tenney, and Pew (1991) and Miller, Galanter, and Pribram (1960) predict that interruptions will be more tolerable on completion of a currently active task, because the contents of explicit focus (or short-term working memory) are being closed and replaced, implying that the level of the goal hierarchy at which the interruption takes place has implications for effects of an interruption (Adams, Tenney, and Pew 1991). Miller, Galanter, and Pribram (1960) predict only whether or not there will be an effect, depending on the existence or lack thereof of an internal index representation. For example, if an operator is asked to perform five discrete tasks, both the cognitive theory of intentions (Miller, Galanter, and Pribram 1960) and the cognitive framework of multiple task management (Adams, Tenney, and Pew 1995) predict that, if interrupted within one of these tasks, the operator would remember the interrupted task and attempt to resume it (Adams, Tenney, and Pew 1991). If the interrupt occurs between the second and third task, Miller, Galanter, and Pribram (1960) predict that, because the second task was completed and the representation for the third task not yet required, there is no residual working memory representation, the operator does not recall the interruption point, and therefore does not resume the interrupted task set (Adams, Tenney, and Pew 1991). In contrast; because Adams, Tenney, and Pew (1991) assume that the entire goal hierarchy of the mission is activated, they suggest that the working memory, or active memory, representation for the complete mission, *e.g.*, “perform a set of five tasks” remains (Adams, Tenney, and Pew 1991). This representation then compels the operator to resume performing the set of tasks. Adams, Tenney, and Pew (1991, 1995) propose that interruptions should be less tolerable between sub-goals than between goals; that is, the higher up the goal hierarchy, the more tolerable the interruptions. Lower-level goal interruptions will be more resistant to interruption, and, at some atomic level, goals will be impervious to interruption.

In summary, the extension (Adams, Tenney, and Pew 1991, 1995) of intention and motivation theories (Lewin 1926, 1951; Miller, Galanter, and Pribram 1960) suggests that the representational structure of the mission, or ongoing procedure, has implications for interruption handling. Specifically, they predict that interruptions are less disturbing when they occur at cognitive breakpoints in an ongoing task, *i.e.*, upon completion of a sub-goal, and further, that interruptions are less disturbing when they intervene between higher-level goals than between activities comprising lower-level sub-goals.

Scheduling Theory

If an interrupting task does not engender performance immediately following interpretation and access to associated performance requirements, the task may be explicitly scheduled into the future of the ongoing procedure’s performance. Scheduling theory suggests factors relevant for optimal task scheduling in job shops and provides a normative model of human scheduling. Empirical research comparing human scheduling behavior to optimal scheduling rules describes deviations from this normative model. This empirical research suggests that operator scheduling behavior is subject to human information processing biases and limitations and that operators strategically manage tasks to modulate their workload levels. This section briefly reviews scheduling theory, human scheduling behavior, and strategic workload management as they apply to the intentional integration of interrupting task performance requirements into an ongoing procedure.

Scheduling

The scheduling problem, organizing activities within constraints of resource availability to meet goal criteria is a special case of the more general planning problem (Georgeff 1987). Scheduling theory is an algorithmic approach to this problem (French 1982) and is traditionally applied to determining the processing order and machine assignment of jobs in a manufacturing environment (Sadowski and Medeiros 1982). Scheduling theory specifies problems in terms of; the available processors (machines), processing characteristics of the jobs, processor constraints, job constraints, and the objective of the scheduling problem (French 1982). Job-related constraints include, job processing time (which may differ by processor), availability for processing, due date, and the priority of the job. Characteristics of the processing environment, *e.g.*, sequencing requirements, provide additional constraints. A wide variety of scheduling rules exist for assigning jobs to processors to optimize specific objectives, for example to minimize average due dates or to minimize processor idle times[6]. Traditional algorithmic scheduling theory uses both simple rules and compound rules to accomplish these goals. In addition, associating scheduling decisions with patterns of job characteristics and the job-shop environment, provides a case-based or heuristic method of scheduling. Heuristic-based scheduling provides a more context-sensitive and therefore more sophisticated approach to task ordering (Sanderson 1989). According to traditional scheduling theory, a task ordering is defined based on the objective and on the initial task set and machine characteristics. Using the rigorous method, the introduction of an additional task, *i.e.*, an interrupting task, requires estimating the new task in terms of scheduling rule parameters, and recalculating the schedule. Thus, integrating an unexpected task into a schedule requires reconsideration, and this reconsideration is both temporally- and computationally- expensive. Alternatively, if a heuristic set includes the occurrence of a specific additional task; that is, to the degree that this addition is expected, this reconsideration is pre-programmed and requires less time and computational resources.

Given complete specification of all relevant job, processor, and environmental parameters, a well-defined and measurable objective function, unlimited time and computational ability, and a stable environment, scheduling algorithms produce optimal task ordering. Although, algorithmic scheduling theory provides insight into relevant job and processor characteristics and useful performance goals, direct application of this algorithmic approach to human multiple task management in operational environments is inappropriate to the degree that these conditions are not met. The presence of an interruption in multitasking environments increases the variability of the environment. In addition, its occurrence necessitates potentially computationally, and temporally-expensive reassessment and rescheduling. Traditional scheduling theory suggests some characteristics of tasks that may be important in integrating an interruption into the remaining ongoing procedure.

Human Scheduling Behavior

Scheduling theory provides a foil for assessing human scheduling performance in a job shop environment (Sanderson 1989). Comparisons of human scheduling behavior to simple scheduling rules, complex rules, and heuristic rule sets indicate several general conclusions. Human scheduling behavior exceeds automated scheduling to the degree that the

---

[6] See Panwalkar and Iskander (1977) for a review of scheduling rules.

environment is variable (Haider *et al.* 1981) and planning horizons are fairly short (Ben-Arieh and Moodie 1987). Human-generated schedules only outperformed those generated by compound scheduling rules for some rules (*cf.*, Tabe and Salvendy 1988; Tabe, Yamamuro, and Salvendy 1988; Ben-Arieh and Moodie 1987), suggesting some inherent biases in human scheduling (Sanderson 1989). These comparisons for actually scheduling jobs to machines indicate that reactivity to environmental changes, such as interruptions, are very important. Further, these studies indicate that, while usually more reactive to environmental changes, human scheduling performance is sub-optimal.

Operator performance models instantiate theories of human multiple task management of abstract tasks. These models incorporate both characteristics of the tasks presumed to affect task management performance, and model assumptions regarding the limitations of human information processing. Some of these factors include: (1) task availability (Tulga and Sheridan 1980); (2) preview knowledge of task availability (Tulga and Sheridan 1980); (3) task processing time (Tulga and Sheridan 1980; Shankar 1989; Plocher *et al.* 1991; Pattipati *et al.* 1983); (4) velocity of approaching deadline (Pattipati *et al.* 1983); (5) ability to partially process tasks (Pattipati *et al.* 1983); (6) slack time available in tasks (Shankar 1989); (7) sequential and temporal task constraints (Plocher *et al.* 1991; Shankar 1989), (8) rewards for task processing (Tulga and Sheridan 1980; Pattipati *et al.* 1983); (9) costs for not processing tasks (Pattipati *et al.* 1983); (10) operator information processing capacities and limitations (Pattipati, *et al.* 1983; Shankar 1989; Plocher *et al.* 1991); and (11) explicit operator workload modulation goal (Shankar 1989; Plocher *et al.* 1991). Comparisons of human planning behavior to computational models of planning also indicate that human behavior is characteristically opportunistic (*cf.* Hayes-Roth and Hayes-Roth 1979), although it may be represented by hierarchical plans in structured environments (Agre and Chapman 1990).

Comparisons of human performance and these operator models indicate several characteristics of human task management. Operators satisfy performance requirements but do not optimize performance (*e.g.*, Tulga and Sheridan 1980; Pattipati *et al.* 1983; Govindaraj *et al.* 1981; Moray *et al.* 1991). Changes in task management strategies coincide with increases in workload (Tulga and Sheridan 1980; Schumacher and Geiser 1983). Operators strategically use preview information only at intermediary levels of workload (Tulga and Sheridan 1980), potentially because strategies are unnecessary at lower levels and too computationally-expensive to use in higher levels of workload (Tulga and Sheridan 1980). Humans switch tasks less frequently than is optimal, ostensibly due to human information processing limitations such as neuromuscular lags, decision making time loss, and cognitive inertia (Pattipati *et al.* 1983). Operators are not precise in distinguishing among tasks on attributes relevant to defining task execution order (Tulga and Sheridan 1980; Pattipati *et al.* 1983; Govindaraj 1981). Finally, strategic workload modulation appears to be a significant motivation in human task management behavior (*e.g.*, Wickens, Larish, and Contorer 1989).

This final point refers to a field of study in itself, *strategic workload management*. (Moray and Hart 1990). In terms of scheduling theory, one might say that an aspect of the objective function for optimization includes a term for maintaining workload at acceptable levels.

Several frameworks have been proposed for studying strategic workload management. For example, Moray (1990) proposes scheduling theory as a normative model for human strategic task management, where slack time is interpreted as an inverse measure of workload. Hancock (1991) introduces a formulation of strategic behavior as a constraint-satisfaction problem. Hart and Wickens (1990) conceptualize workload modulation as a closed-loop model. The most complete set of empirical data on strategic workload management has been performed by Wickens and his colleagues in the aviation domain. Under conditions of higher workload, the priorities pilots assign to tasks modifies the probability that they perform a given task (Raby, Wickens, and Marsh 1990). Increased workload conditions do not appear to induce strategic performance to optimize task duration, or the time at which tasks are performed (Raby, Wickens, and Marsh 1990). However, in higher workload conditions, pilots do allocate time according to priority in high workload conditions (Raby and Wickens 1990). As workload increases, pilots perform tasks according to their priority, suffering degraded performance to low-priority tasks, and become more efficient in performing tasks. Pilot performances improve when provided with a projections of difficulty demands over a scenario (Segal and Wickens 1990). Wickens and his colleagues' research suggests that the intentional integration of an interruption into an ongoing procedure is particularly influenced by the level of workload experienced by the subject, projected workload demands, and relative priority of the interruption.

In summary, scheduling theory provides a normative model for describing how an interrupting task is integrated with future, known, performance requirements. However, the influence of human information processing biases and limitations on scheduling performance are evident when comparing human performance to that of scheduling rules and optimizing operator models. In addition, to these inherent limitations, interrupting task integration is likely to be subject to strategic goals, particularly the goal to modulate workload. These research perspectives provide a basis for understanding how interrupting task requirements might be strategically integrated with ongoing task requirements.

Constraints on Attention

Many diverse theories of attention include, as a premise, that attention is a resource for information processing that is limited in quantity, required for controlled processes and, with some effort, may be divided over processes in a zero-sum manner (Allport 1992). This limited resource serves many stages in the interruption management model except the first, whose purpose is to exogenously capture this resource. Therefore, factors limiting attention availability affect resources available for all other stages that require this resource. Previous sections refer to the role of attention in descriptions of other information processing mechanisms. This section describes, generally, the implications of two factors that limit the general availability of attention for other facets of interruption management.

Automaticity and Attention

Typically, the additional attention demands associated with managing an interruption detract from ongoing procedure performance. However, processes that have been practiced to the point of automaticity can be performed without attention resources (Shiffrin and Schneider 1977; Schneider and Shiffrin 1977). In these cases, integration of interruptions may not interfere with ongoing procedure performance. Alternatively, controlled processes are capacity-limited and therefore generally serial and subject to interference from other concurrent tasks. So, to the degree that stages in interruption management or execution of the ongoing procedure are automatic, interruption management should not interfere with ongoing procedure performance. Automated mechanisms are not, by definition, interruptable (*e.g.*, Muller and Rabbitt 1989). Therefore, to the degree that the ongoing procedure is automatic, it is resistant to interruption. The task paradigm of the proposed model assumes that ongoing and interrupting tasks require controlled processes.

Environmental Stress and Attention

Attention may be intentionally divided among timeshared tasks requiring controlled processing and task-irrelevant activities (Eysenck 1982). For example, anxiety-level, as a response to internally or externally-imposed stressors, may be considered a secondary task to be time-shared with task-relevant requirements. The additional demands imposed by task-irrelevant concerns decrease performance on task-relevant processes. Stress restricts the breadth of focused attention (Easterbrook 1959; Kahneman 1973), and decreases working memory capacity (Eysenck 1982). Accordingly, interruptions are assumed to be less permeating to ongoing procedure performance under these conditions. However, if interpreted, interruption integration will be more difficult under conditions of increased stress.

4. Experimental Hypotheses

The proposed interruption management model defines interruption management stages, describes effects on ongoing performance and interruption management strategies, and suggests basic research related to the study of interruption management. Hypotheses based on the proposed model and related literature were addressed in a flight simulation environment specifically designed to enable precise experimental manipulation of interruption positions. To authentically demonstrate the effects of interruptions on the flightdeck, this experiment used current airline pilots as subjects and realistic ATC transmissions as interruptions to flightdeck procedures. The experimental component of this research seeks to demonstrate experimentally the deleterious effects ascribed to interruptions on flightdecks in actual operations and consider the significance of several task factors to interruption management performance on the flightdeck. These factors include: (1) the modality of the interrupting and interrupted tasks, (2) the goal-level of the interrupted task in the ongoing procedure, (3) the coupling strength of sequential procedural tasks that are severed by an interruption, (4) the semantic similarity of the interrupting and interrupted tasks, and (5) the environmental stress associated with the interrupted ongoing procedure. Measures of distraction, disturbance, and disruption characterize the influence of these task factors on flightdeck interruption management performance. These factors and the expected results are described below. In addition, I plan to note where individual differences appear in these analyses.

Interruption Hypothesis

<u>*Hypothesis 1:*</u> *Interrupted procedures will contain more errors than uninterrupted procedures, involve a higher rate of flightpath management, and, aside from the additional time required for performing the interrupting task, take longer to perform.*

This hypothesis is based on results of specific laboratory investigations demonstrating the deleterious effects of interruptions on interrupted-task performance (Detweiler, Hess, and Phelps 1994; Gillie and Broadbent 1989; Field 1987; Kreifeldt and McCarthey 1981), observational studies indicating performance decrements associated with interruptions (*e.g.*, Kirmeyer 1988; Paquiot, Eyrolle and Cellier 1986), and consequences of interruptions annotated in incident and accident reports (*e.g.*, Griffon-Fouco and Ghertman 1984; Bainbridge 1984; Turner and Huntley 1991; Monan 1979; NTSB 1988, 1973). Although this prior research clearly demonstrates the negative effects of interruptions on human performance, interruptions have had both extending (*e.g.*, Paquiot, Eyrolle, and Cellier 1986; Kreifeldt and McCarthey 1981; Field 1987) and contracting (*e.g.*, Cellier and Eyrolle 1992) effects on overall performance time.

Modality Hypotheses

Three hypotheses are proposed based on the modality of the interruption, the interrupted task, and the interaction of interruption and interrupted task modalities.

Interruption Modality Hypothesis

Hypothesis 2: *Interruptions presented aurally should be more distracting than interruptions presented visually.*

Auditory information is more attention-directing than visual information (*e.g.*, Neisser 1974; Posner *et al. 19*76; Stanton 1992). Based on this, other authors suggest that an auditory task is more likely to preempt an ongoing task than a visual task (Segal and Wickens 1991). Although the visually-presented interruptions in this experiment begin with a momentary auditory annunciation, to equalize diversion effects, it does not persist and therefore does not continue to be attention-demanding. Contrary to this implication from basic research, Datalink research finds that pilots typically respond more rapidly to datalink, or visual, messages than to aural radio calls (Kerns 1990). Datalink, or visually-presented, ATC messages also precipitate longer delays before resuming interrupted tasks (Williams 1995).

Interrupted Task Modality Hypothesis

Hypothesis 3: *Interruptions to visual tasks should be more distracting, and less disturbing and disruptive than interruptions to auditory tasks.*

Interruptions to tasks that retain interruption position information externally experience less performance degradation than tasks that do not (Kreifeldt and McCarthey 1981; Field 1987). In this experiment, interrupted visual procedural tasks provide an externally-available reminder to resume the interrupted task and therefore do not require subjects to retain an internal representation of the interruption position. This reduced memory load and external aid should facilitate subjects performance compared to that with interrupted auditory procedural tasks.

Modality-sharing Hypothesis

Hypothesis 4: *Cross-modality conditions should be more distracting, and less disturbing and disruptive than same-modality conditions.*

Differentiated-resource models of attention suggest (*e.g.*, Wickens 1984) and supporting empirical results from timesharing research (*e.g.*, Triesman and Davies 1973; Rollins and Hendricks 1980) indicate that tasks are more easily performed simultaneously when they require different processing resources.

Goal-Level Hypothesis

Hypothesis 5: *Interruptions should be less distracting, more disturbing, and more disruptive to the degree that they are embedded in a procedure.*

Specifically, interruptions presented external to the procedure, either before or after, should be more distracting, less disturbing, and less disruptive than interruptions either between or within procedures. Similarly, interruptions between procedural tasks should be more distracting, less disturbing, and less disruptive than interruptions within procedural tasks. Adams, Tenney, and Pew (1991, 1995) extend theories of intention formation (*cf.* Lewin 1951; Miller, Galanter, and Pribram 1960) and their interaction with working memory to suggest that interruptions within low-level goals of the ongoing task set are more destructive than interruptions between high-level goals. Interruption research finds that increased memory load at the interruption point, defined by lower-level interruption in a hierarchical ongoing task, significantly degrades performance (Detweiler, Hess, and Phelps 1994). Psycholinguistic research describing perceived interruption points in speech also supports this hypothesis (Cairns and Cairns 1976). However, an attempt to demonstrate this goal-level effect in a laboratory setting was not successful (Lorch 1987).

Coupling-Strength Hypothesis

Hypothesis 6: *Interruptions should be less distracting, more disturbing, and more disruptive if presented between tasks perceived as strongly-coupled, or associated, than if presented between tasks that are perceived as less strongly-coupled.*

The goal-level hypothesis attempts to predict effects on interruption management based on an objective analysis of the ongoing procedure's structure. However, research suggests that operators come to make associations among procedural tasks into meaningful sub-units (*e.g.*, Elio 1986). The coupling-strength hypothesis considers subjects' constructed associations among procedural tasks.

Similarity Hypothesis

Hypothesis 7: *Interruptions semantically similar to the interrupted task should be more distracting, and less disturbing and disruptive than dissimilar interruptions.*

Theories of associated memory suggest that responding to and integrating information associated with a new stimulus is facilitated by the degree to which requisite memory structures are already activated, or resident in working memory, or are related to those structures in active memory (*e.g.*, Anderson 1976; Sanford and Garrod 1960). Adams, Tenney, and Pew (1991, 1995) extend this concept to predict that interrupting events are most easily assimilated to the degree that they map to activated, current, memory structures, ostensibly those associated with the interrupted task. A previous laboratory investigation with alphanumeric stimuli did not confirm this hypothesis (Cellier and Eyrolle 1992).

However, tasks in a realistic operational setting may have more elaborate memory associations and may therefore be more appropriate to testing this hypothesis.

Environmental Stress Hypothesis

Hypothesis 8: *Interruptions to procedures performed in higher stress conditions should be less distracting and more disturbing and disruptive than interruptions to procedures performed in lower stress conditions.*

Attention theory and research suggests that stressful conditions diminish attentional resources available for task-related activities (*e.g.*, Eysenck 1982). Attention research indicates that subjects should be less divertable, and therefore less distractible, at higher stress levels. However, operator task scheduling research suggests that people become more opportunistic in higher stress conditions (*e.g.*, Tulga and Sheridan 1980; Schumacher and Geiser 1983), and perhaps may be more likely distracted. Pilots response times to datalink messages decrease in more stressful conditions, operationalized by decreasing altitude and distance to runway (*e.g.*, Diehl 1975; Waller and Lohr 1989).

Observations on Individual Differences

Hypothesis 9: *Individual subject performances will be significantly different in response to interruptions on the flightdeck.*

Personality (Kirmeyer 1988) and cognitive style (Jolly and Reardon 1985) characteristics have been associated with differentiated responses to interruption. Because this experiment is conducted in a realistic task setting, subject behavior is not constrained as tightly as would be the case in most laboratory experiments. For this reason, despite the commonality that all subjects are commercial airline pilots of certain experience, significant individual differences may be particularly salient. This experiment provides a realistic task context, and is therefore less restrictive on subject behavior than traditional laboratory investigations. Thus, even considering that subjects are from a restricted population, significant individual differences may be particularly salient.

5. Experimental Methods

Participants

Participants in this experiment included those required to design and pre-test the experimental scenario and those required to perform the experiment. Participants of the former category included domain expert consultants and preliminary subjects. Participants in the latter category included experimental subjects and experimental personnel. The characteristics and roles of these participants are described below.

Experiment Design and Development Participants

In preparation for this investigation, several questionnaires and card-sorting tasks were given to 46 current airline pilots. The results of these preliminary studies informed the design of the procedures and interrupting tasks, the operationalization of independent factors' levels, and the design of experimental materials. Extensive interviews during a two-year development period with two retired United Airlines pilots who are experienced in pilot training, and an experienced air traffic controller, informed scenario design and development to maximize operational validity of the scenario and efficacy of the training regime. Three NASA researchers, a NASA test pilot, and eight current airline pilots with the same qualifications as the experimental subjects, served as preliminary subjects to refine the experimental scenario and materials, training regime, and experimental protocol.

Experimental Subjects

The fourteen experimental subjects were transport airline Captains or First Officers who were currently flying a Boeing 737-300, 737-400, 747-400, 757, or 767 aircraft, had at least one year of FMS/CDU and glass-cockpit experience, and minimally 5,000 flying hours (Appendix 5.1). Experimental subjects were recruited by advertisement and each compensated $200.00 plus accommodations and *per diem* for their two days of participation.

Experimental Personnel

The expert pilot consultants also provided simulation training on flightpath management skills. The expert air traffic controller performed all real-time ATC and airline company coordination communications, and pre-recorded all aural interruption annunciations. Additional personnel operated the simulation facility hardware and software. Personnel of the Human Engineering Methods group of the Crew-Integration branch at NASA Langley placed sensors and operated apparatus to collect physiological data from subjects for a related experiment, not described here.

Apparatus

This section describes the simulation platform for the experiment and additional apparatus required to provide subjects with ATIS (Automatic Terminal Information System) information, real-time responses to flightdeck-initiated interactions from ATC and the airline company coordinating services, and for interjecting interrupting ATC requests, the interruption annunciations.

The Simulator

The simulation platform was the NASA Langley's Transport Systems Research Vehicle (TSRV) fixed-base simulator. The TSRV flight-deck is similar to a Boeing 737-300 but possesses some unique features and was modified specifically for this research as the TSRV-IIC. Software modifications included development of equipment logic specifically for the experimental scenario, key-stroke level data collection, definition of run characteristics, sensing interruption triggering conditions and introducing interruptions precisely (Appendix 5.2). Additional software was designed to extract dependent measures from raw time-stamped keystroke and event posting simulation data specific to each experimental condition (Appendix 5.3). Hardware modifications included the alteration and addition of equipment necessary for procedure performance, and installation of sensing mechanisms to enable keystroke-level data collection and interruption insertion. Specific physical characteristics of the TSRV-IIC are described below in terms of their use in this study.

5.2.1.1 Control Mechanisms

The TSRV-IIC used sidestick controller input device rather than the standard yoke and column. Pilots flew the simulator in Attitude Control Wheel Steering (ACWS), a highly-manual, reduced form of the autopilot in which the sidestick controller inputs provide rate commands to the autopilot. Once a bank angle or attitude was achieved, if the pilot released the sidestick controller, it returned to the neutral position while the aircraft maintained the established bank angle and attitude. Neither full autopilot nor autothrottles were available for use in the experimental scenario.

In normal airline operation, pilots enter target speeds, altitudes, and attitudes in a mode control panel as input to autopilot guidance. In this experiment, these target speeds, altitudes, and attitudes were preprogrammed in the simulation program. These preprogrammed parameters did not drive autopilot controls, but were reflected in primary flight display features. The display features for these target parameters were “bugs”, or markers, and text boxes that indicated target attitudes for descents and level-offs, and target speeds and altitudes for crossing all waypoints. Subjects did not interact with the mode control panel during this experiment.

Primary Flight Display

The primary flight display (PFD), located directly in front of the pilot, provided guidance information for flight parameters, and contained the following major display features (Figure 5.1): (1) turn thumbtack, 2) horizontal path deviation indicator, (3) aircraft reference symbol / flightpath angle (FPA) diamond, (4) pitch indicator, (5) FPA reference bar, (6) speed indicator (including actual, trend, and target information), (7) altitude indicator (including actual, trend, and target information), (8) distance to the next waypoint, (9) radio altimeter (not shown in Figure 5.1), and (10) video archiving information (including subject number, run number, and elapsed run time), (11) name of and distance to the next waypoint. The following sections describe the information provided by PFD display features in the experimental scenario.

Lateral Information and Guidance

Two PFD display features conveyed lateral path information. The indicator, *thumbtack*, indicated the airplane's track-bearing relative to the desired track-bearing. When the aircraft reached a calculated distance from a turn, the thumbtack moved in the direction of the turn, assuming an instantaneous 15 degree bank. A scale on the top of the PFD provided bank angle information in degrees. While the thumbtack provided guidance to remain on the flightpath, it alone did not provide enough guidance to get back onto the flightpath. That is, it did not provide true lateral deviation information. The horizontal path deviation indicator (HPDI) provided true lateral deviation information. Each demarcation on the HPDI scale represented 3750 feet of lateral deviation. If the aircraft was 3750 feet to the left of the correct track and on a parallel course, the thumbtack would be in the center of the screen, but the HPDI would be centered on its scale's second demarcation. The aircraft's heading remained constant if the HPDI showed no deviation, and the thumbtack and the aircraft reference symbol (also referred to as the FPA diamond) were coincident. Lateral deviations were also indicated on the navigational display by a separation of the aircraft symbol and the plan view representation of the flightpath.

Attitude Information and Guidance

The PFD had two attitude indicators. The first was a standard pitch indicator, this reflected the pitch angle of the aircraft. The second, the FPA diamond, displayed the lateral position and attitude of the aircraft and presented attitude information in terms of FPA. When manually controlling attitude with pitch, one must make constant adjustments to compensate for different aircraft configurations, airspeeds, altitudes, and winds. The FPA diamond display feature allows pilots to "fly the center of gravity of the aircraft"; that is, to control the aircraft's direction rather than just its heading, and obviates the need for fine lateral or vertical compensatory adjustments in response to winds or altitude changes. The experimental scenario provided FPA reference attitudes for descents and indicated level-offs with the PFD's FPA reference bar. Upon passing a waypoint, this bar dropped from the horizon line to the target FPA for that descent. At 300 feet above a level-off altitude, the bar returned to the horizon line to signal the pilot to level-off at that altitude. Although most pilots were unfamiliar with FPA attitude control, it provided an easier method for achieving descent rates than pitch-references and, once stabilized, enabled hands-off flying with zero flightpath deviation.

Speed and Altitude Information and Guidance

The PFD also provided actual and trend information for speed and altitude in the form of two tape display features. The actual speed and altitude were framed on their respective tape display features and shown in text at the bottom of these tapes. Speed and altitude deviations were readily apparent by comparing the relative distance between actual values

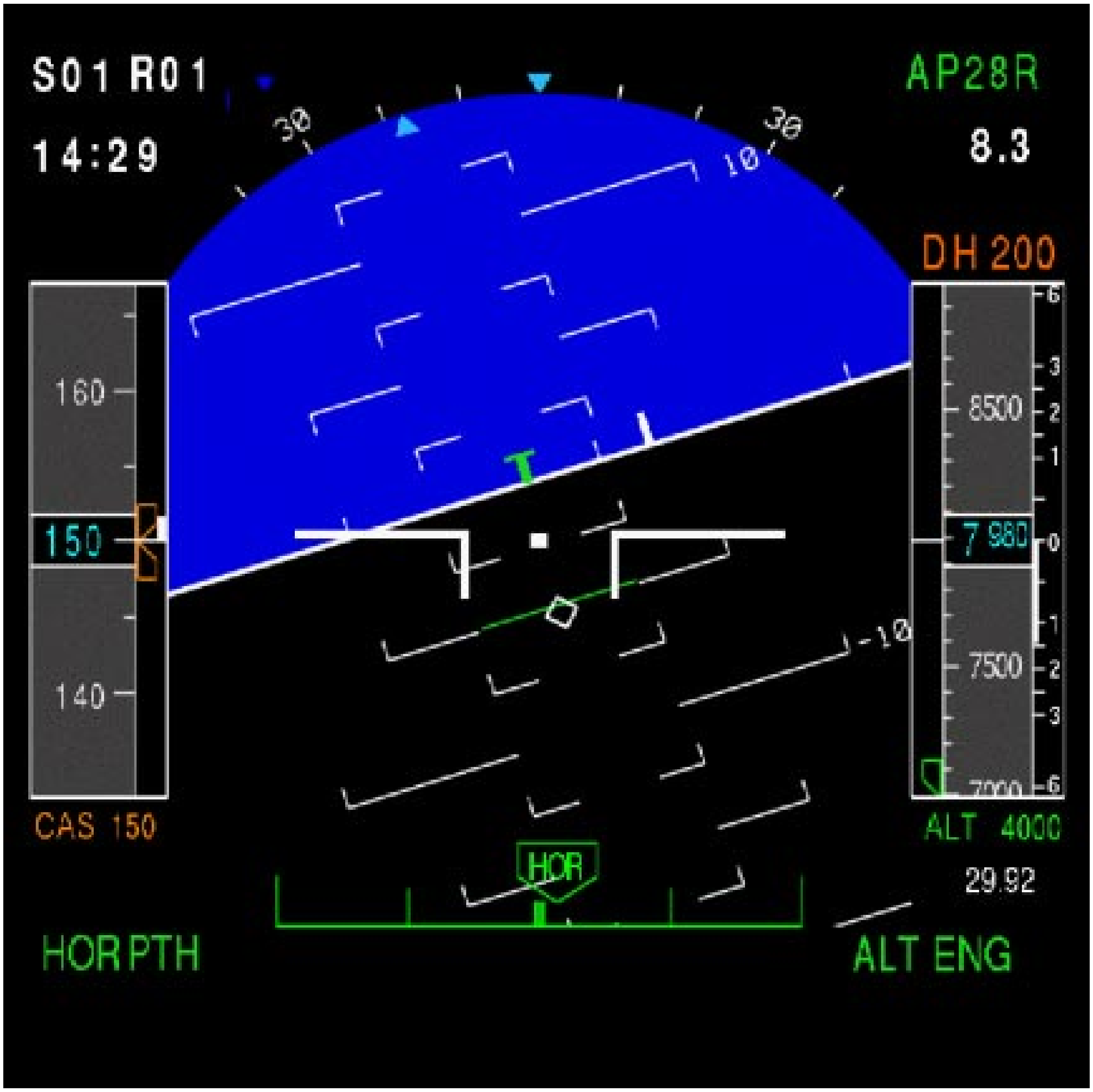


Figure 5.1. TSRV-IIC Primary Flight Display

and marked target values on display tapes. A narrow white tape to the right of the speed bug indicated the projected speed in 10 seconds. Another white tape, to the right of the altitude tape, indicated rate of altitude change, or *vertical speed*. While vertical speed indicators are standard in current aircraft, the speed trend tape is not. This experiment required manual throttle management for thrust control. As most commercial pilots typically fly with autothrottles, the speed trend information provided very useful information for manual throttle management.

Other PFD Display features

Other PFD display features included those specific to landing and those added to aid video archiving. Upon reaching 1000' above field elevation, a radio altimeter feature indicated the feet remaining to field elevation as "RA ###". After passing the approach point (2 nm from and 500' above the touchdown point), a graphical representation of a runway was presented on the PFD. The name of and distance to the next waypoint was displayed in the PFD's upper right corner.

Navigational Display

The Navigational Display (ND) (Figure 5.2), located below the PFD, provided: (1) a track-up, plan-view of the remaining flightpath, (2) waypoints on the remaining flightpath annotated with programmed crossing speed and altitude restrictions, (3) an aircraft symbol annotated with actual speed and altitude, (4) current heading, (5) the name and distance to the next waypoint. Although the ND scale was variable between runs, once a run began the scale was fixed to the 20 nm scale. At the approach point, the scale changed to 2 nm to aid landing. The ND displayed the aircraft symbol in the center of the screen with a trend line off the top of this symbol. This trend line had three segments of 5 units each. Each of the segments represented 30 seconds of projected aircraft movement. The whole trend line provided 90 second prediction, given the current speed, altitude and heading. The ND also displayed the flightpath pre-programmed in the FMS/CDU. The ND displayed the names of all remaining waypoints within 20 nm and provided the crossing altitude and speed restrictions for the next waypoint. The aircraft symbol was annotated with the current speed (KIAS) and altitude. The ND also displayed the current heading at the top of the display, and the name of, and distance to the next waypoint in the upper right corner.

Engine Instrument Display

The engine instrument display, located to the right of the PFD, presented engine parameter information, including engine pressure ratio (EPR), N1, and fuel flow and capacity values in a format similar to current aircraft. This information was not specifically manipulated or required by the experimental scenarios but is fundamental to piloting.

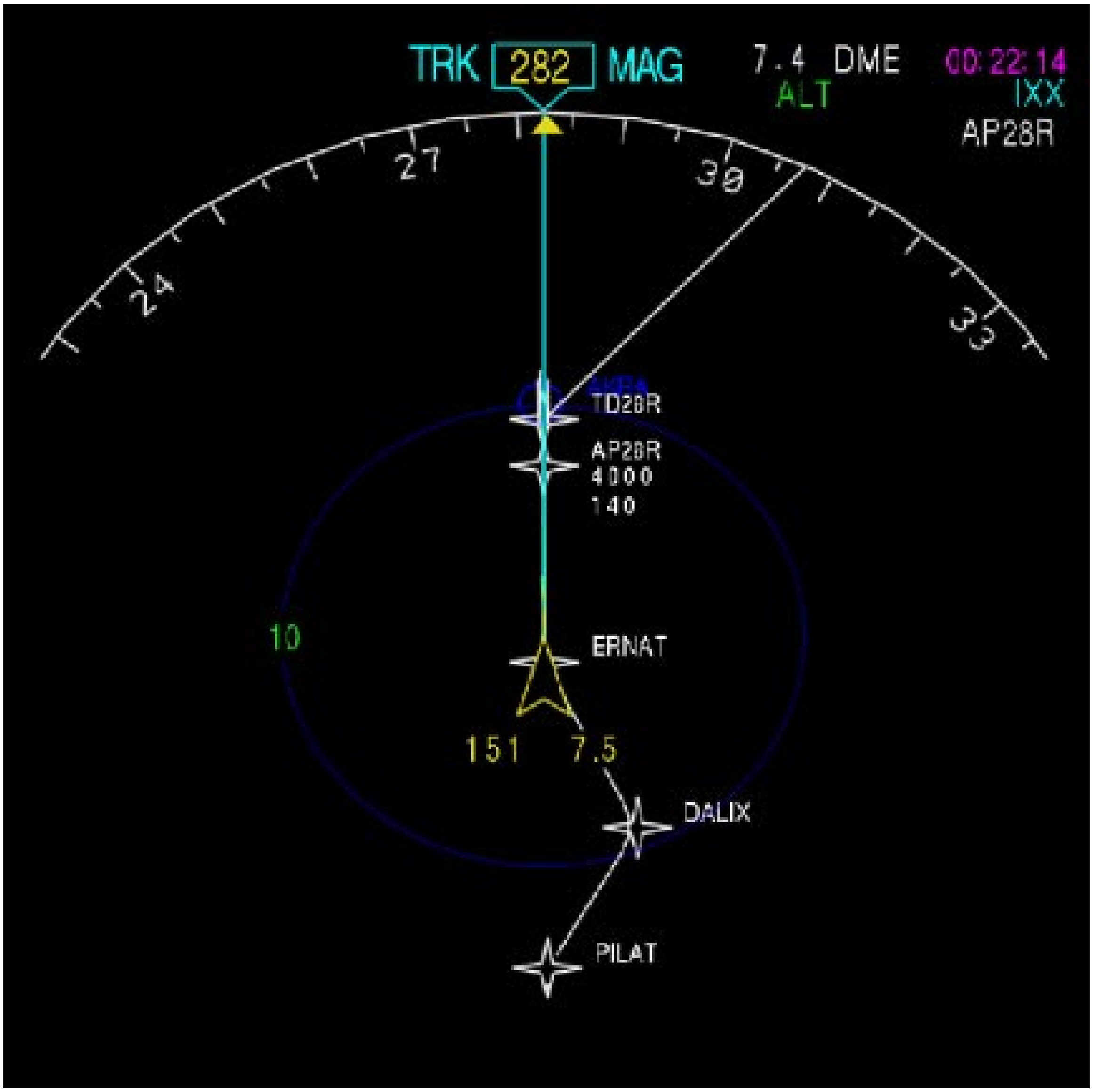


Figure 5.2. TSRV-IIC Navigational Display

Flaps Information Screen

The flaps information screen, a CRT located to the right of the engine instruments display, presented a flaps schedule according to pre-defined minimum speeds. This schedule was based on the specifications of the Boeing 737 manual on limit speeds and adapted for this experiment's flightpath requirements.

Checklist Touchscreen

A touchscreen display presented relevant checklists organized in a simple menu structure (Appendix 5.4) below the engine instrument display. Upon touching the screen, red cross-hairs were displayed to provide subjects localization and selection feedback. The contents of these checklists were based on the Boeing 737 training manual and modified for this experiment. This experiment only required pilots to use the approach and Final Descent checklists (Appendix 5.5). The menu structure required two selections to access each of these checklists. The checklist implementation did not include any facility for place-keeping and reverted to the Main Menu after 30 seconds of inactivity.

Datalink Touchscreen

Various implementations of the datalink concept have been suggested (Kerns 1990). This experiment's implementation provides a limited datalink menu structure on a dedicated (*cf.* Hinton and Lohr 1988; Williams 1995) CRT touchscreen (*cf.* Knox and Scanlon 1990) to the right of the checklist system. The datalink touchscreen provided subjects with localization and selection feedback similar to that provided by the checklist system. The experimental scenario allowed pilots to interact with the datalink system only to receive ATC messages and to respond to these messages in a very limited manner. As such, none of the labels on the initial Main Menu screen were touch-sensitive. When an ATC message was transmitted to the flightdeck, a mechanized voice announced "incoming message" and the screen changed to one presenting the ATC instruction in a text box and two touch-sensitive labels, ROGER and STAND-BY. Selecting ROGER signaled ATC that the flightdeck recipient planned to accomplish the contents of the message immediately. Upon selecting ROGER, the datalink system reverted to the Main Menu screen. Selecting the STAND-BY label signaled ATC that the flightdeck recipient had received the message, but did not plan to accomplish the task immediately. Upon selecting the STAND-BY label, the label outline and text turned green and the ATC message remained until selecting ROGER. Appendix (5.6) displayed the datalink initial screen and a sample ATC incoming message.

Central Quadrant

The TSRV-IIC's central quadrant was standard for a Boeing 737-300. The central quadrant included the speedbrake, throttle, and flap controls. The scenario was designed to require manual throttle control, no speedbrakes, and for flaps to be selected according to the specified schedule and procedure instruction.

Communication Systems

This experiment used two communication channels, COM1 and COM2. Each of these channels had two tuning heads. Each communication channel had a transfer toggle switch (TFR) that selected the active tuning head. Pilots selected the transmitting channel using the transmit-selector knob. Selecting a communication channel allowed the pilot to hear and transmit to the frequency dialed on the active head of that channel. Pilots communicated to passengers by tuning the transmit-selector knob to the public announcement (PA) position. Pilots could listen to additional active frequencies by toggling the associated "listen-to" switches. The subject wore a headset microphone. Communication channels were opened for flightdeck transmission by holding down either the trigger switch on the sidestick controller, or a button under the front edge of the subject's left armrest. Using the armrest microphone switch minimized inadvertent control inputs that could occur when using the sidestick controller's trigger switch. Communications from other agents or mechanisms in the simulation were presented through speakers in the simulator cab behind the subject.

Overhead Panel

The TSRV is not equipped with any of the standard B-737-300 overhead panel controls or displays. For this experiment, several simple discrete, back-lit buttons were implemented on the overhead panel for functions required in the 18K' and FAF procedures. Specifically, buttons were designed to control and indicate the status of the seatbelt sign, no-smoking sign, landing lights, anti-skid, and autobrakes (Figure 5.3). The anti-skid and autobrakes were mechanically related. If the anti-skid was not on, autobrakes could not be selected. Once both are selected, deselecting anti-skid also deselected autobrakes. These overhead panel buttons were a dimly-back-lit green when off and brightly back-lit green when on. In addition to these buttons, the overhead panel also contained a display for leading edge devices, the gear handle, and gear position indicator lights.

FMS/CDU

The Flight Management System (FMS) interfaces with other computers and systems in the aircraft to provide automatic navigation, guidance, map display, and in-flight performance optimization. The FMS receives pilot input and displays information to the pilot through the control display unit (CDU). Together, this system is referred to as the FMS/CDU (Figure 5.4). The FMS/CDU's Legs page provided the most useful information for normal flightpath monitoring. This page listed the remaining waypoints of the flightpath, their corresponding crossing restrictions, and headings and the distance between these waypoints. The Legs page also displayed the distance from the aircraft's current position to the next waypoint. At the onset of a run, most of the scenario flightpath is pre-loaded and the Legs page lists all waypoints and distances up to the final approach fix. Selection of the appropriate runway augments the Legs page for the remaining three waypoints, the approach point, the touchdown point, and the missed approach fix. The FMS/CDU interface in the TSRV-IIC was very similar to current commercial aircraft.

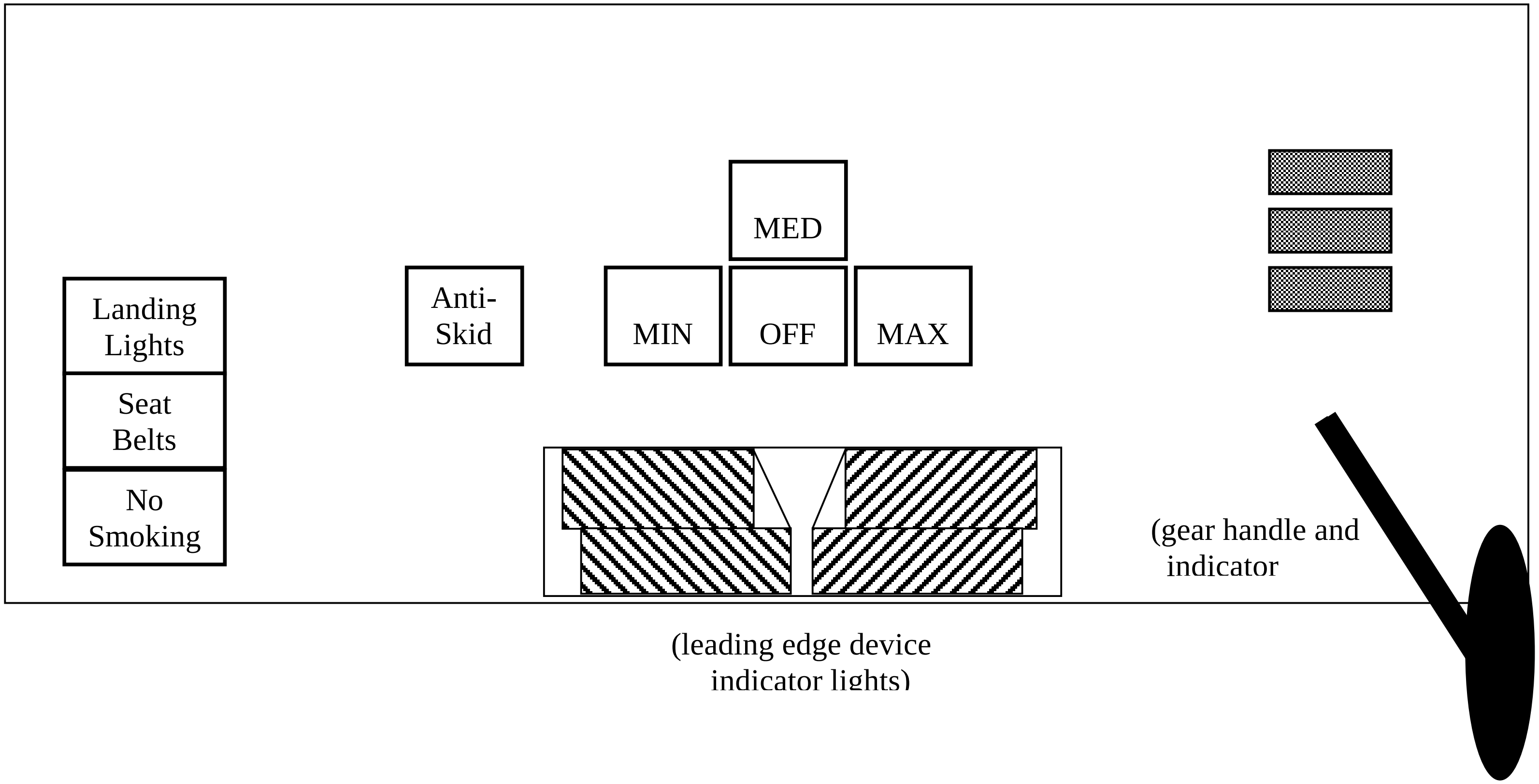


Figure 5.3 A Schematic of the TSRV-IIC Overhead Panel

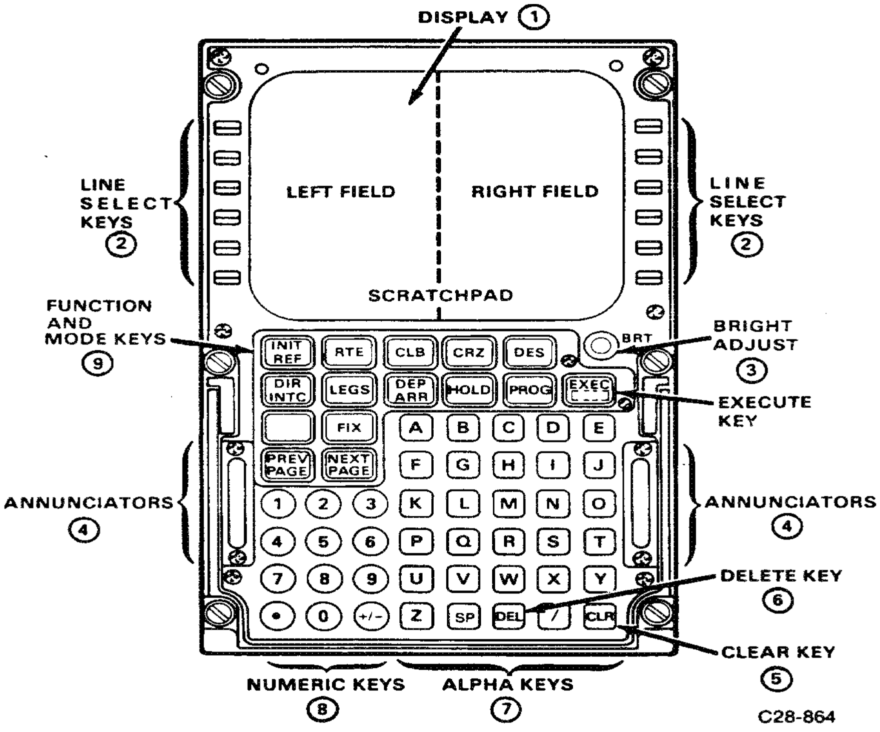


Figure 5.4. The TSRV-IIC Flight Management System's Control Display Unit

TSRV-IIC Ambient Characteristics

The experimental scenario did not include winds, nor did it provide subjects with an external visual scene. Light levels in the simulation cab were low to facilitate video recording and de-emphasize the lack of external visual scene. Engine sounds, presented through a speaker behind the subject, were approximately 60 dBa for 18 degrees of throttle at cruise-altitude (19000') and 290 KIAS.

ATIS Message System

In real airline operations, the Automatic Terminal Information Service (ATIS) provides a continuous broadcast of recorded airport terminal information to provide pilots with useful weather, and airport condition information. ATIS reports are typically 30 to 45 seconds in length. The experimental implementation of ATIS was almost identical to that in real operations. ATIS recordings were thirty seconds in length and played continuously throughout a run. Although the ATIS tape played continuously, the subject could hear the ATIS information only if he selected the ATIS tuning head on COM2 and either switched COM2's listen-to toggle switch up or selected COM2 with the transmit-selector knob. The ATIS tape repeated until the channel was deselected. ATIS messages were in a different, female voice to minimize interference with and the real-time transmissions from ATC and airline coordinating services and interrupting ATC requests which were presented by a male voice. ATIS messages were projected from a speaker behind the subject, at approximately 72 dBa.

Flightdeck-Initiated ATC & Company Communications

An experienced air traffic controller operated in real-time with the simulation to respond to flightdeck calls to ATC approach control, ATC tower, and the airline company's coordination services. Subjects interacted with this individual for procedurally-required calls, to acknowledge interrupting tasks, and, if necessary, to clarify previous transmissions. The controller's responses to company and tower calls were scripted for each run. The controller produced two intelligible microphone clicks as a response to subject acknowledgments to minimize interference with consequent procedural tasks. The controller also had all interruption annunciations and ATIS scripts so he could respond to queries from subjects and compensate for any communication equipment problems. Standardized responses were scripted for those queries most frequent among preliminary subjects. In non-standard interactions, the controller provided requested information as succinctly as possible. The controller did not offer helpful information or ensure that clearances were received, as would occur in normal line operations to minimize interference with subject performance and maximize experimental control. Procedural ATC and airline company communications were announced from a speaker behind the subject at approximately 74 dBa. The controller's sound level was calibrated at the beginning of each day and mid-day and the controller maintained a standard distance from the microphone for all real-time interactions.

Interrupting ATC Communications

A pre-recorded, automated system presented ATC interruptions through a speaker behind the subject. The voice used to record the ATC interruption scripts was that of the confederate performing real-time ATC communications to maximize scenario coherence. These voiced data files were associated with interruption positions in the procedural tasks and different interrupting tasks to operationalize experimental conditions (Appendix 5.7). Scripted ATC interruption annunciations occurred to the flightdeck when subjects performed the triggering activity of the intended experimental condition.

Scenario

The experimental scenario was created to incorporate several design goals in addition to the overarching goal of minimizing subject participation time. An overview of the rationale for scenario design decisions precedes a more detailed description of the physical characteristics and functional requirements of the scenario.

Scenario Design Rationale

The scenarios and, consequently, some features of the simulation apparatus, were designed to achieve several experimental goals. These goals, generally, were to: (1) minimize variability of factors not investigated in this study, (2) maximize operational validity, (3) operationalize independent variables, and (4) collecting dependent variables and introducing interruptions.

Minimizing Variability in Immaterial Factors

Goals for minimizing the variability of factors immaterial to this investigation included: minimizing unintentional distractions, minimizing the effect of individual differences in flightpath management technique, controlling the amount of externalized memory available, minimizing the effect of individual differences in familiarity with the experimental scenario, and minimizing learning and fatigue effects over runs.

To reduce the occurrence of distractions, no external scene was used in this simulation, display modification options on the ND and PFD were inoperative during runs, and flying techniques were designed to reduce flightpath deviations during procedural intervals. To control the FPM difficulty across subjects, subjects were instructed and trained to follow specific flying techniques (*e.g.*, selecting flaps according to a schedule) and to execute these techniques as cued by pre-programmed PFD display features. The amount of externalized memory available to subjects was controlled by requiring externalization of some information on the kneepad form and prohibiting subjects from noting any information not explicitly required on this form. Additionally, the checklist system was designed to revert to the main menu after a time determined to be just long enough to perform the checklist. This feature and specific instruction reduced the possibility that subjects would use checklists as externalized memory to guide procedures rather than as a verification task. The effect of individually-different familiarity with the experimental scenario was reduced by disguising the terminating airport; this was done by renaming it, changing its altitude, and creating fictitious waypoint locations and names surrounding it. Features of the flightpath that were assumed to have no bearing on performance requirements or information availability were varied to reduce monotony of repeated runs. These variations included using four orientations for approaching the terminal (corresponding to the four pairs of parallel runways), the direction of the two dog-legs in the flightpath, and by using non-imaginable, confusable waypoint names.

Maximizing Operational Validity

Goals for maximizing scenario operational validity included: maximizing the operational validity of performing procedures during the scenario; creating meaningful, definite, and appropriate start and end points for procedural intervals; and encouraging that subjects be immersed in the scenario before any experimental conditions occurred. To improve operational validity, the flightpath was designed to make obvious those legs in which subjects should perform procedures. This was accomplished by creating two types of legs, procedural intervals and non-procedural intervals. Procedural intervals were designed, based on preliminary subject data and assuming ACWS, to afford hands-off, zero-deviation flight long enough to perform the procedures. Non-procedural legs were designed by closely juxtaposing waypoint crossings, turns, and level-offs, to require intensive flightpath management. Non-procedural intervals were, therefore obviously not appropriate for procedure performance. Flightpath management difficulty was designed to peak around waypoints using hard crossing restrictions, tight turns, and manual throttle control to identify natural starting points for the procedural intervals. Procedural intervals also had deadline conditions of increased flightpath management difficulty imposed by level-offs or an abrupt speed reduction. The FPM difficulty of these termination points exacerbated the FPM requirements associated with turns and crossing restrictions at waypoints.

Operational validity was enhanced by anchoring these procedural intervals at meaningful points in the approach and descent. The first set of procedural tasks, the *top-of-descent procedure*, began after leaving cruise altitude, at the top of descent (TOD). The second set of procedural tasks, the *18 thousand-foot procedure*, began after descending from 18,000' (18K') a transitional altitude at which many commercial carriers reset the altimeter. The final approach fix (FAF) is an operationally significant point in the flightpath that some pilots use to check that the aircraft is configured for landing. The third set of procedural tasks, the *final approach fix procedure*, began after passing this point and were primarily concerned with aircraft configuration for landing. Subjects were provided with a short uneventful interval prior to the first procedural interval to encourage immersion in the scenario.

Operationalizing Independent Variables

Operationalizing independent variables fundamentally required a set of procedures, a set of interrupting tasks, and a flight phase in which to perform them. The approach and descent flight phase was chosen because it afforded natural opportunities for operationalizing independent variables. Preliminary interviews and testing refined the manner in which task factors were operationalized to increase external validity.

The environmental stress variable required opportunities for data collection at two different levels of environmental stress. Assuming that proximity to the ground and touchdown point imposed an increasing form of environmental stress, isomorphic procedures at 18,000 feet and 8,000 feet provide the conditions for this factor. The goal-level variable required a procedural task hierarchy with at least three levels of observable decomposition. Approach and descent phases naturally include many flightdeck and aircraft configuration tasks observable at the keystroke level. For this experiment, these tasks, and some additional flightdeck tasks, were arranged into three procedures. Levels of the coupling variable were operationalized by designing three pairs of adjacent tasks to supply three levels of coupling-strength. Two procedural tasks, similar in execution, and two interrupting tasks similar and dissimilar in semantic content to these procedural tasks were required to operationalize the similarity factor conditions. The modality variables required two types of interruption positions; one at the lowest goal-level of an auditory task, one at the lowest goal-level of a visual task. It also required two types of interrupting tasks; one which presented interrupting task information aurally, the other which presented this information visually. Finally, to isolate the effect of these independent factors, other interruption position and interrupting task characteristics were selected and designed to be as constant as possible; *e.g.*, the interrupting tasks were all initially announced aurally, required acknowledgment and entailed FMS/CDU tasks of similar length and complexity.

Collecting Dependent Variables and Introducing Interruptions

This experiment collected both reaction-time and error dependent measures. Interruption positions and interrupting tasks were designed and selected to require frequent physical interaction with the simulation equipment to enable keystroke-level time data. Simulation equipment was modified or specifically designed to sense and capture these interactions. This capability not only allowed keystroke-level data collection, but was necessary to trigger the introduction of interruptions at specific points in procedure performance. Subjects were trained to perform scripted procedures in a highly-constrained manner to not only define precise interruption triggering conditions, but also to provide a standard by which to define procedural performance errors.

Physical Characteristics of Scenario

The physical characteristics of the scenario include those of the terminal environment, flightpath profile, and flightpath plan views.

Terminal Environment

AKRA International Airport (Figure 5.5), a fictitious airport based on the design of the San Francisco International Airport, served as the terminal environment. AKRA had the runway configuration of San Francisco; *i.e.*, four pairs of parallel runways in a cross orientation: runways 1 left and right, 10 left and right, 19 left and right, and 28 left and right. AKRA’s terminal environment included two missed approach fixes, MAFAT and MAFAB. MAFAT was the missed approach fix for runways 1 left and right, and 28 left and right. MAFAB was the missed approach fix for runways 10 left and right, and 19 left and right.

Profile View of Flightpath

The flightpath profile was a complex, step-down, non-precision, instrument approach with crossing restrictions at each waypoint (Figure 5.6). These crossing restrictions specify the exact target altitude and speed to achieve at each waypoint. Each run used the same scenario flightpath profile. The subject began the scenario with this profile pre-loaded in the FMS/CDU minus the final three points; the approach point, the touchdown point, and the missed approach fix. These three points were added to the path upon selecting the destination runway.

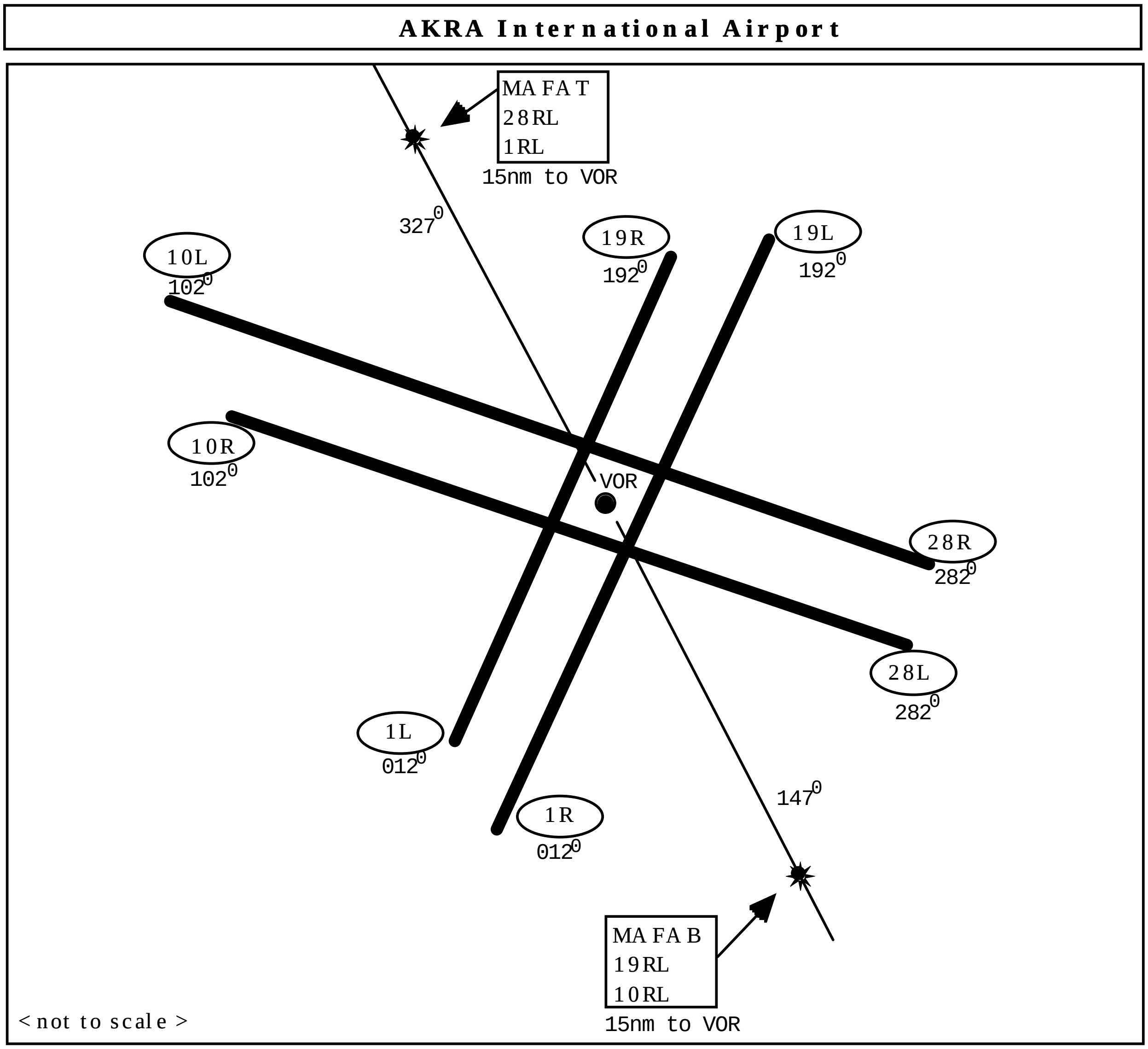


Figure 5.5. AKRA International Airport Terminal Area Map

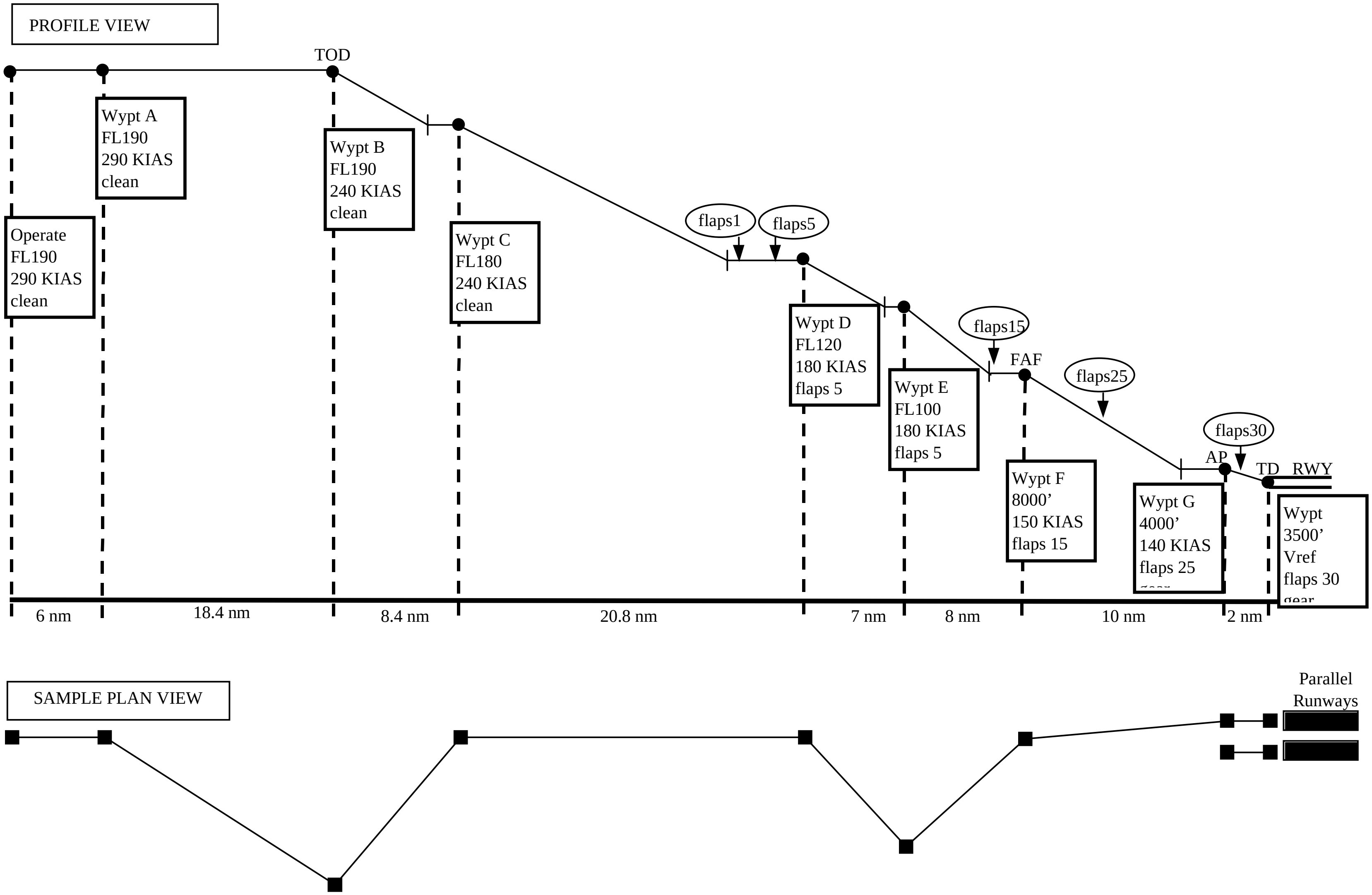


Figure 5.6 Profile View of the Scenario Flightpath.

Plan View of Flightpath

All plan views had the same basic features (Figure 5.7). That is, all leg distances, number of turns, and turn radii were the same. All plan views were aligned to the center of a pair of parallel runways and had two 'doglegs' from this initial heading; as if there were two obstacles to performing a straight-in approach. Some features of the plan view varied. Approaches were oriented to all four pairs of parallel runways. Each of the 'doglegs' in the path could be either to the left or right, independently, creating four configurations. These variations, four initial headings and four configurations, defined sixteen possible plan views (Appendix 5.8). Because the waypoints defined by these plan views occupied different positions in physical space, waypoint names also varied for a given position depending on the plan view (Appendix 5.9).

Functional Requirements of Scenario

The functional requirements of the scenario are described in terms of the pilot’s role, flightpath management performance requirements, procedure performance requirements, interrupting task requirements, and integration requirements as follows:

Pilot Roles

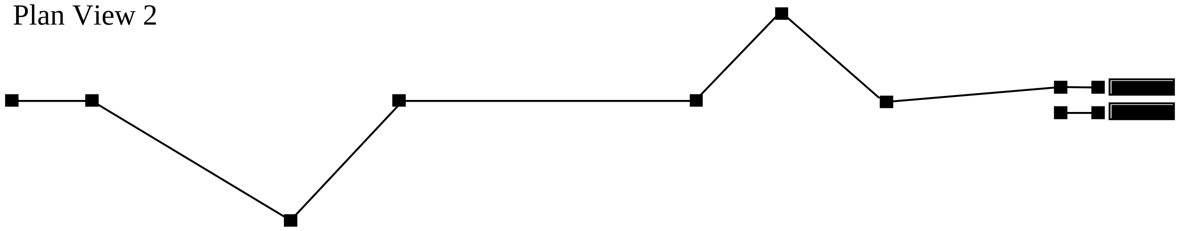
Subjects performed according to single-crew member operation rules; that is, subjects were told to assume responsibility for performing both Pilot-Flying (*e.g.*, flightpath management, FMS/CDU entry) and Pilot-Not-Flying (*e.g.*, communications, checklists) duties. The scenario required single-crew member operation to increase workload and ensure intended task loading on the subject.

Flightpath Management

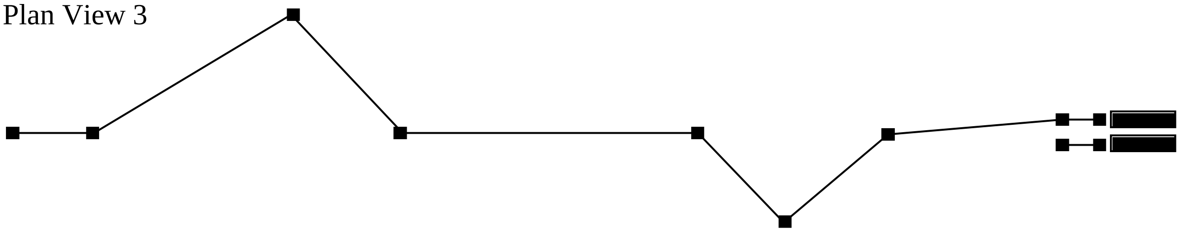
The flightpath was designed to induce a specific profile of FPM difficulty over the scenario (Figure 5.8). The flightpath contained three procedural legs of purported low-FPM difficulty. These three low-FPM difficulty legs were separated by higher-FPM difficulty legs. These higher FPM legs maximized independence of procedural legs, minimized active rehearsal before procedural legs, and emphasized that procedures were to be conducted entirely within the designated legs. Flightpath management demands were designed to peak at waypoints to further emphasize procedural leg deadlines. Purported FPM difficulty was designed by manipulating the number of parameters requiring adjustment at any point in time, requiring subjects to use ACWS and manually manage throttles, and requiring subjects to perform flightpath management actions in response to PFD features.

Plan View 1

Plan View 2



Plan View 3



Plan View 4

Figure 5.7 Plan Views of the Scenario Flightpath.

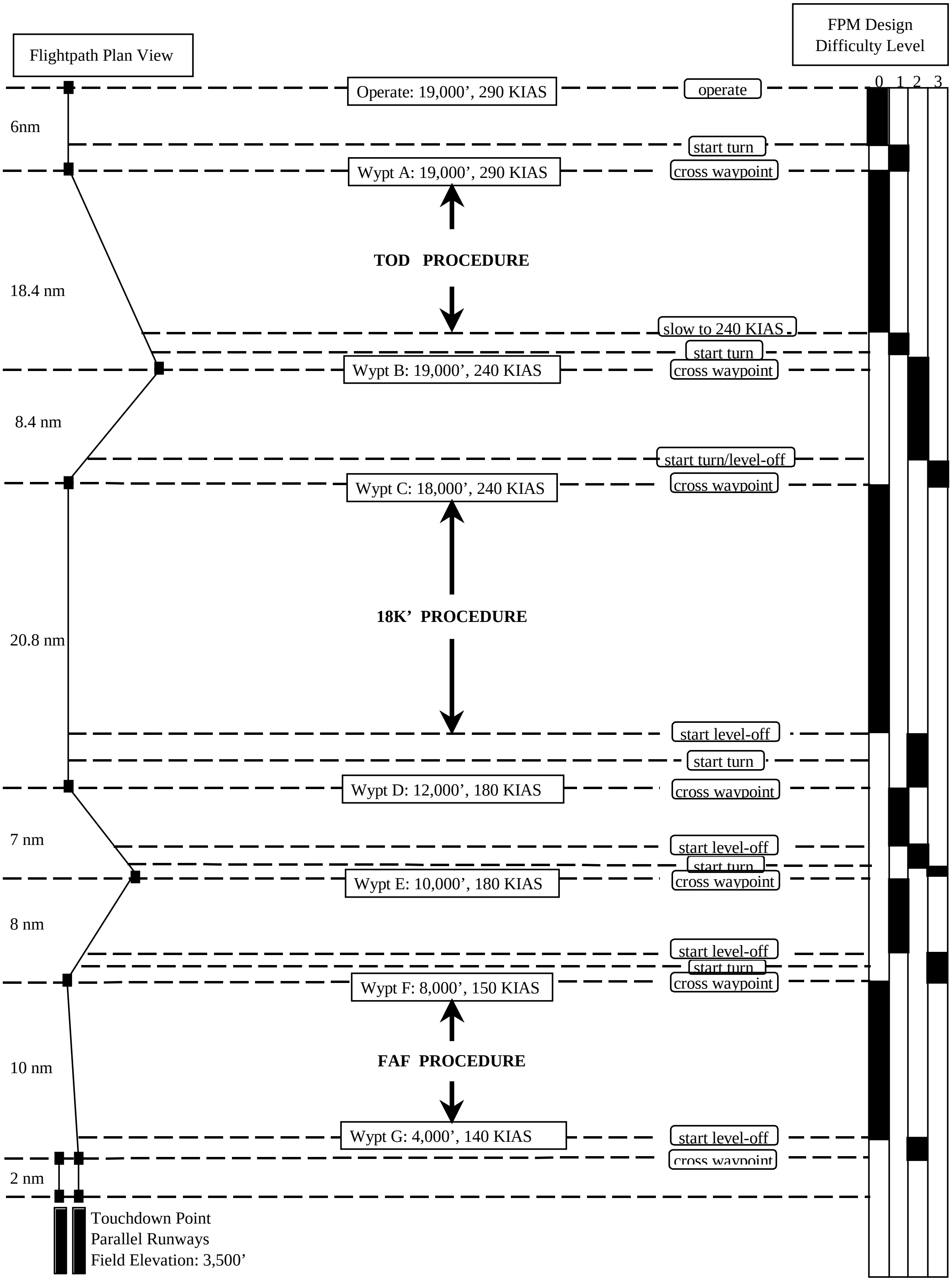
In addition, FPM difficulty and the time available in procedural intervals was controlled, to some degree, across subjects by requiring specific FPM techniques:

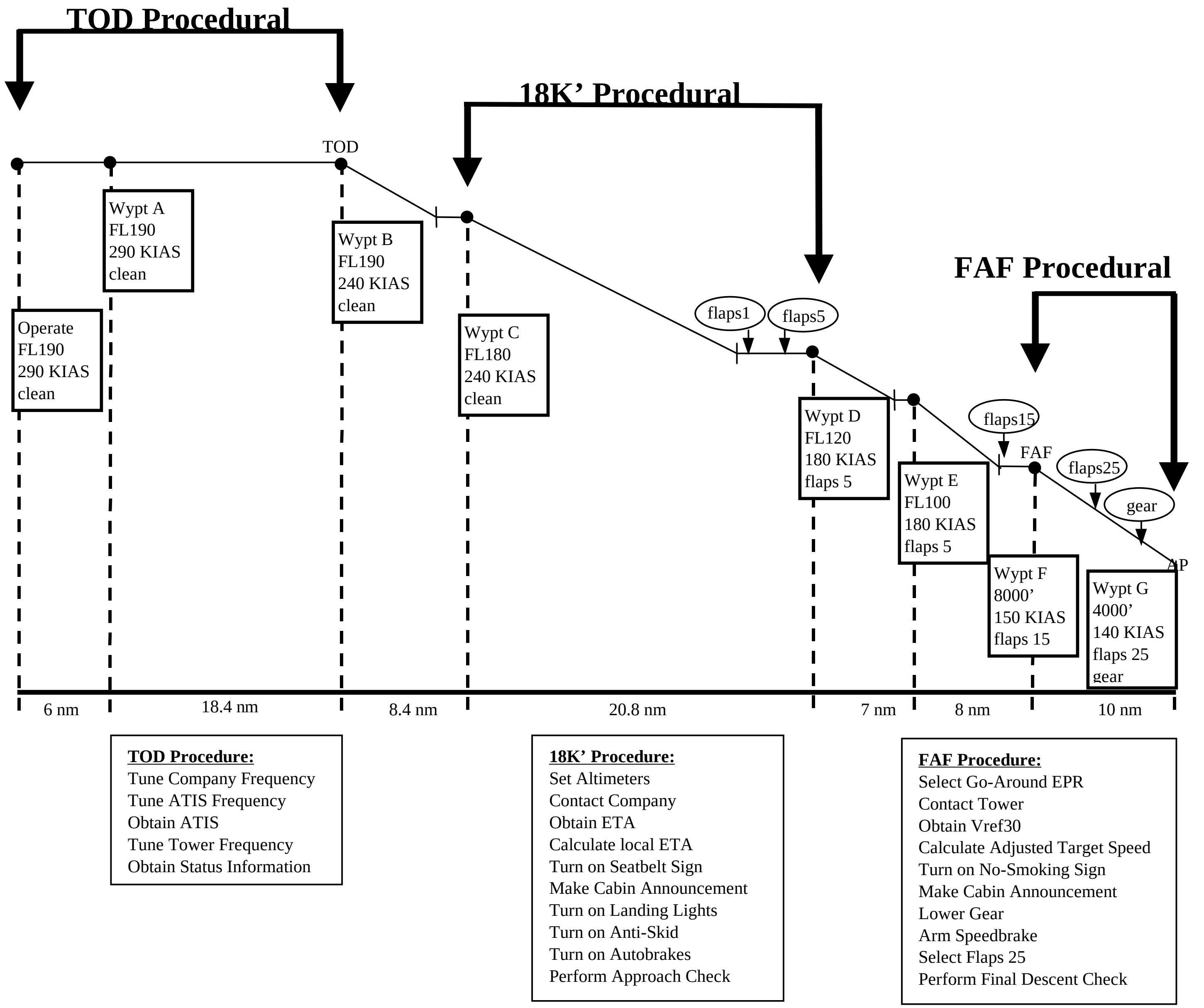
(1) Crossing restrictions were to be accomplished by first descending and then, if necessary, slowing only at 300' above the target altitude during the level-off.
(2) All descents were to be performed at idle power.
(3) All descents were to be performed at the specified FPA attitude for that interval.
(4) Flaps were to be taken according to the speed schedule.
(5) Turns were to be initiated only in response to thumbtack movement.
(6) On turns, the initial bank angle was to be approximately 20 degrees.
(7) Speedbrakes were to be avoided and used only to stabilize before procedural intervals.

There were a few exceptions to these rules. The first crossing restriction required no descent. The final descent to the runway required additional thrust to attain the adjusted target speed for landing. Flaps 25 and 30 were not selected according to the speed schedule but rather according to the FAF Procedure and upon seeing the runway was in sight, respectively.

Procedure Performance

Figure 5.9 displays the procedural intervals on the flightpath profile view and provides a task-level description of the TOD, 18K', and FAF procedures. The 18K' and FAF procedures were designed to be isomorphic; that is, the flow of the tasks and the task types were similar at each step. During preliminary testing of the scenario, it was evident that performing these procedures without any form of external memory was, in addition to unrealistic, unfeasible. A kneepad form was designed to allow restricted externalization of memory items. This kneepad form contained ATC and company radio frequencies, and the go-around EPR reference material, and provided blanks for noting other information, *i.e.*, the tower frequency, altimeter, inoperative items, estimated local time of arrival (ETA), destination gate, and adjusted target approach speed (Figure 5.10). Subjects were instructed that notations on the kneepad forms, other than those required by blanks, would be considered errors in performance. Subjects received a new kneepad form for each run with different reference information. The following sections describe in more detail the performance of these procedures. Appendix 5.10 provides an activity-level description of each procedure.


Flightpath Plan View
FPM Design Difficulty Level
0 1 2 3
Operate: 19,000’, 290 KIAS
operate
6nm
start turn
Wypt A: 19,000’, 290 KIAS
cross waypoint
TOD PROCEDURE
18.4 nm
slow to 240 KIAS
start turn
Wypt B: 19,000’, 240 KIAS
cross waypoint
8.4 nm
start turn/level-off
Wypt C: 18,000’, 240 KIAS
cross waypoint
18K’ PROCEDURE
20.8 nm
start level-off
start turn
Wypt D: 12,000’, 180 KIAS
cross waypoint
7 nm
start level-off
start turn
Wypt E: 10,000’, 180 KIAS
cross waypoint
8 nm
start level-off
start turn
Wypt F: 8,000’, 150 KIAS
cross waypoint
FAF PROCEDURE
10 nm
Wypt G: 4,000’, 140 KIAS
start level-off
cross waypoint
2 nm
Touchdown Point
Parallel Runways
Field Elevation: 3,500’



TOD Procedure:
Tune Company Frequency
Tune ATIS Frequency
Obtain ATIS
Tune Tower Frequency
Obtain Status Information

18K’ Procedure:
Set Altimeters
Contact Company
Obtain ETA
Calculate local ETA
Turn on Seatbelt Sign
Make Cabin Announcement
Turn on Landing Lights
Turn on Anti-Skid
Turn on Autobrakes
Perform Approach Check

FAF Procedure:
Select Go-Around EPR
Contact Tower
Obtain Vref30
Calculate Adjusted Target Speed
Turn on No-Smoking Sign
Make Cabin Announcement
Lower Gear
Arm Speedbrake
Select Flaps 25
Perform Final Descent Check

Figure 5.9 Profile View of the Scenario Flightpath.

Company Frequency 119.50

ATIS Frequency 124.20

ATIS Information

altimeter _______

tower frequency _______

INOP Items

CADC1 _______

CADC2 _______

IRS _______

RADAR _______

TRANS1 _______

TRANS2 _______

COMM1 _______

COMM2 _______

COMM3 _______

NAV1 _______

NAV2 _______

NAV3 _______

Gate _____

ETA-Local _____

GA-EPR 2.153

Adjusted Target Speed _____

Figure 5.10. Kneepad form.

Performing the TOD Procedure

To accomplish the TOD procedure, the subject referred to the kneepad to tune the company's frequency, then the ATIS frequency, he listened to the ATIS (noting the altimeter setting, braking conditions, and tower frequency on the kneepad form), tuned the tower frequency, and obtained status information from the FMS/CDU (recording inoperative items on the kneepad form). While the altimeter setting was obtained in the TOD procedure, the subject actually entered the altimeter setting as the first task in the 18K' procedure. In actual airline operations, braking advisories are only included in ATIS if conditions are poor. In this experiment ATIS always advised whether runway braking conditions were good, fair, or poor, corresponding directly to the level of autobrakes required; minimum, medium, and maximum. Tower frequencies are normally obtained from published approach plates. For this experiment, subjects were told that the published tower frequencies were incorrect and that ATIS would convey the correct tower frequency as a Notice to Airmen (NOTAM). Sets of items were selected for inoperative status such that at least one redundant device for each system was operative. The subject was also told that inoperative items would have no consequence for the aircraft's operability due to this redundancy.

Performing the 18K' Procedure

To accomplish the 18K' procedure, the subject first refereed to the kneepad form for the appropriate altimeter setting and entered it in the CDU. The subject then informed the airline company's coordinating services of the inoperative items indicated on the STATUS page and obtained gate information. After calling the company, the subject obtained ETA-Zulu time (Greenwich mean time) from the FMS/CDU and converted it to ETA-local time by subtracting five hours. The subject then reached to the overhead panel to turn on the seatbelt sign. The next task was to inform the passengers that the Seatbelt sign was on, and to provide the ETA and gate information. Following the cabin announcement, the subject turned on the landing lights. Then the subject turned on the anti-skid and selected the appropriate level of autobrakes. Subjects were instructed to select medium autobrakes if they did not remember the braking conditions in the ATIS. Finally, the subject performed the Approach checklist. Subjects were told to read the checklists aloud and to announce that autobrakes were set to "default" if they were unable to remember the braking conditions.

Performing the FAF Procedure

To accomplish the FAF procedure, the subject first referred to the kneepad for the appropriate go-around (GA) EPR setting and entered it in the CDU. The subject then conveyed the aircraft's location, the name of and distance to the next waypoint, to the tower and obtained the landing winds. After the tower transmission, the subject obtained the correct reference speed for the flaps 30 landing configuration (VRef30) from the FMS/CDU and calculated the adjusted target speed. While normally it is adequate to estimate this value, for this experiment the subject was asked to calculate it exactly using the following formula:

$$\text{(5.1)} \quad \text{adjusted target speed} = \text{VRef30 speed} + 0.5 * \text{steady wind (knots).}$$

Subjects were to use this value as the target speed for the final leg of the scenario. After calculating the adjusted target speed, the subject reached to the overhead panel and turned on the no-smoking sign. The next task was to inform the passengers that the no-smoking sign was on, and to prepare for landing. After completing the cabin announcement the subject lowered the gear, armed the speedbrakes and selected flaps 25, a landing configuration for this aircraft. Finally the subject performed the Final Descent checklist by reading it aloud.

Procedure Performance Techniques

Subjects attempted to adhere to the following techniques when performing procedures:

(1) Procedures were performed in the appropriate flightpath interval.
(2) All tasks within each procedure were performed as specified in training.
(3) Tasks within each procedure were performed serially and in the specified order.
(4) The kneepad form were used in performing the procedures such that all specified fields were filled and no other information was noted.

Interrupting Task Performance

The interrupting tasks (ITs) included: entering the initial approach, changing to the parallel runway, setting up a holding pattern, and changing the crossing speed and altitude at the missed approach fix. The following sections describe the performance requirements of each interrupting task. Appendix 5.11 presents activity-level descriptions of the interrupting tasks.

Entering the Initial Approach

Initial approach clearances were either auditory ATC calls or visual datalink screen presentations. For an auditory presentation, the subject acknowledged the transmission by returning a radio call to ATC. For a datalink presentation, the subject touched either the ROGER or the STAND-BY label to acknowledge the transmission. The subject selected ROGER only if he intended to enter the approach in the FMS/CDU at that time. To enter the initial approach, the subject accessed the Departure/Arrivals page in the CDU by pressing the DEP/ARR key, selected the Arrivals page, selected the appropriate runway, executed this revision by pushing the EXEC key, and finally returned to the Legs page. Upon executing this revision, the FMS/CDU revised the path to include the approach point, a touchdown point, and a missed approach fix associated with the desired runway. The initial runway was always one of the two parallel runways associated with the initial flightpath heading.

Changing Runways

Auditory ATC calls introduced runway changes. Runway changes were always a sidestep to the parallel runway. To change a runway, the subject accessed the Departure/Arrivals page in the CDU by pressing the DEP/ARR key, selected the Arrivals page, selected the new runway, executed this revision by pushing the EXEC key, and finally returned to the Legs page. Upon executing this revision, the FMS/CDU revised the path to change the approach point, touchdown point, and missed approach fix to that for the new runway.

Setting up a Holding Pattern

Auditory ATC calls introduced requests to set up a holding pattern in the CDU. These ATC calls always requested holding patterns to be set up at the missed approach fix. Holding pattern requests always followed the standard hold pattern already set in the CDU: that is, all holding patterns had right-turns and 1 minute legs. Subjects did not need to alter any of these parameters. To set up a holding pattern, the subject pressed the HOLD key on the CDU, selected the missed approach fix as the holding waypoint, executed this revision by pushing the EXEC key, and finally returned to the Legs page. Upon executing this revision, the FMS/CDU added four waypoints to the path between the touchdown point and the missed approach fix. These four waypoints defined the corner-posts of the holding pattern.

Changing Speed and Altitude Crossing Restrictions

Auditory ATC calls introduced requests to change altitude and speed restrictions. These ATC calls only requested changes to the restrictions at the missed approach fix. To change a crossing restriction, the subject typed the new restriction value into the FMS/CDU (typing a "/" after the value if it was a speed change), selected the missed approach fix on the last of the Legs pages, executed this revision by pressing the EXEC key, and finally returned to the first Legs page. Altitude changes required subjects to enter four digits. Speed changes required subjects to enter three digits followed by a backslash.

Interruption Performance Techniques

Subjects were required to acknowledge any interruption annunciation before taking any action to accomplish the interrupting task. For the aurally-presented interrupting tasks, subjects acknowledged by selecting the COM1 frequency for transmission and repeating the informative elements of the announcement for verification. For the visually-presented interrupting tasks, subjects acknowledged by touching either the STAND-BY or ROGER label on the datalink screen. Subjects were required to perform interrupting tasks according to the specified keystroke method.

Integration Requirements

Subjects were substantially restricted in how they conducted flightpath management, procedure performance, and interrupting task performance. Subjects were less constrained as to how they integrated the different aspects of the scenario. The three constraints on task integration were: (1) Procedures were to be performed wholly in the specified procedural intervals, (2) Flightpath deviations were to be nullified before beginning procedures or interrupting tasks, (3) All procedural tasks and any interrupting tasks that occur were to be finished prior to the next non-procedural interval.

Experimental Protocol

Subjects were mailed two items to complete before arriving for the experiment. Subjects participated in the experiment for two days. Subjects were trained on the first day and performed testing trials on the second day. The items in the pre-test mailer and the protocol for days 1 and 2 of the experiment are described below.

Pre-Test Mailer Items

The pre-test mailer included the subject background questionnaire (Appendix 5.12) and the task ordering exercise (Appendix 5.13). The subject background questionnaire obtained information about each subject's aviation experience, education, and demographic information. The Task Ordering exercise required subjects to order the tasks constituent to the TOD, 18K' and FAF procedures in the context of the experimental scenario's flightpath profile and a single-crew-member operation.

Day 1 Protocol

Subjects received a full day of instruction and training, divided into two phases (Table 5.1). Phase 1 occurred in a briefing room and familiarized the subject with the objective performance requirements of the scenario. Phase 2 occurred in the TSRV simulator and developed psychomotor skills for using the sidestick controller and following PFD guidance, and reinforced scenario objective performance requirements in context. The following sections briefly describes the training regime.

Phase 1 Training

The goal of training phase 1 was to provide an introduction to the TSRV-IIC, PFD display features, flightpath management performance requirements, and the procedures and interrupting tasks associated with the scenario. The experimenter provided phase 1 training information. The following sections briefly describe the phase 1 training process.

Welcome & Introduction

The experiment was introduced to the subject as an experiment in individual differences in a high-workload task environment, the goal of which was to characterize the manner in which pilots integrate manual flight performance, procedural flight deck tasks, and interrupting tasks during approach and descent. The introduction emphasized that the scenario was designed to be high-workload and that some aspects of the scenario were somewhat artificial. The subject received a description of the experiment's schedule and the measures to be collected. Finally the subject signed an informed consent form.

Introduction to Simulator & PFD Display Features

In this section of phase 1 training, the subject received an overview of the TSRV-IIC flightdeck and the TSRV-IIC's PFD display features. First, the subject was presented with a static picture of the TSRV-IIC flightdeck and the experimenter reviewed its major components. The subject then watched a videotape that described each of these components in detail and highlighted their usage in the experiment.

Table 5.1 Day 1 Experimental Protocol.

Day 1 Activity	min.	Start	Stop
Subject Arrives / Welcome	10	0800	0810
Training Phase 1: TSRV Overview, PFD, Flightpath Management	60	0810	0910
Break	10	0910	0920
Training Phase 1: Procedure Performance	90	0920	1050
Break	10	1050	1100
Training Phase 1: Interrupting Task Performance, Review	30	1100	1130
Break	5	1130	1135
Physiological Sensor Placement	15	1135	1150
Lunch	45	1150	1235
Training Phase 2: Flightpath Management (runs 1,2,3)	60	1235	1335
Break	10	1335	1345
Physiological Apparatus & Sensors Connected	10	1345	1355
Subjective Evaluation of Workload (runs 4,5,6)	60	1355	1455
Physiological Apparatus & Sensors Disconnected	5	1455	1500
Break	10	1500	1510
Off-line Procedure/Incidental Task Training: (runs 7,8)	50	1510	1600
In-context Procedure Training: (runs 9,10)	45	1600	1645
Break	10	1645	1655
Physiological Apparatus & Sensors Connected	10	1655	1705
Whole-Scenario Training: (runs 11,12,13)	60	1705	1805
Physiological Apparatus & Sensors Disconnected	10	1805	1815

negotiating turns (Appendix 5.15). The subject watched a videotape segment of the PFD and ND during the scenario's final 4000' level-off and landing to reinforce these concepts. An expert pilot narrated this videotape describing how PFD display features are used in vertical and lateral tracking.

Introduction to Flightpath Management Performance

The subject received figures and text describing the scenario's terminal environment, profile view, and plan views. The subject was told to assume that AKRA International airspace had been cleared and to expect to perform the approach as preprogrammed in the FMS/CDU. The subject received detailed descriptions of the techniques required for successful flightpath management performance and rules for anticipating guidance in the PFD (Appendix 5.16). Finally, the subject viewed a videotape of the PFD and ND as an expert pilot narrated the techniques and PFD guidance for the entire scenario. The subject was encouraged to refer to the profile view and rules for PFD guidance, while watching this videotape.

Introduction to Procedure Performance

The subject received a figure depicting the location of the three procedural intervals on the flightpath profile, definitions of the three procedures at the task and activity levels, and an example of the kneepad form. The subject also received a description of how to perform each of the three procedures, how to use the kneepad form, and the general techniques required for successful procedure performance. A videotape, narrated by an expert pilot, demonstrated correct performance of each procedure in the context of its flightpath interval. Following these descriptions, subjects practiced each procedure using a mock-up of the TSRV-IIC flightdeck, and actual kneepad forms. The experimenter simulated communications from other agents/mechanisms in the scenario. Subjects practiced each procedure until they were able to perform all three procedures from memory, without error twice. Following this practice, the subject was asked to complete the sequential coupling task with respect to performing the tasks as instructed and using the TSRV-IIC equipment and experimental materials (Appendix 5.17).

Introduction to Interrupting Task Performance

The experimenter informed the subject that, in order to make the scenarios more realistic and dynamic, ATC communications may require him to accomplish additional, “incidental” tasks. The experimenter informed subjects that these incidental tasks were so termed not because they were unimportant, but because they would occur at unspecified times throughout the scenario. The subject received a text description, a table describing activity-level performance, and a narrated videotape segment as instruction for performing each incidental task. The subject practiced performing incidental tasks on a TSRV-IIC mock-up simulator until able to perform each without error twice.

Review of Performance Requirements

Phase 1 of training concluded with a review of the techniques that defined successful performance for each aspect of the scenario and for integrating scenario aspects.

Phase 2 Training

The second phase of training occurred in the TSRV and reinforced Phase 1 instruction in context. Phase 2 presented subjects with the three aspects of the scenario, flightpath management, procedure performance, and interrupting task performance, hierarchically. First, subjects practiced FPM techniques and then rated the difficulty of FPM over the scenario. Subjects then performed the three procedures in the context of the scenario’s FPM requirements. Finally, subjects performed the complete scenario; including FPM, procedures, and interrupting task performance requirements. Both the experimenter and an expert pilot provided information during phase 2 training. The expert pilot was responsible for training subjects on FPM techniques in early runs, and for assessing FPM performance during all runs. The experimenter was responsible for training subjects on procedural and interrupting task performance. During phase 2, subjects were instrumented to collect physiological data for a related study.

Accommodation to Simulator

Upon arriving in the TSRV, subjects were quickly re-introduced to the major elements in the simulator and made adjustments to the seat and rudder pedals.

Flight Path Management Training

One of the expert pilots served as the FPM trainer during phase 2. Before beginning each of the flightpath management training runs, the trainer performed the *Flightpath Management Review* exercise (Appendix 5.18) with the subject to reinforce use of PFD display features and required FPM techniques. On the first run, the trainer provided standard information on FPM techniques at specific points in the flightpath (Appendix 5.19) and customized instruction on compensatory FPM techniques when necessary. Prior to the second and third runs, the trainer performed the Flight Path Management Review again with the subject. During the second and third runs, the trainer encouraged the subject to provide verbal protocols during the runs. The trainer provided only attention-directing and compensatory instruction during these two runs, withdrawing more on each successive run.

Subjective Difficulty Assessment

During the next three runs, the subject provided subjective assessments of flightpath management difficulty at point estimates of approximately 1 nm intervals along the flightpath. The subject rated flightpath management difficulty using the Bedford scale (Lysaght *et al.* 1989) (Figure 5.11). The Bedford scale obtained subjective judgments about workload based on ability to complete tasks and the amount of spare capacity available (Lysaght *et al.* 1989). " The experimenter presented the Bedford scale for review and instructed the subject on its usage. The subject was instructed to provide a subjective rating upon hearing the experimenter say the word "rating". In response to this prompt, the subject provided a Bedford rating indicating the perceived spare capacity available to perform an additional task at the time of query; *i.e.*, to answer an ATC call to modify a crossing restriction in the FMS/CDU. The experimenter instructed the subject to respond as soon after the query as possible while using the scale. The scale was displayed on a card placed over the checklist CRT, within easy view, during the assessed runs. The experimenter instructed the subject to not talk during these runs other than to provide subjective assessment ratings. The trainer observed the subject from a remote location during these three runs, and provided critique of FPM performance at the conclusion of each run.

Procedure Training

Prior to actually performing the procedures, the subject reviewed the procedures and familiarized himself with performing the procedures using the actual TSRV-IIC equipment. The subject performed two runs in which he performed each of the procedures and each of the interrupting tasks twice without attending to flightpath management. On the second of these context-free runs, the experimenter encouraged the subject to perform the procedures and interrupting tasks as rapidly as he could without error.

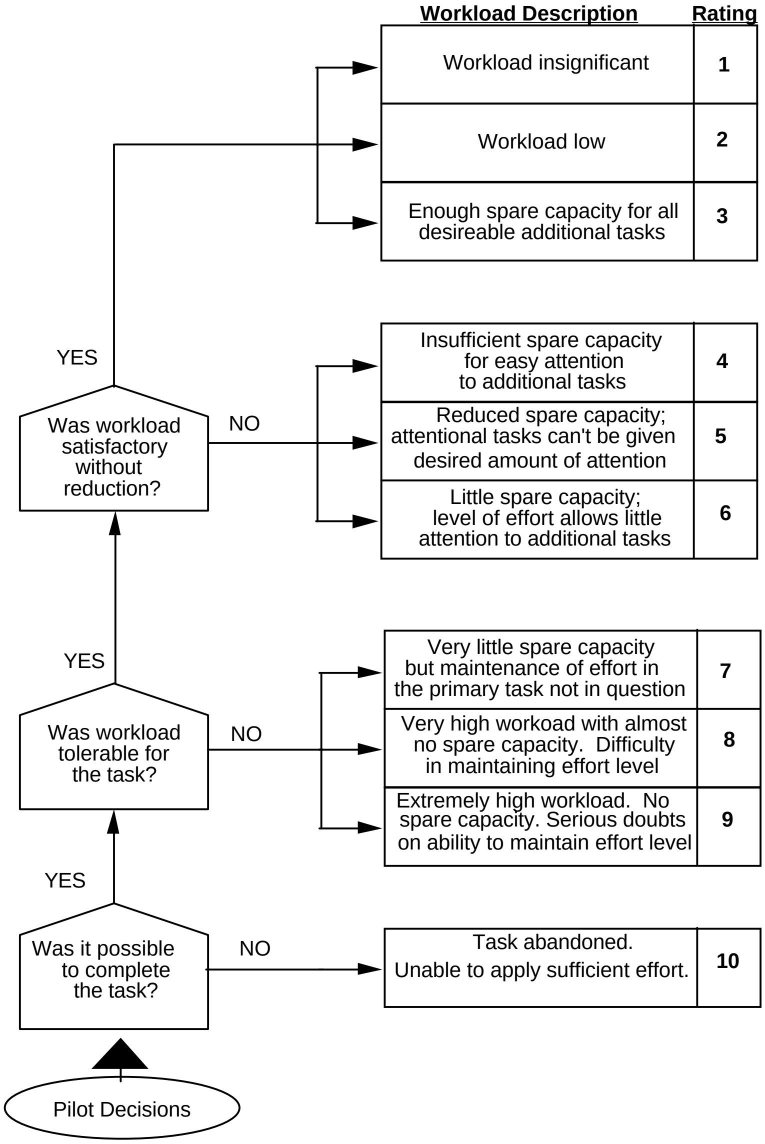


Figure 5.11 The Bedford Scale (Lysaght *et al.* 1989)

After practicing the procedures four times during the previous two runs, subjects performed two runs in which they both managed the flightpath and performed procedures. During these runs, the trainer observed from another room and the experimenter sat in the right seat. The experimenter intervened only in the event of a serious flightpath management problem, or to correct errors in procedure performance. The trainer provided a critique of flightpath management performance at the conclusion of each run and the experimenter reviewed procedure and interrupting task performance.

Interruption Management

The final segment of the phase 2 training regime provided the subject with three runs of the full scenario; including the flightpath management, procedure performance, and interruption integration. The trainer and experimenter acted in the same capacity as in the previous two runs. The three whole-scenario runs provided the subject with an instance of each interruption type and demonstrated early and late interruption positions in procedures (Table 5.2). The trainer provided a critique of flightpath management performance at the conclusion of each run and the experimenter reviewed procedure and interrupting task performance.

Table 5.2 Composition of Phase 2 Whole-Scenario Runs

Run #	Procedure	Interruption Task Type	Interruption Position
11	TOD	Initial Runway- Auditory	Before TOD Interval
	18K’	Change Runway	Within Approach checklist
	FAF	Change Speed Restriction	Within Go-Around EPR setting
12	TOD	Initial Runway- Visual	After tune tower frequency
	18K’	Establish Hold Pattern	Within Altimeter setting
	FAF	Change Runway	After Flaps 25 are set
13	TOD	Initial Runway- Auditory	Within obtaining Status
	18K’	Establish Hold Pattern	Before 18K’ Procedure
	FAF	Change Altitude Restriction	Within Final Descent checklist

Take-Home Materials

The subject received excepts from the phase 1 training manual as a take-home reference. These excerpts summarized the performance requirements for the three aspects of the scenario; flightpath management, procedure, and interrupting task performance. The take-home package also included an annotated figure of the scenario’s profile view, a sample kneepad form, and activity-level task analyses of the procedures and interrupting tasks.

Day 2 Experimental Protocol

Day 2 began with a review of scenario requirements, and allowed subjects three refresher runs before beginning data collection (Table 5.3).

Review of Scenario Requirements

Upon arriving, each subject reviewed performance requirements for the scenario then conducted the Flightpath Management Review. Finally, the experimenter reviewed procedure performance with the subject, asking him to recite each of the procedures from memory. The experimenter informed subjects of any errors in their recitation and provided subjects with a description of the procedures at the task level to review.

Table 5.3. Day 2 Experimental Protocol

Day 2 Activity	min.	Start	Stop
Subject Arrives		0745	0750
Scenario Review	15	0750	0805
Refresher Trials (runs 14,15,16)	60	0805	0905
Break	*10*	*0905*	*0915*
Testing Unit 1 (runs 17,18,19,20)	80	0915	1035
Break	*15*	*1035*	*1050*
Testing Unit 2 (runs 21,22,23,24)	80	1050	1210
Lunch	*50*	*1210*	*1300*
Testing Unit 3 (runs 25,26,27,28)	80	1300	1420
Break	*15*	*1420*	*1435*
Testing Unit 4 (runs 29,30,31,32)	80	1435	1555

Refresher Runs

Upon arrival to the simulator, the subject was reminded that the experimenter would not be able to answer any questions during a run but might need to intervene if a problem occurred with the simulation. The three refresher runs, runs 14, 15, and 16, exposed subjects to each interruption type. On run 14, the experimenter corrected FPM as well as procedure and interrupting task performance errors as soon as they were committed. Errors committed during runs 15 and 16 were discussed at the conclusion of run 16.

Testing Runs

During a testing run, the experimenter interacted only with the simulated ATC approach control, ATC tower, and company coordinating services. To save time, subjects landed only on run numbers 16, 20, 24, 28, and 32; however if a landing appeared extremely unstable, the experimenter aborted the simulation early and asked the subject to land on the next run. The subject was told that he would not be landing on all the runs in order to save time, but was not informed which runs would require landing. On non-landing runs, the experimenter informed the subject that the run was over shortly after flying beyond the approach point and the simulation was reset. The next simulation run began after the experimenter reset simulation switches, changed the ATIS information tape, provided the subject with a new kneepad form, and the subject indicated he was prepared to begin again. The average inter-run period was approximately three minutes. Subjects performed 16 data collection runs in four sets of four. Subjects received a 15 minute break after the first set of runs, approximately an hour lunch break after the second set of runs, and another 15 minute break after the third set of runs.

Experimental Conditions and Run Definitions

Experimental conditions were defined by the interrupting task and the point in the procedure at which the interruption occurs, the interruption position. The following sections describe pertinent characteristics of the interruptions, the interruption positions, their interaction to define experimental conditions, and the arrangement of experimental conditions to define experimental runs.

Interrupting Tasks

The five interrupting tasks required subjects to: (1) Enter the initial runway for the approach (IR), (2) Change to a parallel runway (CR), (3) Amend the flightpath to include a standard hold pattern at the missed approach fix (EH), (4) Change the crossing altitude for the missed approach fix (CA), and (5) Change the crossing speed for the missed approach fix (CS). The IR interruption had two variants; one condition was presented aurally (IRA) as a radio call, the other was presented visually (IRV), as a datalink message. The performance requirements for these interrupting tasks were previously described. This section emphasizes the construction of the interrupting tasks.

While this experiment investigated some factors hypothesized to affect interruption management, other factors were left unexplored. To minimize any effects these unexplored factors may have on interruption management, interrupting tasks were designed to be similar in several respects. These controlled characteristics included; modality of initial alert, urgency, announcement time, performance requirements, and the equipment interface. Subjects were initially alerted to all interrupting tasks by a voiced message. All interrupting tasks required changes in the terminal area and were therefore assumed to imply the same urgency. All interrupting tasks were FMS/CDU tasks, with approximately the same number of keystrokes, and the same structure (Appendix 5.20). Other characteristics were designed into the interruption task set to define certain experimental conditions, *i.e.*, modality of the interruption message, and similarity or dissimilarity to the interrupted task. The task set was designed such that IRA, CR, and EH could be considered replicates. IRA and IRV were designed to differ only in the modality of the interruption message. CA and CS were designed to differ only in the conceptual difference of changing altitude versus changing speed.

Interruption Positions

Most characteristics of interrupting tasks were designed to be constant. Experimental conditions were defined by interjecting these interruptions at different interruption positions. Several interruption positions were defined for each of the three procedures. Test conditions in the TOD procedure interjected interruptions at the following positions; before the subject performed the procedure, between two procedural tasks, within an auditory task, and within a visual task. Interruptions could be interjected at seven interruption positions in the 18K' procedure; before the subject started the procedure, after the subject finished the procedure, within a procedural task, and between physically-coupled, functionally-coupled, and uncoupled sequential procedural tasks. The FAF procedure was also interrupted in seven places. As the 18K' and FAF procedures were structurally isomorphic, so too were the interruption positions for these two procedures. Intervention positions in the FAF procedure were before the subject started the procedure, after the subject finished the procedure, within a procedural task, and between physically-coupled, functionally-coupled, and uncoupled sequential procedural tasks. Further, FAF interruption positions were in the same order and relative position as the 18K' interruption positions. In addition to these intervention positions, two null conditions, *i.e.*, uninterrupted procedure conditions, were constructed for each procedure.

Experimental Conditions

Experimental conditions were defined by pairing interrupting tasks with interruption positions (Appendix 5.21). Condition numbers contain the replication number, as the tens digit; the procedure number, as the ones digit; and an ordinal index of the experimental condition within this procedure, as the decimal component. These decimal values indicate both interruption conditions, by decimals values of 0.01 to 0.10, and uninterrupted conditions, by decimal values of 0.11 and 0.12. The set of experimental conditions tested was reduced from the originally designed set to accommodate time constraints, and therefore the decimal values of condition numbers' are not continuous.

Composition of Runs

Each run included three experimental conditions, one in each of the three procedural intervals (Appendix 5.22). Runs 1 through 13 were used for training. Of this set, runs 1-10 were uninterrupted to allow subjects to practice flightpath management and then FPM and procedural performance unimpeded by interruption training. Runs 11, 12, and 13 include interruptions in the scenario. Runs 14, 15 and 16 were refresher runs. Runs 17 through 32 were testing runs. The 16 testing runs were constructed as two replication blocks of eight runs. Blocks A and B provided exact replicates for the TOD conditions. Blocks provided quasi-replicates for 18K' and FAF conditions, with one exception. For conditions in the 18K' and FAF procedures with decimal values of 0.03, blocks were not considered replicates as they were provided different levels of the similarity factor. Two run-lists were constructed and assigned to subjects alternately to counterbalance any order effect for the first and last half of the testing runs. Subjects receiving run-list 1 performed testing runs in block A then testing runs in block B. Subjects receiving run-list 2 first performed testing runs in block B then testing runs in block A. The allocation of paths to runs was the same for blocks A and B.

Refresher runs were designed to expose subjects to each interruption and to relatively early and late interruption positions in each procedure. Within each of the A and B blocks, experimental conditions were assigned according to several design rules. Only one of any interruption type occurred in a run. Uninterrupted conditions were combined to provide one completely uninterrupted run *per* block. Additional uninterrupted conditions were positioned to provide at least one uninterrupted procedure every other run. Experimental conditions were assigned within runs to minimize interference between procedures; *i.e.,* a late condition was not followed by an early condition in the next interval. Visual interruptions were maximally spaced in blocks A and B. Finally paths were allocated to runs such that, with one exception, neither runway nor path configuration were the same for any consecutive pair of runs. The exception exists in both block A and B, for runs 23 and 24, and for runs 31 and 32. In this exception the path shape differs but the path is still oriented to runways 1 left and right.

Dependent Measures

Measures collected to assess these scenario features are described in addition to those measuring interruption management performance in general and in response to task factor manipulations.

Scenario Assessment Measures

Several measures were collected to test scenario assumptions. These measures described: (1) subject perceptions of FPM difficulty during the scenario, (2) the ability of subjects to perform the scenario's FPM requirements, (3) consistency of the designed procedures with subject task orders, (4) subject perceptions of coupling-strength and coupling type for the six coupling factor conditions.

Measuring Perceived FPM Difficulty

Subjects' perceptions of FPM difficulty were provided as Bedford scale ratings, integers from 1 to 10. These values were averaged for each subject over pre-defined flightpath regions (Figure 5.8). An overall perceived-FPM-difficulty score for each flightpath region was obtained by averaging over all subjects.

Measuring FPM Skills

FPM performance criteria were defined for altitude, speed, and lateral deviations when crossing a waypoint. These criteria required deviations of less than: 200 feet altitude, 10 knots of calibrated airspeed, 0.5 dot on the horizontal path deviation indicator scale (1875 feet). Three FPM performance criteria measures were constructed:

1) Altitude Deviation Criterion (ADC) ;

$$(5.2)\ \text{ACD} = \max\left[\ (\ |\ |\ \text{Altitude-Deviation}\ | - 200\ |\ ),\ 0\ \right]$$

2) Speed Deviation Criterion (SDC);

$$(5.3)\ \text{SDC} = \max\left[\ (\ |\ |\ \text{Speed-Deviation}\ | - 10\ |\ ),\ 0\ \right]$$

3) and Lateral Deviation Criterion (LDC);

$$(5.4)\ \text{LDC} = \max\left[\ (\ |\ |\ \text{Lateral-Deviation}\ | - 1875\ |\ ),\ 0\ \right].$$

In addition to these criteria measures, absolute altitude, speed, and lateral deviations were collected at each waypoint.

Measuring Perceived Coupling-Strength and Type

For each of the six experimental conditions for the coupling factor, subjects provided a rating of coupling-strength as an integer from one to five (see Appendix 5.17). Subjects also noted the form of coupling if they perceived coupling strength to be greater than a rating of three.

Interruption Management Measures

Dependent measures reflected three of the effects defined by the interruption management model, distraction, disturbance, and disruption. No measures were taken of effects on diversion. These measures are described below and summarized in Table 5.4.

Measuring Distraction

Distractibility of the ATC interruptions was measured by pilots' acknowledgment times to the interruptions. Interruption acknowledgment time was the elapsed time from initial announcement of the interruption to the event signaling subject's acknowledgment of the interruption's content. For aurally-presented interruptions, this event was the first open-microphone event following the interruption. For visually-presented interruptions, this event was the first response selection on the datalink touchscreen.

Measuring Disturbance

Interruption disturbance was associated with four dependent measures; interruption initiation time, interruption performance errors, procedure resumption time, and standardized resumptive FPM activity. Interruption initiation time was defined as elapsed time from the acknowledgment event to the first event required to perform the interrupting task. For all interrupting tasks, this first event required subjects to press a key on the FMS/CDU. Several forms of errors were defined to describe accuracy of interruption task performance. These interruption performance errors included; not acknowledging the interruption, beginning interruption task performance before acknowledging it, not executing the FMS/CDU revision, not returning to the Legs page on the FMS/CDU, returning to the Legs page before executing the revision, and selecting or entering an incorrect item or value. Resumption measures, procedure resumption time and resumptive FPM activity, were demarcated by two events; the last event required to perform the interrupting task, *i.e.*, returning to the Legs page on the FMS/CDU, and the next constituent event of the ongoing procedure. If the subject did not return to the Legs page or did not perform any procedural events after returning from the interruption, these resumptive measures were not defined. Resumptive FPM activity measured the number of sidestick controller inputs in this interval standardized by the length of this interval. Increased resumptive FPM was interpreted as a fidget response indicating interruption disturbance.

Measuring Disruption

Three measures evaluated the degree to which an interruption disrupted an ongoing procedure; procedure performance errors, ensemble performance time, and ensemble FPM activity. Procedure performance errors included; task omissions, task order errors, and performance of extraneous tasks. These error forms combined to form a single count of procedure performance accuracy. The ensemble interval started with either the first event required for the procedure or the first event required for the interruption, and ended with either the last event associated with the procedure, or the last event associated with the interrupting task, which ever occurred first. The ensemble interval for interruptions before the subject started the first procedural task conditions, those with a condition decimal value of 0.02, began with the first event required to perform the interruption and ended with the last event required to perform the procedure. The ensemble interval for interruptions after the subject finished the last procedural task, those with a condition decimal value of 0.10, began with the first event required to perform the procedure and ended with the last event required to perform the interruption. For all other interruption conditions, the ensemble interval began with the first event required to perform the procedure and ended with the latter of the last events required to perform the interruption or the procedure. If either the starting or terminating conditions were missing, the ensemble performance measures were declared missing. Ensemble performance time measured the performance time of the integrated interruption and procedure. Standardized ensemble FPM activity counted the number of sidestick controller events contained in the ensemble interval.

To ascertain the temporal effect of interruptions on performing procedural tasks, ensemble performance times, for which interruptions occurred within a procedure, were compared to constructed “composite” times. Composite times were constructed by adding the average of uninterrupted procedure times and interruption performance times for all possible subject, procedure, and interrupting task triplets to eliminate effects of these variables. Interruption conditions in which the interruption occurred before the subject started procedure performance, those with condition decimal values of 0.02, were used to construct composite times.

Table 5.4. Interruption Management Dependent Measures

Interruption Effect	Dependent Measure	Relationship
Distractibility	Interruption Acknowledgment time	inverse
Disturbance	Interruption Initiation Time	direct
	Interruption Performance Errors	direct
	Procedure Resumption Time	direct
	Resumptive FPM Activity	direct
Disruption	Procedure Performance Errors	direct
	Ensemble Performance Time	direct
	Ensemble FPM Activity	direct

Analyses

Analyses are presented for testing design and training assumptions, characterizing the effects of interruptions generally, and evaluating the effects of task factor manipulations on interruption management performance measures. Analyses performed for other purposes are reinterpreted for significant individual difference effects.

Validating Scenario Design Assumptions

Preliminary analyses confirm design and training assumptions. Subjects are assumed to experience the FPM difficulty profile as designed. Subjects are assumed to be adequately trained on FPM skills. Procedures are assumed to be consistent with subject task orderings. Subject coupling-strength ratings and type assignments are assumed to be consistent with designed levels.

Flightpath Management Workload Profile

A full factorial analysis of variance assessed whether the designed difficulty ratings significantly account for variability in averaged Bedford ratings for all subjects, that is not accounted for by subject or run variability or interaction terms. Run number and design-level were fixed factors in this analysis[7]. Scheffé *post-hoc* tests on Bedford rating means by design-level were examined to determine if perceived FPM difficulty increased significantly for each step increase in design-level. In particular, a contrast on means examines whether subjects perceived non-procedural intervals significantly less demanding than non-procedural regions. The same analyses were conducted on each subject's data

---

[7] The parametric analysis of variance provides a conservative analysis of this rating data and allows for a convenient multi-factor partitioning of effects.

individually to determine the degree to which individual subjects experienced the FPM difficulty levels as intended.

Assessing FPM Training

Subject flightpath management skills were evaluated for evidence that they had reached FPM criterion before adding procedure and interruption performance to the scenario, retained this skill in whole-scenario runs, and remained at a fairly constant level of FPM over testing runs. Each subject's ADC, SDC, and LDC values were calculated for waypoints in runs 4, 5, and 6 and were analyzed with a two-sided *t*-test against a hypothesized mean of zero. This analysis was repeated on deviations during runs 15 and 16 to determine if subject FPM skills were within criterion prior to testing runs. Analyses of variance were conducted on absolute speed, altitude, and lateral deviations on runs 15 and 16 to determine if subjects significantly differed in FPM skill prior to testing runs. The stability of subject FPM skills were evaluated by regressing run number on the absolute value of altitude, speed, and lateral deviations separately. Two-sided *t*-tests on the slopes of these regressions tested whether these slopes statistically differed from zero to indicate stability over runs.

Procedure Design

The designed task order resulted from an ordinal enumeration of procedural tasks as they existed in the TOD, 18K', and FAF procedures and concatenating these procedures in order of their performance in the experimental scenario. This designed task order was compared to each subject's task order using Kendell's *tau* statistic. In addition, the task orders provided by subjects were analyzed for consistency of opinion using Kendell's Coefficient of Concordance, *W*.

Coupling-strength Assessment

Coupling-strength ratings were analyzed by a mixed-model, full factorial analysis of variance of the form; 14 (Subjects) X 2 (Procedure Legs: 18K', FAF) X 3 (Hypothesized Coupling-Strength: Low, Medium, High); to determine if subject coupling-strength ratings were consistent with hypothesized levels. Procedure Leg and Coupling-type were considered fixed, within-subject factors. Coupling-type assignments were analyzed across subjects to determine if conditions were perceived as the intended type. A Friedman non-parametric *F*-test was conducted on possible type-assignments for each coupling condition to determine if subjects identified coupling-types consistently with assumed types[8].

General Interruption Management Effects

Analyses were designed to describe characteristics of interrupting task performance and to ascertain the effects of interruption on performing procedures on the flightdeck. Constraints

---

[8] The non-parametric Friedman test was used to analyze coupling rating scores for a more sensitive univariate analysis for each coupling condition type.

on simulation availability and experimental run time necessitated far fewer uninterrupted experimental conditions than interrupted experimental conditions, introducing a potential for outliers in the uninterrupted condition data to bias results with more leverage than outliers in the interrupted condition data. The data partitioning scheme and statistical models for these analyses are provided in Appendix 5.23. Appendix 5.24 uses the same data partitioning scheme to indicate the allocation of path types to experimental factors. While path types are not counterbalanced for each analysis, the orientation and configuration of paths is irrelevant to performance within each of the straight procedural legs.

General Effects on Interrupting Tasks

Performance of realistic interrupting tasks is characterized by mean tables of acknowledgment time, initiation time, and interruption performance errors over all interruption conditions and subjects. Analyses of variance on these measures indicated the degree to which different experimental interruption conditions and subject variability are significant[9].

General Effects on Procedure Performance

Three analyses evaluated the disruptive effects of interruptions on ongoing task procedure. A two-sided, paired *t*-test evaluated whether the difference ensemble times and composite times significantly differed. Not all conditions were included in this analysis. Interruption conditions in which the interruption precedes procedure performance, for which decimal condition values are 0.02, were not included in the time comparisons because interruption performance times for these conditions were used to construct composite times. Interrupting tasks that were never performed external to the procedures were not included because composite times could not be constructed for them. Omitted conditions included those requiring subjects to change speed or altitude restrictions (12.03, 22.03, 13.03, 23.03) or were visually-presented (11.06, 21.06, 11.08, 21.08).

The effects of interruption on procedure performance errors[10], ensemble performance time and standardized ensemble FPM activity were each considered in a mixed-model, full factorial analysis of variance; 14 (Subjects) X 3 (Procedure Leg: TOD, 18K', FAF) X 2 (Condition: Interrupted, Uninterrupted). The Procedure Leg and Condition factors were considered fixed, within-subject variables. All experimental conditions were included in these analyses. In particular, general analyses of interruption effects include conditions 11.02, 21.02, 11.03, and 21.03 to equalize the number and diversity of interrupt conditions in each procedural leg. These conditions, however, were not included in analyses of specific task factors. Table 5.5 displays the levels and experimental conditions for these analyses.

---

[9] Analyses of variance were conducted for all analyses of interruption error rate data because condition cells were insufficiently populated to calculate X^2 statistics. The analysis of variance for error rate data provides a conservative estimate of significance.

[10] Analyses of variance were conducted for all analyses of procedure error rate data because condition cells were insufficiently populated to calculate X^2 statistics. The analysis of variance for error rate data provides a conservative estimate of significance.

Table 5.5. Levels and Experimental Conditions for Testing Effects of Interruptions.

Procedure Leg	Interrupted Conditions	Uninterrupted Conditions
TOD	11.02, 21.02, 11.03, 21.03, 11.05, 21.05, 11.06, 21.06, 11.08, 21.08, 11.09, 21.09	11.11, 21.11, 11.12, 21.12
18K'	12.02, 12.03, 12.05, 12.06, 12.07, 12.10 22.02, 22.03, 22.05, 22.06, 22.07, 22.10	12.11, 22.11, 12.12, 22.12
FAF	13.02, 13.03, 13.05, 13.06, 13.07, 13.10 23.02, 23.03, 23.05, 23.06, 23.07, 23.10	13.11, 23.11, 13.12, 23.12

Effects of Task Factors on Interruption Management

Constraints on simulator availability and experimental run time required an efficient data collection scheme. For this reason, some experimental conditions operationalize more than one level of the set of hypothesized task factors. The contribution of experimental conditions to each hypothesized factor is presented in Table 5.6. While the data collection scheme is essentially nested in some places, the effects of task factors on interruption management performance were ascertained using separate analyses of variance for each factor. Using separate analyses is acceptable given that the nested factors are fixed. In addition, by analyzing experimental conditions in separate analyses, potential intercorrelations due to run construction sequences are minimized. This section presents the experimental designs and experimental conditions used in each analysis. Statistical models for these analyses are provided in Appendix 5.23.

Table 5.6. Task Factor Experimental Conditions

Experimental Condition	Modality (Task/Interrupt)	Goal-Level of Interruption Position	Coupling -Strength	Similarity	Environmental Stress
11.05, 21.05	Aural/Aural				
11.06, 21.06	Aural/Visual				
11.08, 21.08	Visual/Visual				
11.09, 21.09	Visual/Aural				
12.02, 22.02		Outside Procedure			Low
12.03		Within Task		Similar	
22.03		Within Task		Dissimilar	
12.05, 22.05		Between Tasks	Low		
12.06, 22.06		Between Tasks	High		
12.07, 22.07		Between Tasks	Medium		
12.10, 22.10		Outside Procedure			
13.02, 23.02		Outside Procedure			High
13.03		Within Task		Dissimilar	
23.03		Within Task		Similar	
13.05, 23.05		Between Tasks	Low		
13.06, 23.06		Between Tasks	High		
13.07, 23.07		Between Tasks	Medium		
13.10, 23.10		Outside Procedure			

Effects of Modality on Interruption Management

Effects of task and interruption modality on interruption management dependent measures were considered in mixed-model, partial factorial analyses of variance of the form; 14 (Subjects) x 2 (Task Modality: Aural, Visual) x 2 (Interruption Modality: Aural, Visual) x 2 (Replication). Interaction terms were included for: Subjects x Task Modality, Subjects x Interruption Modality, Task Modality x Interruption Modality, and Subjects x Task Modality x Interruption Modality. Task and Interruption Modality factors were fixed, within-subject variables with two datum *per* subject, *per* condition. Scheffé *post-hoc* tests were conducted on significant task modality and interruption modality main effects. In addition, a planned contrast of means was conducted to compare same-modality (both task and interruption auditory or both visual) with cross-modality (task and interruption modalities different) conditions. Table 5.7 displays the levels and experimental conditions used to test the effects of modality.

Table 5.7. Levels and Experimental Conditions Testing Modality Effects.

	Task Modality	
Interruption Modality	Visual	Aural
Visual	11.08, 21.08	11.06, 21.06
Aural	11.09, 21.09	11.05, 21.05

Effects of Goal-Level on Interruption Management

Effects of interruption position goal-level on interruption management dependent measures were considered in mixed-model, partial factorial analyses of variance of the form; 14 (Subjects) X 2 (Procedural Leg: 18K', FAF) X 3 (Goal-Level: Outside Procedure, Between Tasks, Within Task). Interaction terms were included for: Subjects X Procedural Leg, Subjects X Goal-Level, Procedural Leg X Goal-Level, and Subjects X Procedural Leg X Goal-Level. Procedural Leg and Goal-Level factors were fixed, within-subject variables. Scheffé *post-hoc* tests were conducted on significant goal-level main effects. Table 5.8 displays the levels and experimental conditions used to test the effects of the goal-level at which an interruption occurs.

Constraints on simulation availability and experimental run time necessitated using conditions for testing coupling-strength and similarity as the level 2 and 3 conditions, respectively, for testing effects of goal-level. Because the coupling factor required three distinct conditions, each with two replications per subject, the between-tasks goal-level condition includes a more data than the other conditions of the goal-level. Because the similarity factor does not contain a replication, the within-task goal-level condition has fewer data points than the between task or external-to-procedure conditions. The unequal condition sample sizes for these three conditions presents the opportunity for outliers in lesser-represented experimental conditions to disproportionally bias results.

Table 5.8. Levels and Experimental Conditions Testing Goal-Level Effects.

	Procedural Leg	
Goal-Level	18K’	FAF
1- External to Procedure	12.02, 22.02 12.10, 22.10	13.02, 23.02 13.10, 23.10
2- Between Procedural Tasks	12.05, 22.05, 12.06, 22.06, 12.07, 22.07	13.05, 23.05, 13.06, 23.06, 13.07, 23.07
3- Within a Procedural Task	12.03, 22.03	13.03, 23.03

Effects of Coupling on Interruption Management

Effects of the cohesion between interrupted adjacent tasks on interruption management dependent measures were considered in mixed-model partial factorial analyses of variance of the form; 14 (Subjects) X 2 (Procedural Leg: 18K’, FAF) X 3 (Coupling-Strength: Low, Medium, High). Interaction terms were included for: Subjects X Procedural Leg, Subjects X Coupling-Strength, Procedural Leg X Coupling-Strength, and Subjects X Procedural Leg X Coupling-Strength. Procedural Leg and Coupling-Strength were fixed, within-subject variables with two datum *per* subject, *per* condition. Scheffé *post-hoc* tests were conducted on significant coupling-Strength main effects. Table 5.9 displays the levels and experimental conditions used to test the effects of the goal-level at which an interruption occurs.

Table 5.9. Levels and Experimental Conditions Testing Coupling Effects.

		Coupling-Strength	
Procedural Leg	Low	Medium	High
18K’	12.05, 22.05	12.07, 22.07	12.06, 22.06
FAF	13.05, 23.05	13.07, 23.07	13.06, 23.06

Effects of Similarity on Interruption Management

Interruption management dependent measures were considered in mixed-model, partial factorial analyses of variance of the form; 14 (Subjects) X 2 (Procedural Leg: 18K’, FAF) X

2 (Similarity: Similar, Dissimilar), to test the effects of interrupting procedural tasks with semantically similar vs. dissimilar tasks. Interaction terms were included for: Subjects X Similarity, Subjects X Procedural Leg, and Similarity X Procedural Leg. Procedural Leg and Similarity factors were fixed, within-subject variables with no replication. Scheffé *post-hoc* tests were conducted on significant similarity main effects. Table 5.10 displays the levels and experimental conditions used to test the effects of interrupted task/interruption similarity.

Table 5.10. Levels and Experimental Conditions Testing Similarity Effects.

	Semantic Similarity	
Procedural Leg	Similar	Dissimilar
18K’	12.03	22.03
FAF	23.03	13.03

Effects of Environmental Stress on Interruption Management

Interruption management dependent measures were considered in mixed-model, partial factorial analyses of variance of the form; 14 (Subjects) X 2 (Procedural Leg: 18K’, FAF) X 2 (Replication), to test the effects of environmental stress on interruption management performance. The Subject X Procedural Leg interaction term was also included in the model. Procedural Leg was a fixed, within-subject variable. Only 18K’ and FAF interruption conditions occurring before procedure performance; *i.e.*, interruption conditions whose decimal values are 0.02, were included in these analyses. Analyses of other factors including the procedure leg factor were assessed for significant interactions of procedure leg and other task factors.

Observations on Individual Differences

The significance of individual differences in interruption management was investigated generally by analyzing interruption management dependent measures on all testing conditions in two-way analyses of variance of main effects of the form; 14 (Subjects) X 18 (Interrupted Experimental Conditions). The interaction term served as the residual and error estimate for both factors. In addition to this overview, previous analyses were scrutinized for evidence of significant interactions of subject variability with task factor manipulations. Finally, analyses were reviewed for task factor effects that did not include significant subject differences to identify particularly robust task factor effects on interruption management measures.

6. Experimental Results

Analyses validate assumptions, investigate general effects of interruptions, evaluate the effects of five specific task factors on interruption management performance measures, explore individual differences in interruption management, and finally evaluate the relative utility of the interruption management measures for distinguishing among condition levels.

Validating Assumptions

The experimental scenario was specifically designed to operationalize experimental conditions in a relatively realistic operational context and provide experimental controls. Analyses assessed whether the following design assumptions were met: (1) Subjects experienced FPM workload as intended by the designed difficulty profile. (2) Subjects were adequately trained on FPM skills for the profile both alone and in whole-scenario runs, and FPM skills were stable over testing runs. (3) Procedures presented tasks in an order consistent with the order in which subjects would arrange these tasks. (4) The pairs of adjacent tasks used to operationalize levels of the Coupling-Strength factor reflected distinct levels of coupling-strength as perceived by subjects.

Flightpath Management Workload Profile

An analysis of variance on Bedford scale ratings assessed whether subject perceptions of FPM difficulty throughout the scenario were consistent with the designed difficulty levels over regions in the flightpath. Design-level ratings accounted for a highly significant portion of variance in subjective assessments, $F(3,39) = 90.985$, $p = 0.0001$ (Appendix 6.1), and average subjectively-assessed difficulty generally increased with design-level difficulty. *Post hoc* Scheffé tests demonstrate that design-levels of 3 were rated as significantly more difficult than design-levels of 0 (the design-level for procedural intervals) and 1, all $p <= 0.0001$, but was not rated as significantly more difficult than design-level 2, $p = 0.3353$. Design-level accounted for differences in subjective ratings over the flightpath for each subject individually, $p <= 0.0024$, and subjective assessment means generally increased with design-level (Appendix 6.2).

Flightpath Management Skills

Flightpath management deviations did not exhibit asymptotic relationships with training run number, perhaps due to the step-wise introduction of scenario elements (Appendix 6.3). Rather than analyzing FPM deviation trends over runs, subject FPM skills were assessed against pre-defined criterion at two critical junctures; prior to procedure and whole-scenario training, and on the two runs just prior to testing. Flightpath management deviations for each subject on testing runs were also analyzed for stability.

FPM Training Criterion Assessment

Subject FPM performance during runs 4, 5, and 6; that is, prior to procedure or whole-scenario training, adhered to pre-defined altitude, speed, and lateral FPM performance criteria; ADC, SDC, and LDC, respectively. None of the *t*-tests performed on these three performance criteria for each subject indicated that these measures significantly differed from zero, all $p > 0.1097$ (Appendix 6.4)[11].

Subject FPM Skills Prior to Testing

Subjects still met these performance criteria when they also performed procedural and interrupting tasks on the two runs preceding testing runs (runs 15 and 16), all $p > 0.1176$ (Appendix 6.5)[12] . Runs 15 and 16 were also evaluated to determine if subject FPM skill levels were equivalent prior to test data collection. Analyses of variance were performed on the absolute values of altitude, speed, and lateral deviations for these runs. Results indicated that subjects did not statistically differ in their ability to control speed deviations, $F(13,138) = 1.273$, $p = 0.2363$ (Appendix 6.6), or lateral deviations, $F(13,138) = 1.237$, $p = 0.2598$ (Appendix 6.7) on the two runs just prior to testing. Subjects did statistically differ in their ability to control altitude deviations, $F(13,138) = 2.028$, $p = 0.0227$ (Appendix 6.8), although Scheffé *post hoc* tests, $\alpha = 0.05$, did not indicate any significant differences among subjects.

Stability of FPM Skills in Testing Runs

Subject FPM testing data were analyzed to ensure that FPM skills remained stable over the course of the testing runs. Regressions of altitude, speed, and lateral deviations over testing runs showed, with a few exceptions, slopes not significantly different from zero ($\alpha = 0.05$), low R^2 values, all $R^2 < 0.05$, and relatively few datum per subject outside criteria (Appendix 6.9). Exceptions to this general observation are detailed below.

Subject 13 demonstrated a very slight decrease, slope = -0.038, $p = 0.0218$, in absolute speed deviation over the testing runs. This appears to be due to two extreme values during run 17, and one extreme value during run 20. Subject 13 performed with less than 5 KIAS of speed deviation for all testing runs. Subject 14 demonstrated a very slight increase, slope = 0.301, $p = 0.0256$, in absolute speed deviation over the testing runs. Subject 14's performance on the last run included two datum of speed deviation excursions greater than 30 KIAS which likely caused the apparent inclination of speed deviations over testing runs. The regressions of absolute altitude deviation on run number for subjects 6, 8, and 12, demonstrated slopes significantly different from zero, all $p < 0.05$. Subject 6's altitude-deviations appear to diminish over run number, slope = - 1.514, $p = 0.0238$, however this is negative slope appears largely influenced by an extreme value during run 17. Subject 8's altitude deviation absolute values increased slightly over testing runs, slope = 1.071, $p = 0.0438$. Subject 12's absolute altitude deviations decreased slightly, slope = -1.329, $p = 0.0179$, over the course of the testing runs. Inspection of subjects 8

---

[11] In several cases, a *t* statistic could not be calculated because all values of the criteria measure were zero.

[12] In several cases, a *t* statistic could not be calculated because all values of the criteria measure were zero.

and 12's altitude deviation data did not indicate any particular extreme values to which significant slopes might be attributable. Of the three FPM dimensions, subjects most frequently committed speed deviations outside the criterion (10 KIAS) during the testing runs. Lateral deviations outside criterion (1875') were least frequent. No subject produced more than 5 excursions on any one parameter over all waypoints of the test runs; a total of 112 waypoint crossings *per* subject.

Procedure Design

Procedures were constructed to be familiar in task content and flow to the operational experience of subjects while providing task contexts necessary for experimental control and interruption conditions. Subject orderings of constituent tasks were compared to task orderings in the designed procedures to assess this degree of familiarity. While only the ordering from subject 15 was statistically similar to the procedural task orders, Kendell's *tau* = 0.339, $p = 0.0131$, orders defined by subjects were not statistically different, Kendell's $W(13) = 50.50$, $p < 0.0005$ (Appendix 6.10).

Perceived Coupling-Strength and Type Assignments

To ensure correct operationalization of coupling-strength levels, subjects were asked to rate the coupling-strength of, and the type of, each pair of adjacent tasks in the procedures. Subjects rated the coupling-strength of the three conditions differently, $F(2,26)= 98.581$, $p= 0.0001$, and rated the low-coupling pairs (with assumed type of "uncoupled") lower than that of the moderately-coupled pairs (with assumed type of "physically-coupled"), and the coupling-strength of the moderately-coupled pairs lower than that of the highly-coupled pairs (with assumed type of "functionally-coupled"), all *post hoc* tests, $p < 0.0024$ (Appendix 6.11). This analysis indicated no significant interaction of coupling-strength/type and procedure, F(2,26) = 0.223, $p = 0.8014$.

Subject type-assignments for each pair appeared consistent with assumed type assignments and statistically salient among alternative types, maximum $p < 0.01$ over all X^2 tests, with one exception. Type-assignments for the physically-coupled experimental condition in the FAF procedure were not statistically different, $X^2(4) = 3.923$, $p = 0.4165$, tied-$p = 0.1278$ (Appendix 6.12). While most subjects labeled this condition as physically-coupled, an approximately equal number of subjects considered this condition functionally-coupled as did consider it uncoupled.

General Effects of Interruptions on the Flightdeck

Results characterize the general effects of interruptions to this simulated flightdeck from two perspectives. First, results describe pilot responsiveness to acknowledging and initiating interrupting ATC calls, and error rates in performing these interrupting tasks. Second, results compare pilots' performance of interrupted procedures with performance of uninterrupted procedures in terms of time to perform procedural tasks, procedure performance errors, and the rate of FPM events in a procedural interval. These analyses include those data trials in which subjects committed procedure performance errors. Error data were not extracted for analyses of reaction time and FPM activity measures for several reasons; 1) for most conditions, elimination of error data would result in an approximately 40% loss of data, dramatically reducing the power of the analyses (Appendix 6.13). 2) errors occur disproportionately over conditions, and therefore the randomness of reaction time and FPM measures would be destroyed and sample sizes made, in some cases even more, unequal. 3) error-free performance does not represent the actual time delays and FPM activity incurred by the various conditions, whether they are attributed to the effect of an experimental or secondarily, as an effect of errors induced by these conditions. As an exercise, all planned analyses on the full data set were compared to the same analyses on error-free data. Most significant effects in the full data set retained significance in the reduced data set. Approximately a quarter of the originally significant results were not significant in the reduced set, due to extreme loss of power. One non-significant result in the full data set became significant in the reduced data set. For all these conditions, the relationship among means in the original analyses was preserved in the error-free analyses.[13] The results of the presented analyses, then, characterize, generally, the natural effect of interruptions on a simulated commercial flightdeck, inclusive of secondary effects due to errors induced by these interruptions.

Performing Interrupting Tasks on the Flightdeck

The ability of pilots to perform ATC initiated tasks that interrupt other ongoing flightdeck tasks was characterized by response times associated with acknowledging and initiating these tasks, and interruption performance errors (Appendix 6.14). Measures of central tendency indicate that over 7 seconds elapsed, on average, before pilots acknowledge interrupting ATC calls, and that over 5 seconds elapsed before pilots began performing these interrupting tasks. Although performance was usually error-free, mean error rate over all interruption conditions was 0.171, or one error in approximately every 6 ATC-initiated interrupting tasks.

Analyses of variance were conducted on these measures to indicate the significance of different experimental conditions and subject variability on these effects. Results indicated that for both interruption acknowledgment and initiation time, both experimental condition and subjects were highly significant, $p = 0.0001$ (Appendix 6.15, 6.16). Results of analysis of variance also indicated that interruption performance errors did not significantly differ by experimental condition, $F(17, 407) = 1.386$, $p = 0.1388$, but did significantly differ by subject, $F(13, 407) = 1.650$, $p = 0.0694$ (Appendix 6.17). Analyses of factors suspected to influence interrupted task management determine the extent to which these factors explain why pilot performance is significantly different over experimental conditions. The significant effects due to subject variability on these dependent measures will be described in conjunction with other results in section 6.4.2.

---

[13] Error-free analyses are not presented in this dissertation.

Effects of Interruptions on Procedure Performance

The presence of interruptions in procedural intervals produced on average, a statistically significant 9.6% increase in FPM inputs *per* second in procedural intervals, $F(1,13) = 4.986$, $p = 0.0438$ (Appendix 6.18), and a 53% increase in procedure performance errors, $F(1,13) = 25.809$, $p = 0.0002$ (Appendix 6.19). The frequency of some omitted tasks appeared to be exacerbated if an interruption occurred previously in the procedure (Appendix 6.19). Composite times (times for completing uninterrupted procedures plus times for completing un-embedded interrupting tasks) significantly differed from ensemble times (interrupted procedure performance times) at a modest level, $t\ (242) = -1.672$, $p = 0.0958$ (Appendix 6.20). On average, composite times exceeded ensemble times by 1.63 seconds. This relationship was also evident in a similar analysis of only error-free trials, where composite times exceeded ensemble times by even a larger amount, on average 2.034 seconds, $t\ (132) = -1.665$, $p = 0.0984$ (Appendix 6.20).

Task Factors Affecting Flightdeck Interruption Management

Analyses of variance on distraction, disturbance, and disruption performance measures tested the effects of modality, goal-level, coupling-strength, similarity, and environmental stress on interruption management over interrupted experimental conditions. By analyzing these hypothesized factors separately, it is possible that, for the analysis of one factor, the residual error term may be inflated by the presence of another significant factor. Therefore, the separate analyses for these factors performed here represent a conservative approach to assessing their significance. As for the previous analyses, data in which subjects performed procedural errors were included in the analyses, as were, for these analyses, data in which subjects performed interruption performance errors. These data were included for the same reasons as stated above; statistical power (Appendix 6.13), to preserve the random distribution of the measure and, where possible, roughly equivalent sample sizes, and to represent realistic behavior.

Effects of Modality on Interruption Management

Analyses determined if modality characteristics influenced performance as predicted; specifically, if interruptions to auditory tasks were less likely to distract than interruptions to visual tasks, if auditory interruptions were more distracting than visual interruptions, and if cross-modality conditions were more distracting and less disturbing and disruptive than same-modality conditions.

Distracting Effects of Modality

Analysis of variance on interruption acknowledgment time indicated that the distraction produced by an interruption was significantly related only to the interrupted task modality. Interruptions to auditory tasks were acknowledged, on average, approximately 4 seconds slower than interruptions to visual tasks, $F(1,13) = 4.303$, $p = 0.0585$ (Appendix 6.21). Subject interactions with task modality, $F(13,55) = 5.889$, $p = 0.0001$, and interruption modality, $F(13, 55) = 6.455$, $p = 0.0001$, were highly significant. Individual differences of acknowledgment time in response to task and interrupt modalities may have decreased the significance of task and interrupt modality interaction effects.

Disturbing Effects of Modality

Analyses of variance were conducted on interrupt initiation time, interruption performance errors, procedure resumption time, and resumptive FPM activity to ascertain disturbing effects attributable to modality characteristics.

The interaction of task and interruption modalities significantly influenced initiation time, $F(1,13) = 6.976$, $p = 0.0204$ (Appendix 6.22). Interruption initiation times to cross-modality conditions were significantly slower than to same-modality conditions, $F(1,13) = 7.402$, $p = 0.0175$. Significant main effects of interruption modality indicated that subjects began performance on interrupting tasks more slowly when they were presented visually, $F(1,13) = 3.159$, $p = 0.0989$, and when the interruption occurred to an auditory task, $F(1,13) = 10.298$, $p = 0.0068$. However, inspection of the interaction and *post hoc* Scheffé tests on interaction means indicated that interruption modality only differentially affected interruption initiation time for auditory interrupted tasks. In particular, subjects delayed performing visual interruptions to auditory tasks almost twice as long, on average, than any other interaction conditions.

The interaction between task modality and interruption modality also affected tendency to err in performing the interrupting task, $F(1,13) = 5.2$, $p = 0.0401$ (Appendix 6.23). This interaction was explained by a contrast of cross-modality conditions to same-modality conditions. Subjects made more interruption performance errors in cross-modality conditions than in same-modality conditions. Inspection of interaction means indicated that while the interaction effect is obvious, interruption errors were substantially higher when visual tasks were interrupted aurally than for any other conditions. Neither task modality, nor interruption modality, nor their interaction influenced either procedure resumption time (Appendix 6.24) or resumptive FPM activity (Appendix 6.25).

Disruptive Effects of Modality

Analyses of variance were conducted on ensemble performance time, ensemble FPM activity, and procedure performance errors to evaluate disruptive influences attributable to task and interruption modalities.

Auditory interruptions extended ensemble performance time more than visual interruptions, $F(1,13) = 10.674$, $p = 0.0061$ (Appendix 6.26). The interaction of task and interruption modalities significantly affected procedural errors, $F(1,13) = 9.1$, $p = 0.0099$ (Appendix 6.27). A contrast of same-modality and cross-modality conditions indicated that same-modality conditions induced significantly more procedure performance errors, $F(1,13) = 9.1$, $p = 0.0099$. *Post hoc* Scheffé tests indicated that only the auditory task/auditory interruption condition significantly differed from the other three conditions. The extreme affect of this experimental condition on procedure performance error production created main effects of task modality, $F(1,13) = 16.278$, $p = 0.0014$, and interruption modality, *F*(1,13) = 4.5, *p* = 0.0537. Neither task modality, nor interruption modality, nor their interaction significantly influenced ensemble FPM activity (Appendix 6.28).

Effects of Goal-Level on Interruption Management

Interruptions external to procedure performance were hypothesized to be less destructive than interruptions within a procedure. Similarly, interruptions between procedural tasks were hypothesized to be more distracting and less disturbing and disruptive than interruptions within a procedural task. The following analyses determined if the procedural goal-level at which an interruption was embedded influenced distraction to the interruption or its disturbing or disruptive effects.

Distracting Effects of Goal-level

Analysis of variance on acknowledgment time indicated that the goal-level of an interruption did not significantly influence subject acknowledgment times, *F*(2,26) = 1.910, *p* = 0.1684 (Appendix 6.29). Goal-level significantly interacted with subject variability *F*(2,26) = 6.663, *p* = 0.0001.

Disturbing Effects of Goal-level

Analysis of variance indicated that the goal-level of an interruption significantly affected interruption initiation time, *F*(2,26) = 16.192, *p* = 0.0001 (Appendix 6.30). *Post hoc* Scheffé tests indicated that interruptions occurring within a procedural task, *i.e.*, at the activity level, were initiated significantly more slowly than interruptions either between procedural tasks, *p* = 0.0001, or external to procedure performance, *p* = 0.0012. Initiation times for interruptions between procedure tasks were not significantly different from performance on interruptions external to the procedure, *p* = 0.3606. Inspection of initiation time residuals by the independent conditions for the goal-level factor revealed no obvious distinctions to indicate that differences between within goal-level factors caused the overall effect. Other measures of disturbance; interruption performance errors, procedure resumption time, and resumptive FPM activity, were not differentially affected by the goal-level at which interruptions were introduced (Appendix 6.31, 6.31, 6.32, respectively). Inspection of means for these measures by goal-level revealed no discernible trends.

Disruptive Effects of Goal-level

Subject ensemble performance times, *F*(2,26) = 0.303, *p* = 0.7417 (Appendix 6.34); ensemble FPM activity, *F*(2,26) = 1.724, *p* = 0.1981 (Appendix 6.35); and procedure performance errors, *F*(2,26) = 0.981, *p* = 0.3885 were not differentially affected by interruptions at different procedural goal-levels. Inspection of means by goal-level did not suggest trends in these measures.

Effects of Coupling-Strength on Interruption Management

The following analyses determine if the distraction, disturbance, and disruption produced by interruptions between two sequential procedural tasks was directly related to the perceived coupling-strength of those two tasks.

Distracting Effects of Coupling

An analysis of variance on interruption acknowledgment times indicated a significant effect of coupling on subject acknowledgment times, $F(2,26) = 6.324$, $p = 0.0058$ (Appendix 6.37). *Post hoc* Scheffé tests revealed that subjects were less likely to be distracted by an interruption between tasks of medium coupling-strength (physically-coupled tasks) than between either tasks of low coupling-strength (uncoupled tasks), $p = 0.0249$, or high coupling-strength (functionally-coupled tasks), $p = 0.0079$. Acknowledgment times for highly-coupled and uncoupled tasks did not significantly differ, $p = 0.8879$.

Disturbing Effects of Coupling

Analyses of variance were conducted on interruption initiation times, interruption performance errors, procedure resumption time, and resumptive FPM activity to determine if perceived coupling-strength of adjacent procedural tasks predicted interruption disturbance.

The effect of coupling was significant on all measures of disturbance. Interruption initiation times were significantly affected by coupling-strength level, $F(2,26) = 8.225$, $p = 0.0017$ (Appendix 6.38). *Post hoc* Scheffé tests demonstrated that interruptions between moderately-coupled tasks were acknowledged significantly more slowly than interruptions occurring between either uncoupled tasks, $p = 0.0032$, or highly-coupled tasks, $p = 0.0090$.

Procedure resumption times were significantly affected by coupling-strength level, $F(2,26) = 10.537$, $p = 0.0004$ (Appendix 6.39). *Post hoc* Scheffé tests showed that a procedure was resumed significantly faster after interruptions between highly-coupled tasks than after interruptions between either moderately-coupled tasks, $p = 0.0003$, or uncoupled tasks, $p = 0.0282$.

Similarly, resumptive FPM activity was moderately affected by coupling conditions, $F(2,26) = 2.822$, $p = 0.0778$ (Appendix 6.40). Commensurate with the effect on procedure resumption time, *post hoc* Scheffé tests showed less resumptive FPM inputs *per* second after an interruption between highly-coupled tasks than after an interruption between uncoupled tasks, $p = 0.0652$. The number of resumptive FPM inputs *per* second after interruptions between moderately-coupled procedural tasks was less than that for interruptions between uncoupled tasks and more than that for interruptions between highly-coupled tasks, although *post hoc* Scheffé tests did not find these differences significant.

The effect of coupling on interruption performance errors was significant, but revealed a different relationship than that exhibited by other disturbance measures, $F(2,26) = 3.602$, $p = 0.0416$ (Appendix 6.41). *Post hoc* Scheffé tests marginally indicated that interruptions between highly-coupled tasks produced more interruption performance errors than interruptions between moderately-coupled tasks, $p = 0.0589$.

Disruptive Effects of Coupling

Coupling-strength significantly explained differences in procedure performance errors, $F(2,26) = 6.966$, $p = 0.0038$ (Appendix 6.42). *Post hoc* Scheffé tests revealed that, subjects erred significantly more in procedure performance when interrupted between uncoupled tasks than when interrupted between either moderately-coupled, $p = 0.0056$, or highly-coupled tasks, $p = 0.0677$. Perceived coupling-strength of cleaved tasks did not differently-disrupt ensemble performance times, $F(2,26) = 0.1510$, $p = 0.8608$ (Appendix 6.43), or influence ensemble FPM activity, $F(2,26) = 0.2440$, $p = 0.7851$ (Appendix 6.44). Inspection of coupling level means revealed no latent trends in ensemble performance times or FPM activity.

Effects of Similarity on Interruption Management

The following analyses determine if interruptions that are semantically similar to the interrupted task are more distracting and less disturbing and disruptive than dissimilar interruptions.

Distracting Effects of Similarity

Interruption acknowledgment times were not differentially influenced by similar and dissimilar conditions, $F(1,13) = 0.0030$, $p = 0.9576$ (Appendix 6.45). Inspection of similarity and interaction means did not reveal any latent relationship between task/interrupt similarity manipulations and distraction induced by the interruptions.

Disturbing Effects of Similarity

Analyses of variance on interruption initiation times, interruption performance errors, procedure resumption time, and resumptive FPM activity failed to identify any disturbance effects attributable to similarity conditions based on interruption initiation times, $F(1,13) = 0.0002$, $p = 0.9885$ (Appendix 6.46), interruption performance errors, $F(1,13) = 0.1840$, $p = 0.6753$ (Appendix 6.47), procedure resumption time, $F(1,13) = 0.8060$, $p = 0.3855$ (Appendix 6.48), or resumptive FPM activity, $F(1,13) = 0.6020$, $p = 0.4517$ (Appendix 6.49).

Disruptive Effects of Similarity

Similarity conditions did not differentially extend ensemble performance time, $F(1,13) = 0.0020$, $p = 0.9611$ (Appendix 6.50); increase ensemble FPM activity, $F(1,13) = 0.0430$, *p*

= 0.8390 (Appendix 6.51); or increase procedure performance errors, *F*(1,13) = 0.582, *p* = 0.4591 (Appendix 6.52).

Effects of Environmental Stress on Interruption Management

Interruptions were hypothesized to be less distracting and more disturbing and disruptive when introduced in the FAF procedure. To isolate the effect of environmental stress, analyses compared only interruption management performance on interruptions presented before the isomorphic 18K' and FAF procedures (IP = 0.02). In addition to these focused analyses, previous task factor analyses are reviewed for significant interactions of environmental stress effects with the Goal-Level, Coupling-Strength, Similarity factors.

Distracting Effects of Environmental Stress

An analysis of variance on acknowledgment times indicated that interruptions during a higher stress condition were less distracting than interruptions during lower stress conditions, *F*(1,13) = 14.962, *p* = 0.0019 (Appendix 6.53).

Disturbing Effects of Environmental Stress

Analyses of variance on interruption initiation time, interruption performance errors, procedure resumption time, and resumptive FPM activity evaluated disturbance effects of environmental stress on interruption management. Environmental stress affected interruption initiation time, *F*(1,13) = 4.226, *p* = 0.0605 (Appendix 6.54) and resumptive FPM activity, *F*(1,13) = 10.788, *p* = 0.0059 (Appendix 6.55), but not interruption performance errors, *F*(1,13) = 0.759, *p* = 0.3993 (Appendix 6.56), or procedure resumption time, *F*(1,12) = 2.290, *p* = 0.1541 (Appendix 6.57). Subjects were, on average, almost 1.3 seconds slower to begin interrupting tasks presented in higher-stress conditions. Subjects made, on average, 51% more active FPM inputs *per* second before resuming procedure performance in higher-stress conditions. While not significant, condition means for both procedure resumption time and interruption performance errors show trends which suggest that interruptions in higher-stress conditions were more disturbing than interruptions in lower-stress conditions.

Disruptive Effects of Environmental Stress

Analyses of variance on ensemble performance times, ensemble FPM activity, and procedure performance errors evaluated the disruptive effects of interruptions due to environmental stress. Subjects exhibited more ensemble FPM activity, *F*(1,13) = 41.156, *p* = 0.0001 (Appendix 6.58), and marginally more procedure performance errors, *F*(1,13) = 3.198, *p* = 0.0850 (Appendix 6.59), when interrupted during higher-stress conditions than lower-stress conditions. Subjects made, on average, 23% more active FPM inputs *per* second during the ensemble interval in higher-stress conditions than in lower stress conditions. Interruptions during higher-stress conditions were significantly associated with shorter ensemble performance times than lower-stress conditions *F*(1,12) = 3.437, *p* = 0.0885 (Appendix 6.60).

Interaction of Environmental Stress and Other Factor Effects

The Procedural Leg factor included in analyses of Goal-Level, Coupling-Strength, and Similarity factors includes two levels of environmental stress by providing similarly-structured procedures at two ground-proximity conditions.[14]. The Procedural Leg factor interacted significantly with the similarity of the interrupting task and interrupted task for interruption initiation time, $F(1,12) = 4.707$, $p = 0.0508$ (Appendix 6.46), and with goal-level of the interruption for resumptive FPM activity, $F(2,25) = 2.846$, $p = 0.0772$ (Appendix 6.33). A contrast of interruption initiation time means for the Similarity X Procedural Leg interaction revealed that pilots initiated similar interrupting tasks more quickly than different interrupting tasks when in the 18K' Procedure, and initiated different interrupting tasks more quickly than similar interrupting tasks when in the FAF Procedure. Inspection of Goal-Level X Procedural Leg resumptive FPM activity means indicated that while resumptive FPM is essentially constant over Goal-Level conditions in the 18K' procedure, it appears that there is much less resumptive FPM after interruptions between procedural tasks than after interruptions either outside the procedure or within a procedural task (Appendix 6.33).

Interaction of Environmental Stress and Subject Effects in Task Factor Analyses

While Subject X Procedural Leg interactions in the analyses for stress effects were insignificant for all dependent measures, all $p > 0.1909$, the Subject factor did interact with Procedural Leg in analyses of other task factors. These interactions were significant in analyses of goal-level effects for subject procedure resumption times, $F(13, 180) = 1.975$, $p = 0.0251$ (Appendix 6.32); procedure performance errors, $F(26, 252) = 1.898$, $p = 0.0307$ (Appendix 6.36); and ensemble performance times, $F(26, 229) = 1.990$, $p = 0.0225$ (Appendix 6.34). Subject X Procedural Leg interactions were moderately significant in analyses of coupling-strength effects for subject interruption initiation times, $F(13, 82) = 1.667$, $p = 0.0839$ (Appendix 6.38) and ensemble FPM activity, $F(13, 74) = 1.807$, $p = 0.0576$ (Appendix 6.44). The Subject X Procedural Leg interaction was also significant in the analysis of similarity effects on subject interruption acknowledgment times, $F(13, 12) = 4.427$, $p = 0.0073$ (Appendix 6.45).

Observations on Individual Differences among Pilots

Although not the focus of this research, individual differences were suspected to account for a large proportion of variance in this data. Analyses of task factor effects examined effects of subjects and interactions of subjects with task factors. This section summarizes the significance of individual differences in performance over all interruption conditions and significant interactions with task factors.

---

[14] Decreasing altitude and distance to runway has been used in previous studies to operationalize stress conditions on the flightdeck (Waller and Lohr 1989; Diehl 1975).

Individual Differences in Interruption Management

Individual differences of subjects were highly significant for many measures, when assessed for all interrupted experimental conditions; interruption acknowledgment time, $F(13,407) = 5.675$, $p = 0.0001$ (Appendix 6.15); interruption initiation time, $F(13,403) = 3.183$, $p = 0.0001$ (Appendix 6.16); resumptive FPM activity, $F(13,280) = 4.564$, $p = 0.0001$ (Appendix 6.61); ensemble performance time, $F(13,382) = 10.094$, $p = 0.0001$ (Appendix 6.62); ensemble FPM activity, $F(13,382) = 19.362$, $p = 0.0001$ (Appendix 6.18); procedure performance errors, $F(13, 417) = 4.801$, $p = 0.0001$ (Appendix 6.19). The number of interruption performance errors committed over all conditions was less significant, $F(13,407) = 1.650$, $p = 0.0694$ (Appendix 6.17). Procedure resumption times were only moderately variable over subjects, $F(13,302) = 1.561$, $p = 0.0954$ (Appendix 6.63). For most measures (*i.e.*, acknowledgment time, initiation time, resumptive FPM activity, ensemble performance time, and ensemble FPM activity), both experimental conditions and subjects were highly significant, $p= 0.0001$. However, for a the error measures (*i.e.*, interruption performance errors and procedure performance errors) individual differences of subjects accounted for a larger proportion of variance than did experimental conditions. Experimental conditions only accounted for more variance than subjects on one measure, resumption time.

Individual Differences and Task Factor Effects

Individual differences were also evident in subject responses to some task factor manipulations. Individual differences were most pervasive in conjunction with task and interrupt modality conditions. Interactions between individuals and task modality accounted for a significant proportion of variance in measures of; interruption acknowledgment time, $F(13,55) = 5.889$, $p = 0.0001$ (Appendix 6.21); interruption initiation time, $F(13,52) = 1.983$, $p = 0.0413$ (Appendix 6.22); and the number of procedure performance errors, $F(13,55) = 3.257$, $p = 0.0011$ (Appendix 6.27). Interactions between individuals and interruption modality also significantly influenced interruption acknowledgment time, $F(13,55) = 6.455$, $p = 0.0001$ (Appendix 6.21); interruption initiation time, $F(13,52) = 4.807$, $p = 0.0001$ (Appendix 6.22). Further, subject responses to task/interrupt modality pairings were significantly different in terms of interruption initiation time, $F(13,52) = 1.839$, $p = 0.0612$ (Appendix 6.22). Subject acknowledgment times for goal-level conditions also differed significantly, $F(26,242) = 6.663$, $p = 0.0001$, and differed for goal-level conditions in different procedural legs, $F(26,242) = 2.664$, $p = 0.0001$ (Appendix 6.29).

Although individual differences were significant in many analyses, some significant task factors effects appeared to be consistent among subjects (Table 6.1). Subjects committed more interruption performance errors in cross-modality conditions than same modality conditions, and by far more errors when a visual task was interrupted aurally (Appendix 6.23). Generally, subjects committed more procedural errors when interrupted visually than when interrupted aurally, and when an auditory task is interrupted than when a visual task is interrupted, but conditions in which an auditory task was interrupted aurally produced by far the most procedure performance errors (Appendix 6.27). Subjects resumed the interrupted procedure more slowly after an interruption between functionally-coupled tasks than after interruptions between physically-coupled or uncoupled tasks (Appendix 6.39). Finally, under conditions of higher environmentally-imposed stress, subjects were slower to begin performing interrupting tasks (Appendix 6.54), engaged in more resumptive FPM (Appendix 6.57), and performed ensemble tasks faster (Appendix 6.58).

Table 6.1 Robust Task Factor Effects

Effect	Dependent Measure	Effect *p*-value	Subject *p*-value
Task X Interrupt Modality	interruption errors	0.0401	0.8614
Task Modality	procedural errors	0.0537	0.1133
Interruption Modality	procedural errors	0.0014	0.1133
Task X Interruption Modality	procedural errors	0.0099	0.1133
Coupling-Type	resumption time	0.0004	0.1133
Environmental Stress	initiation time	0.0605	0.5659
Environmental Stress	resumptive FPM	0.0059	0.2048
Environmental Stress	ensemble time	0.0885	0.3163

Summary of Results by Interruption Management Effect

Table 6.2 summarizes the results of analyses of interrupted vs. uninterrupted trials, all interruption conditions and subjects, and the main factors hypothesized to affect interruption management. While the absolute levels of significance should not be compared across analyses, due to different power of analysis, the pattern of results demonstrates some interesting findings. For the most part, general tests indicated that the disruption measures were sensitive to effects of the interruptions used in this experiment, and that measures of distraction and disturbance, as well as disruption, were sensitive to differences among interruption conditions and subjects. However, the individual task factors tested were only moderately successful in explaining this variability, and had differing success for different dependent measures.

Table 6.2 Summary of Results by Interruption Management Measures and Effects (*p*-values < 0.10 highlighted).

Independent Variable	Distraction	Disturbance				Disruption		
	Ackn.T	Init.T	IT Err	Res.T	Res.FPM	Pr.Err	Ens.T	Ens.FPM
Interrupted v. Uninterrupted	-----	-----	-----	-----	-----	0.0002	-----	0.0438
Subjects(*)	0.0001	0.0001	0.0694	0.0954	0.0001	0.0001	0.0001	0.0001
Interruption Conditions	0.0001	0.0001	0.1388	0.0001	0.0001	0.0012	0.0001	0.0001
Task Modality	0.0585	0.0068	0.5830	0.1384	0.5398	0.0537	0.3345	0.9032
Interrupt Modality	0.3046	0.0989	0.3854	0.5588	0.2466	0.0014	0.0061	0.8660
Task * Interrupt Modality	0.7204	0.0204	0.0401	0.6932	0.3488	0.0099	0.2684	0.3777
Goal-level	0.1684	0.0001	0.8760	0.6977	0.1177	0.3855	0.7417	0.1981
Coupling-Strength	0.0058	0.0017	0.0416	0.0004	0.0778	0.0038	0.8608	0.7851
Similarity	0.9576	0.9885	0.6753	0.3855	0.4517	0.4591	0.9611	0.8390
Environmental Stress	0.0019	0.0605	0.3993	0.1541	0.0059	0.2347	0.0885	0.0001

* Subject factor in "Interruption Conditions" analyses of general effects of interruptions on procedure performance.

Abbreviations: "Ackn.T"= Acknowledgment Time,
"Init.T"= Interruption Initiation Time,
"IT Err"= Interruption Performance Errors,
"Res.T"= Procedure Resumption Time,
"Res.FPM"= Resumptive Flightpath Management Activity Frequency,
"Pr.Err"= Procedure Performance Errors,
"Ens.T"= Ensemble Performance Time,
"Ens.FPM"= Ensemble Flightpath Management Activity Frequency.

7. Discussion of Experimental Results

This discussion primarily focuses on the general effects of interruptions and effects of task factor manipulations on interruption management found in the present simulation experiment. However, prior to this content, I describe the degree to which the experiment attained design goal of simulation validity. Further, I critically consider the advantages and limitations of the simulation environment and implications of these for interpreting and extending experimental results. After discussing the general effects of interruptions and the effects of task factors, I discuss experimental results associated with individual differences in interruption management. Finally, I consider experimental results in the context of the proffered interruption management model.

Simulation Validity

The simulated commercial flightdeck and scenario designed for this experiment successfully allowed context-sensitive introduction of realistic interruptions to ongoing flightdeck tasks, provided keystroke-level data collection of subject performance, and successfully imposed a specific profile of FPM difficulty over the scenario. Analyses also demonstrated that subjects were adequately trained to control FPM on this profile and that FPM skills were relatively stable over testing runs.

Procedures were designed to provide a task context for interruption conditions. These requirements necessitated some tasks that are not typically performed during approach and descent and irregular placement of some tasks. Within these constraints, the procedures were constructed to present tasks in a logical order. The assumption that this order, the procedures, would be consistent with subject orderings of these tasks was validated for only one subject. Comments made by subjects in reviewing the task ordering exercise and inspection of these orders revealed a prevailing strategy; tasks should be performed as soon possible. While temporally-unconstrained tasks were positioned early in subject orders, only tasks that were temporally, or positionally-anchored late in the approach were placed toward the end of the ordered list. In view of this strategy, it is not surprising that most subject orders were not consistent with the procedure order. Although subjects' unfamiliarity with scenario procedures may have influenced their initial acceptance of them, the result that most subject orders differed from the procedure order, and that all subject orders were similar, indicated that subjects were similarly disadvantaged by procedures inconsistent with their preferred orders. This result suggests that subjects' performance was unbiased by their familiarity with procedures, however it also suggests that subjects did not consider this aspect of the scenario consistent with real airline operations. Subjects were, however, trained to criterion on procedure performance.

Subjects performed a pretest exercise to validate coupling-strength and coupling type assignments. Subjective assessments of coupling-strengths and type-assignments for each of the six coupling experimental conditions revealed a clear distinction between coupling-strength for each coupling-type, and demonstrated that subject type-assignments were consistent with the procedural task pairs used to operationalize coupling strength levels in the simulation experiment. Assessments of coupling-strengths for the experimental conditions indicated that subjects perceived functionally-related consecutive procedural tasks as more strongly-coupled than physically-adjacent consecutive procedural tasks.

In addition to these results which address the validity of specific aspects of the scenario, the results of other analyses suggest some measure of overall simulation validity. The simulation employed in this experiment differed from actual aircraft operations in many ways. The TSRV simulator's controls and displays differed somewhat from the aircraft currently flown by the subjects. In deference to experimental control, the experimental scenario included several departures from realism; a one-person crew, lack of external view and surrounding aircraft, intermittent high-difficulty flightpath management, and rigidly proceduralized flightdeck tasks. Finally, the conditions of experimentation, *i.e.*, repeated trials of similar scenarios, and unrealistically high event rates, and unspecified effects on motivation distinguish this simulation from real aircraft operations in the typical case. For this reason, what appear to be high error rates and slow times, particularly acknowledgment times, for some experimental conditions may not be representative of the behavior that would be observed in actual operations. Many of these departures from realism, however, are possible characteristics of high-workload "worst-case" real situations due to incapacitated crew members, low visibility, frequent ATC vectoring, and fatigue. In addition, conducting this experiment in a simulated flightdeck improves generalizability of results over traditional laboratory investigations by operationalizing constructs as realistic conditions, using subjects who represent operators in the actual domain, and an increased, if not perfect, representation of the task environment. The following experimental results, therefore, must be considered in light of the benefits and limitations of this simulation environment.

General Effects of Interruptions on the Flightdeck

The main hypothesis of this experiment is that the intervention of one task into the context of another set of ongoing tasks, here an ongoing procedure, will degrade performance on both the interrupting task and the interrupted procedure.

Performing Tasks that Interrupt

Results indicate that performance on even short, simple, and familiar tasks can degrade performance when embedded unexpectedly in a proceduralized ongoing task set. The effects on interrupting tasks are further interpreted in terms of their operational significance. The, on average, seven seconds required by subjects to respond to an interruption annunciation, and additional five seconds to begin performing the interrupting task may be unacceptable in time-urgent situations. Error rates for interrupting tasks were fairly low compared to traditional laboratory experiments, but seem excessive in the context of real operations. Considering that only six types of interruption performance errors were counted, and that interrupting tasks required only six activities, even this low error rate is noteworthy. While rare, subjects committed some interruption performance errors of particular operational significance; for example, entering the incorrect destination runway, or failing to execute a revision to the flightpath. As all interrupting tasks were interjected into the context of the simulation scenario in procedural intervals, no control condition exists for comparing performance of these tasks as interruptions versus as simple keying exercises without context.

General Effects on Procedure Performance

The effects of interruptions are also evident on the ongoing task set. Here, simple, familiar, and, to some degree, expected, interruptions demonstrated disruptive performance effects on a proceduralized ongoing task set. In particular, results indicated that, as anticipated, interruptions to flightdeck procedures significantly disrupted performance by inducing more errors in procedure performance and increasing FPM activity. The increase in procedure performance errors is most salient and operationally significant. If a procedure is interrupted, pilots are 53% more likely to make an error in accomplishing that procedure than if it is not interrupted. In other words, whereas one can expect some procedure performance error in one of every three uninterrupted procedures, one can expect a procedure performance error in one of every two interrupted procedures. Some of the procedure performance errors committed are particularly operationally significant. For example, an incorrectly-tuned tower frequency minimally causes confusion and increased radio traffic, and maximally, if left uncorrected, could prevent a pilot from receiving life-saving instructions in time to take appropriate actions. Other procedural errors committed in this experiment would not be considered significant in real operations, particularly many sequence errors. However, to the degree that subjects internalized the performance requirements of the scenario, *i.e.*, that procedure tasks must be performed in the order specified, these errors indicate that interruptions reduce the probability that subjects perform as intended[15]. Therefore, whereas not all the specific errors forms manifested in this experiment are of operational significance, it is of utmost operational significance that interruptions demonstrably disrupt intended performance.

Anticipated performance decrements associated with interruptions were also evident by an increased rate of FPM inputs in interrupted procedures. However, this decrement, of approximately 2 FPM inputs *per* minute of the procedural interval, is not likely to be operationally damaging. Because subjects were instructed to focus on procedure performance during procedural intervals and FPM was not required, increased FPM also demonstrated the ability of interruptions to disrupt intended performance, as anticipated.

Finally, subjects were expected to require additional time to integrate the interrupting task with the ongoing procedure. Comparison of composite performance times with ensemble performance times contradicted this hypothesis, and demonstrated that, although only marginally significant, procedure performance times were actually compressed in interrupted procedures. This result is inconsistent with most previous

---

[15] Subjects appeared to internalize these procedure performance requirements, as evidenced by their occasional discussion of operationally-insignificant procedural errors during the post-run period.

research (Kreifeldt and McCarthey 1981; Field 1987; Gillie and Broadbent 1989; Detweiler, Hess, and Phelps 1994), however is consistent with one study (Cellier and Eyrolle 1992). This result may be interpreted as evidence that subjects responded to the temporal pressure induced by high-FPM workload conditions at the end of procedural intervals and compensated for the additional demands imposed by an interruption by performing procedural tasks in the post-interruption period faster. Cellier and Eyrolle (1992) explain their similar results as a mobilization of untapped resources. The results are also consistent with previous research which suggests that people strategically manage tasks to modulate workload (Hart and Wickens 1981; Moray and Hart 1990; Segal and Wickens 1991; Hancock 1991). The observed increased error rates in interrupted tasks may be the result of a speed/accuracy trade-off effect rather than a direct effect of interruption disruption. An alternative explanation is that interruption management was simply not time-consuming, perhaps that subjects adopted a mechanism for integrating interruptions easily (*cf.* Hess and Detweiler 1994). Although this interpretation might explain results of time comparisons, it is inconsistent with evidence of increased procedure performance errors and FPM activity.

The significant, although operationally minimal, effects on interrupting task performance and effects on interrupted procedures demonstrate experimentally what has previously been primarily anecdotal (*e.g.,* Monan 1979; Barnes and Monan 1990; Degani and Wiener 1990; Turner and Huntley 1991) and, in a few cases, observed in retrospect (*e.g.,* Linde and Goguen 1987; Williams 1995); that is, that interruptions, even familiar, simple, interruptions, measurably degrade performance over uninterrupted conditions. Accidents typically result from the confluence of several off-normal conditions. While the degree of performance degradation induced in this experiment was not, overall, of dramatic operational significance, results provide empirical support for the consideration of even familiar, expected, and straightforward flightdeck interruptions as contributing factors in accident and incident analyses.

Influence of Task Factors on Interruption Management

There are deleterious effects of performing a task when it intervenes during an ongoing procedure as well as performance degradation effects of an interruption on the interrupted procedure. Analyses indicate that most performance measures were significantly affected by the various experimental conditions employed in this experiment. This result alone indicates that there are some systematic contextual factors which mediate the degree to which an interruption degrades performance. This experiment found supporting evidence that four of the five task factors had significant main effects on interruption performance. This section discusses task factor manipulations in terms of their distracting, disturbing, and disruptive effects, in this order.

Influence of Interruption and Task Modality

Modality influences include those of the interrupting task, the interrupted task. Further, they include the interaction of the interrupting and interrupted task modalities. The following results are presented according to these three categories of effects.

Influence of Interruption Modality

On average, auditory interruptions were acknowledged faster than visual interruptions, although this difference was not significant. This trend is consistent with previous research claiming that aurally presented information is more attention-directing, "alerting", that visually-presented information (*e.g.*, Neisser 1974; Posner *et al*. 1976; Segal and Wickens 1991; Stanton 1992). While the experimental conditions used to test this hypothesis are realistic ATC clearance presentations, the expected alerting advantage of aurally-presented interruptions was not ideally examined. This advantage may only be evident in a comparison of aural and visual interruption annunciation, which convey the same amount of information about the performance requirements of the interruption. This advantage would, then, be evident in a measure of diversion, *e.g.*, simple response time to a content-less annunciation stimulus. In this experiment, the datalink (visual) condition announced the existence of an interruption aurally and then presented information required to interpret the interruption visually. In contrast, in the radio (auditory) condition, the aurally-presented interruption annunciation stimulus also conveyed the message content. Additionally, the datalink condition required attention switching from a short auditory annunciation signal to the visual content before acknowledgment, the radio condition did not require this attention switching. Such, attention switching between modalities is time consuming (*e.g.,* Wickens 1984) and may have contributed to the effect on acknowledgment time. Mean acknowledgment times for the datalink conditions in this experiment are slightly longer than the, on average, 10 seconds found in previous investigations (Kerns 1990). The insignificant trend observed in this experiment is counter to previous results that suggest pilots interpret and acknowledge datalink messages faster than voiced messages (Kerns 1990).

There was no significant difference in either procedure resumption time or standardized resumptive FPM activity for datalink (visual) and radio (auditory) interruptions in this study. This result is contradictory to findings that suggest that pilots take longer to recover from datalink interruptions than from voice interruptions (Williams 1995). This discrepancy might be attributed to different implementations of the datalink technology. In the current implementation, the datalink system is on a dedicated CRT located over the throttle quadrant. Williams tested a shared-display implementation of datalink by incorporating this functionality in the FMS/CDU. Although both implementations initially indicate datalink interruptions aurally, the content of the interruption message is immediately available in the current implementation but must be accessed with a button press in Williams' implementation.

Subjects in this experiment initiated aurally-presented interruptions significantly faster than visually-presented interruptions following acknowledgment, although inspection of interaction means indicated that this effect was significant only for interruptions to auditory ongoing tasks. This result was expected due to the relative persistence of an auditory annunciation message, and therefore continued attention-directing (*e.g.*, Wickens 1984). In addition, as information required to accomplish visually-presented interruptions was externally-persistent, subjects may have strategically utilized this interface feature to delay performance on visually-presented interruptions. Subjects could acknowledge datalink (visual) interruptions by using the STANDBY key, and retain clearance information on the screen. Although subjects could also acknowledge aurally-presented interruptions with a verbal "standby" reply and re-engage ATC later to obtain clearance information, this behavior is much more time consuming than the analogous behavior with the datalink condition.

Auditory interruptions were more disruptive to procedure performance than visual interruptions as evidenced by extending ensemble performance times and increased procedure performance errors. The ability to delay interpretation of the annunciation message and the relative ease of repeated access to this information; in short, the greater flexibility afforded in managing visual interruptions, seems to reduce the deleterious effects on procedure and ensemble performance. Other research has noted that the flexibility afforded by datalink technology may decrease pilot workload (Hrebec *et al.* 1994).

Influence of Task Modality

Task modality was a significant factor in determining acknowledgment and initiation times, as well as procedure performance errors, as anticipated. Subjects were much slower to respond to, and to begin interrupting tasks, and procedural performance errors were twice as likely when an auditory task was interrupted than when a visual task was interrupted. This result can also be interpreted in terms of the external permanence provided by the visually-presented interruption messages. These results are consistent with previous research suggesting that interfaces which provide an external index of the interruption point reduce memory load and, therefore, mitigate deleterious effects of interruptions (Kreifeldt and McCarthey, 1981; Field 1987; Degani and Wiener 1990).

Interaction of Interruption and Task Modality

Cross-modality conditions were hypothesized to disturb and disrupt procedure performance less than same-modality conditions. This hypothesis was supported by only one result. Only the auditory/auditory same-modality condition's effect on procedure performance errors demonstrated the hypothesized performance degradation predicted by multiple-resource attention theory (*e.g.*, Wickens 1984). This confirming evidence is apparently contradicted by the result that the visual/visual condition was the condition least prone to procedure performance errors. Re-examination of experimental conditions and observational notes suggests an explanation for the dissociation of the same-modality conditions. The original modality interaction hypothesis assumed that interpretation of the interruption and some portion of the ongoing task would be performed coincidentally, or time-shared. The datalink system was located outside the immediate visual field of the interrupted task focus and may have made timesharing these two tasks unlikely for visual/visual conditions. Performance on same-modality conditions that subjects managed serially rather than in parallel may have benefited from the facilitating effects of resource priming (Wickens 1984, p. 253).

Other significant effects contrary to this hypothesis dictated a closer inspection of results. Subject responses were significantly associated with task and interruption modality interactions for only three measures, interruption initiation time, interruption performance errors and procedure performance errors. These significant interactions each indicated that subject performances were better for same-modality conditions than cross-modality conditions. Further inspection of interaction means indicated that in each case, one experimental condition's mean was significantly larger than, approximately twice, any of the other three: (1) Interruption performance errors are much more prevalent when aurally-presented interruptions interrupt visual tasks. (2) Visually-announced interrupting tasks are much more slowly initiated when they interrupt auditory tasks. (3) Procedure performance errors occur much more frequently when aurally-presented interruptions occur to an auditory ongoing task. These three cases may be explained by the degree to which the interruption or the interrupted task is externally-available. In the first case, performance associated with the interruption suffers because information regarding the interrupted task, and, therefore, also the position in the procedure, is externally available. The second case is explained above as an artifact of the misuse of datalink response keys. In the third case, procedure performance is most degraded by the condition which most significantly loads memory by providing external cues for neither the interruption nor the interrupted task, and requires the same processing resources.

Influence of Interrupted Goal-level

Results did not statistically confirm the hypothesis that subjects were less distractible when interrupted at lower levels of a procedure goal hierarchy, or that disruption increased with interruption goal-level. However subject interruption initiation times confirmed a weak form of this hypothesis: Execution of interruptions to the lowest level of the procedure was more disturbing than interruptions either between procedural tasks or external to procedure performance. Inspection of residuals by goal-level conditions did not exhibit within level variations indicative of a spurious effect from coupling or similarity conditions. Disruptive effects due to goal-level manipulations were not apparent and may have been offset by strategic delays in actually performing the interrupting tasks until more easily integrated.

The Subject X Goal-Level interaction was more significant than the goal-level effect in measures of interruption acknowledgment time, interruption performance errors, procedure performance errors, and standardized ensemble FPM activity. This significant individual variability over goal-level conditions may have masked a latent goal-level effect. Inspection of condition means indicates two trends consistent with hypothesized effects for two of these measures, interruption acknowledgment time and procedure performance errors. Although not statistically significant, trends suggested that: (1) Subjects tended to acknowledge interruptions more quickly to the degree that they were less embedded in the ongoing procedure. (2) Interruptions were more likely to induce procedural performance errors to the degree that they were imbedded in the procedure. However, not all trend information indicates the hypothesized effects. Two other non-significant trends were inconsistent with hypothesized effects: (1) Subjects seemed less likely to err in interruption performance to the degree that the interrupt was embedded in the procedure. (2) Interruptions external to procedure performance appeared to induce more FPM activity than interruptions at the lowest goal-level, which induced more FPM activity than interruptions induced between procedural tasks.

Although some main effect and trend evidence are consistent with the goal-level hypothesis, the preponderance of the evidence suggests that this factor does not significantly affect interruption management performance. These results are inconsistent with predictions of Adams, Tenney, and Pew (1991, 1995), research operationalizing memory-load as a level of procedure nesting (Detweiler, Hess, and Phelps 1994), and research in speech perception (Cairns and Cairns 1976). Results are consistent with a previous failed attempt to demonstrate that goal-level determines vulnerability to distraction (Lorch 1987). Lorch (1987) proposed that although her results did not indicate a significant effect of goal-level, the effect might be demonstrated in a more realistic task context. The present experiment failed to demonstrate this effect strongly but suggests that the effect may not be significantly evident due to subject differences and adaptive strategies or reflexes to minimize performance decrements.

Influence of Procedural Task Coupling

The coupling hypothesis is based on research in procedure performance which suggests that operators arrive at associations among procedural tasks (Elio 1986). The associated tasks, then, form a subset which is performed as a unit, with diminished need for attentional control, and therefore is more impervious to interruption (Shiffrin and Schneider 1977; Schneider and Shiffrin 1977). In the aviation domain, pilots refer to this notion as “flow”; that is, they actively attempt to associate tasks to “make sense”. Subjects frequently noted their reasoning in developing these associations during the experiment’s phase 1 training sessions for procedure performance. This experiment assumed that subjective ratings of coupling-strength between adjacent procedural tasks would validly represent internalized representations of procedural task associations.

Subjects did rate coupling-strength levels commensurate with designed coupling-types. However, their performance when interrupted between adjacent tasks of high, medium, and low coupling-strengths did not confirm hypotheses that interruptions between more strongly-coupled tasks would be less distracting, and more disturbing and disruptive. Coupling manipulations did significantly affect several measures, however the directions of these results were mixed. Figure 7.1 represents, schematically, the relationships between perceived coupling-strength ratings and condition means of significant coupling effects. Subjects were least distractible when interrupted between tasks of moderate coupling-strength; that is, involving proximal activation. Interruptions between moderately-coupled tasks were more disturbing than uncoupled or highly-coupled tasks, in terms of interruption initiation and resumption time. However they were less disturbing than uncoupled and highly-coupled tasks in terms of resumptive FPM and interruption performance errors, respectively. Subjects were more likely to commit an interruption performance error when it occurred between highly-coupled tasks. Disruptive effects were only evident in the form of procedure performance errors and indicated that interruptions between uncoupled tasks were more disruptive than interruptions between coupled tasks.

None of these measures statistically support the strong form of the coupling-strength hypothesis, that performance effects due to an interruption between two adjacent tasks are proportional to the perceived coupling-strength of these two tasks. The incidence of interruption errors provides evidence for a weakened form of the hypothesis: Interruptions between highly-coupled tasks are more error prone than those interruptions between either uncoupled or moderately-coupled tasks. Lack of supporting evidence in other performance measures suggests that perceived coupling-strength of severed tasks is not a particularly useful construct, by itself, for predicting the degree of distraction, disturbance, or disruption induced by interruption between tasks. One explanation of the failure of such a hypothesis is that subjects were unable to use the rating exercise to accurately reflect the strength of coupling experienced when actually performing tasks. However, the internal consistency and salience of these results suggests a more complex

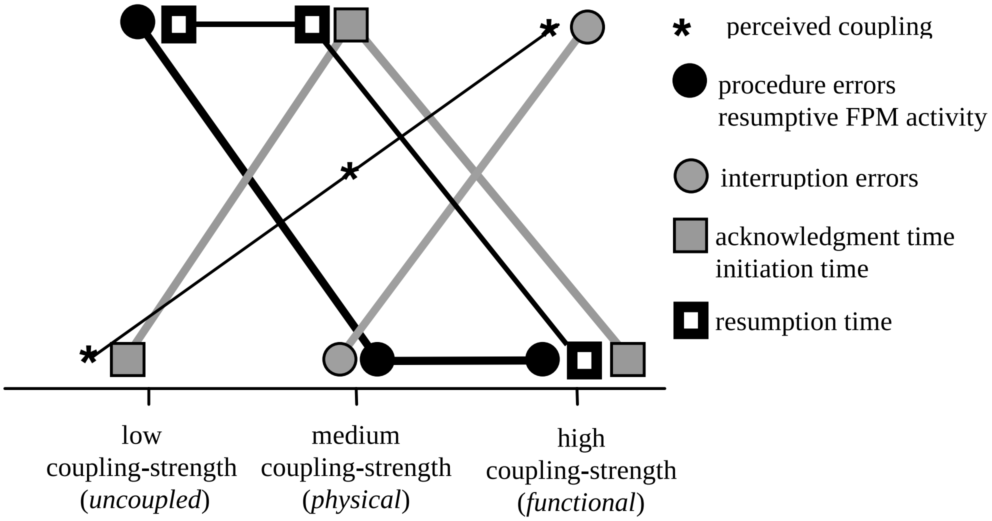


Figure 7.1. Schematic of Coupling-Strength Condition Means for Significant Effects.

interpretation. More likely, rather than increasing levels of coupling-strength, qualitative characteristics of these coupling types, defined by characteristics of the antecedent task and the subsequent task, affect performance differently.

Reinterpreting these results in terms of coupling-types rather than levels of coupling-strength reveals potential explanations for this pattern of results. To restate the results in terms of coupling-types: (1) Interruptions between uncoupled tasks are disruptive, causing more errors in the post-interrupt procedure. (2) Interruptions between physically-coupled tasks are least distractible and most temporally disturbing. (3) Interrupting task performance between functionally-coupled tasks is most error prone. To understand this pattern of results, I review the nature of the experimental conditions operationalizing these coupling-types. Uncoupled conditions in both the 18K' and FAF procedures were operationalized by an antecedent task requiring simple menu selection and data entry in the FMS/CDU and a subsequent task requiring a radio call in which information is received. Physically-coupled conditions in both the 18K' and FAF procedures were operationalized by two simple manual tasks, proximally-located. Functionally-coupled conditions in both the 18K' and the FAF procedures were operationalized by an antecedent task requiring subjects to illuminate a cabin sign by pressing an overhead panel button, and a subsequent task requiring subjects to communicate to the passengers information related to that sign. Note that the subsequent tasks in both the uncoupled and functionally-coupled conditions are, relatively, lengthy verbal tasks, whereas the subsequent task for the physically-coupled condition is a simple, fast manual task. Completion of antecedent tasks in both the physically-coupled and functionally-coupled conditions are externally visible. Interruptions between physically-coupled tasks were least distractible and most temporally disturbing. Performance of those interrupting tasks announced between functionally-coupled tasks was most error prone. Interruptions between uncoupled tasks instigated the highest incidence of procedure performance errors and resumptive FPM activity.

Reconsidering antecedent and subsequent task characteristics suggests the following explanation for the pattern of significant results. The antecedent tasks of both the physically-coupled and functionally-coupled conditions provide obvious indications of having been performed and therefore provide an externalized representation of the interruption and resumption point, the uncoupled condition does not. Previous research indicates that providing an externalized representation of the interruption point facilitates post-interruption performance (Kreifeldt and McCarthey 1981; Field 1987; Degani and Weiner 1990). Significantly higher procedural error rates for the uncoupled condition may be attributed to the absence of an obvious interruption position marker.

The pattern of results associated with time measures reflects a typical strategy that subjects employed. Subjects interrupted between physically-proximal tasks rarely preempted performance of the subsequent task to perform the interruption. Rather, in this condition, subjects completed the subsequent task before acknowledging the interruption. Such behavior may be either strategic or automatic. Subjects may recognize that this interruption condition does not provide an external representation of the interruption point, that there is no conceptual link to the next task, and determine that they should not interrupt the procedure at that point. Alternatively, proximal and consecutive procedural tasks may be compiled as an automated sub-unit of the procedure and simply may be resistant to interruption (*e.g.*, Muller and Rabbitt 1989).

Results associated with interruption errors provide the only, albeit weak, evidence that perceived coupling-strength affects interruption performance in the hypothesized direction. Rather than considering this result as derivative of perceived coupling-strength, one might interpret this result in terms of activation theory (*e.g.*, Anderson 1976; Adams, Tenney, and Pew 1991, 1995). Because antecedent and subsequent tasks in the functionally-coupled conditions are semantically-related, performing the antecedent task in this condition theoretically accentuates the activation level of the subsequent task, and therefore avails less resources for managing the interruption. According to this interpretation, and extending the prediction of memory-based intention theories (*e.g.,* Miller, Galanter, and Pribram 1960), the increased activation of the functionally-coupled condition's subsequent task may facilitate procedure resumption more than conditions in which the antecedent task has less priming effect on the subsequent task. Experimental results confirm this interpretation and provide supporting evidence that task tension, here, association strength of procedural tasks, is memory-based (*cf.* Adams, Tenney, and Pew 1991). Procedure resumption times and standardized resumptive FPM indicate significantly more efficient procedure resumption for functionally-related conditions than uncoupled conditions.

Influence of Task and Interruption Similarity

No main effects of task/interruption semantic similarity were evident. Previous research suggests that pre-load (ongoing) and loading (interrupting) tasks similar in resource demands result in interference and associated performance decrements (*e.g.,* Liu and Wickens 1995). Other research suggests that interruptions which activate knowledge structures consistent with previously activated structures are more easily processed than those that require activation of competing structures (*e.g.,* Adams, Tenney, and Pew 1991, 1995). The experimental hypothesis asserted that in a relatively realistic environment, imbued with semantic meaning, effects of facilitation would outweigh effects of resource interference for simple tasks. The insignificance of similarity manipulations in this experiment can not be distinguished from counteracting effects of interference and facilitation.

Alternative explanations for this lack of significance may be traced to the operationalization of experimental conditions. To adequately test this hypothesis, the selected interrupting tasks and interrupted tasks would need to evoke task-related knowledge to working memory, and this knowledge would need to be consistent with that of the ongoing task for the 'similar' condition, and would be inconsistent for the 'dissimilar' condition. In this experiment, the first assumption may have failed if subjects performed these tasks by rote; that is, without evoking conceptual constructs associated with the task. Due to time constraints and the familiarity and simplicity of the interrupting tasks, it is highly possible that subjects performed these tasks automatically, without activating, to any useful degree, semantic information related to the interrupting task. This possibility garners support from the general observation that experimental manipulations, if they had an effect on performance, rarely produced propagating, or disruptive, effects to the remaining procedure. Secondly, interruptions were paired with interrupted tasks to construct the 'similar' and 'dissimilar' conditions based only on expert pilot judgment. The evocative strength and contents of knowledge structures associated with these interrupting tasks were not pre-tested to operationalize these conditions.

A marginally significant interaction of similarity with procedure leg for interruption initiation time revealed that pilots initiated similar interrupting tasks more quickly than different interrupting tasks when in the 18K' Procedure, and initiated different interrupting tasks more quickly than similar interrupting tasks when in the FAF Procedure. One explanation for this effect is that interference effects may be exacerbated in more attention-demanding environmental conditions and may overwhelm any facilitating effect of semantic similarity evident in less stressful conditions. An alternative explanation is that the interrupting tasks are significantly different in some aspect other than semantic category. While the two interrupting tasks were designed to be as alike in annunciation signal and keystroke requirements as possible, the annunciations differed in the digits and units announced and by the utterance of an extra digit for the altitude change interruption. The interrupting tasks differed only in that in the altitude change interrupting task, subjects typed "6500" and in the speed change interrupting task, subjects typed "160/" (Appendix 5.10). These experimental conditions used to implement similarity conditions prevent distinguishing between these two possible explanations.

Influence of Environmental Stress

Increased environmental stress, imposed by increased ground proximity, longer interruption acknowledgment times, longer interruption initiation times, and increased active FPM rates in both the resumptive interval and ensemble interval as a whole. These results are consistent with theories suggesting that stressful conditions diminish the availability of attentional resources (Eysenck 1982; Easterbrook 1959; Kahneman 1973) and confirm the hypotheses that environmental stressors would decrease the distractibility and increase the disturbance and disruption induced by an interruption. These results are, however, inconsistent with results from datalink studies indicating that acknowledgment times decrease in lower altitude, higher stress conditions (Kerns 1990).

While ensemble FPM activity and procedure performance error results confirmed expectations, subject ensemble performance time response did not. Ensemble performance times were hypothesized to increase due to disruption induced by an interruption at higher stress levels. Results indicated that ensemble performance times during higher stress levels actually were shorter than those for lower stress conditions. Alternatively, subjects may compensate during the post-interrupt procedure for having been interrupted, and that compensation was more pronounced in higher-stress conditions. Results of analyses comparing ensemble performance times to composite times provide evidence of this strategic compensation for deadline conditions. Cellier and Eyrolle (1992) observe this compensatory behavior in response to time pressure, a form of environmental stress. In light of this other evidence, the compressed ensemble performance times during higher environmental stress conditions may be cautiously interpreted as supporting evidence for the exacerbating effects of environmental stress on interrupted procedure performance.

Individual Differences and Interruption Management

Significant individual differences were expected based on previous research relating operator characteristics to performance differences in interrupting conditions (*e.g.*, Kirmeyer 1988; Jolly and Reardon 1988) and the flexibility afforded by this relatively realistic task environment, even given significant restrictions in scenario performance. Results indicated that subjects were significantly variable in many interruption management performance measures. Although not surprising, this result is important for several reasons. First, it is methodologically important, underscoring the importance of a within-subject experimental design in studies investigating interruptions. In analyses of the full set of interruption conditions, several measures accounted for more variability than did differences among subjects; *i.e.*, interruption initiation time, procedure resumption time, ensemble performance time, and standardized ensemble FPM activity, differences. These measures may therefore be more sensitive measures of interruption management effects if a within-subject design is not possible.

Second, in several cases, subject variability interacted significantly with experimentally-manipulated factors. These subject-by-factor interactions were most prevalent across measures for task and interruption modality conditions, and for the interaction of task and interruption modalities. Subjects also responded differently to the goal-level factor, and the interaction of the goal-level factor and environmental stress condition for isolated measures. Dependent measures that captured these subject-by-task factor interactions include principally interruption acknowledgment time, interruption initiation time, and in one case each, procedure performance errors and standardized resumptive FPM activity. These significant interactions provide a foundation for investigating individual differences in interruption management and, ultimately, determining significant operator characteristics that mediate interruption management performance. Kirmeyer (1988) found that type-A personalities report more active control actions in post-interrupt periods than type-B personalities. Significant differences in resumptive FPM among subjects in the present study may be indicative of a similar effect.

Finally, subject variability contributed significantly to experimental variance in most analyses. However in some cases, subject variability was not significant, indicating that, some effects of independent task factors on those interruption management performance measures are consistent across subjects. These cases predominantly involved measures of disturbance and effects of modality and environmental stress.

Results and the Interruption Management Model

Results from the task factor experiments contribute to an understanding of the benefits and limitations of the interruption management model. First, I consider task factors in terms of the interruption management stages. I then discuss the interruption management dependent measures used in the present experiment as reflections of the interruption effects constructs (distraction, disturbance, and disruption) proposed by the interruption management model.

Effects of Task Factors

Experimental results are reviewed for those factors which significantly contribute to the distracting, disturbing, and disruptive properties of interruptions on the flightdeck.

Properties of Interruptions that Distract

Factors that appear to have most significantly affected the distractibility of an interruption include task modality, coupling-strength/type, and environmental stress. Interruptions within visual tasks, between uncoupled or functionally-coupled tasks, and in low stress conditions were more distractible than interruptions to auditory tasks, between physically-proximal tasks, or in high stress conditions.

Properties of Interruptions that Disturb Performance

All task factors significantly affected initiation time except the similarity factor. According to this measure, an interruption was particularly disturbing if it was an aurally-presented and occurred to an auditory task, presented within a procedural task, presented between physically-coupled tasks, or presented in a high stress condition. Whereas aurally-presented interruptions to auditory tasks resulted in initiation time disturbance, cross-modality conditions and interruptions between functionally-coupled tasks resulted in disruption to interruption performance accuracy. Temporal disturbances were induced in the resumptive period by interruptions between either uncoupled or physically-coupled tasks. Subjects performed activity in the resumption interval indicative of disturbance when interrupted between uncoupled tasks or in high stress conditions.

Properties of Interruptions that Disrupt Performance

Disruptive effects were illustrated equally by measures of procedure performance and ensemble performance time. Procedure accuracy disruptions were induced most significantly by auditory interruptions, and interruptions to auditory tasks and between uncoupled procedural tasks. Auditory interruptions and interruptions during low stress conditions appear disruptive by extending ensemble performance times. As previously mentioned, ensemble performance time may not be a valid measure of disruption when low-stress conditions provide no incentive to complete the ensemble task as soon as possible. Although composite times were significantly longer than ensemble times, this effect was only modestly significant. In addition, a more robust disruption measure, FPM activity, indicated that higher stress conditions were more disruptive than lower stress conditions. Generally, then, the most reliable evidence confirms hypothesized effects of environmental stress.

Measuring Interruption Management Performance

The interruption management model described effects of interruptions; diversion, distraction, disturbance, disruption. Dependent measures were constructed to measure effects of distraction, disturbance and disruption on the flightdeck. A review of experimental results indicates that these measures successfully demonstrate deleterious effects of interruptions in general, and the differences between interruption conditions and individual subjects, in general. Interruption initiation time was most sensitive to manipulated factors, demonstrating significant differences for all but the similarity factor. The number of procedure performance errors also distinguished among factor conditions for more than half of the specific hypothesized effects. Procedure resumption time and ensemble FPM activity were particularly unaffected by most manipulations.

In general, measures prior to actually performing the interrupting task were more significant, and significance more prevalent, than measures associated with performing the interrupting task and subsequent procedure resumption and performance. These results combine to suggest that the experimental conditions may have primarily affected attentional focus and working memory load at the time of interruption, rather than the more extensive memory manipulations that would be expected in later stages of the model. To the extent that interruptions are familiar, easy, expected, they will require less attentional control and working memory to perform. To the degree that interrupting task performance is so automated, disruptive effects should be minimized. As current airline pilots, experimental subjects routinely encounter in real operations the form of interruptions provided in this experiment. Because subjects were interrupted in a well-structured task context on which they were well-trained, and most certainly came to expect these interruptions, it is reasonable, in retrospect, that most measures of disruption and measures of disturbance after initial departure of the ongoing task would not be particularly affected. This interpretation, and the dissociation of these measures for several factors, suggest that the grouping of disturbance measures used in this experiment may be insensitive to an important distinction. In particular, a distinction should be made between measures prior to actually performing the interrupting task, those associated with the time course and quality of interrupting task performance, and those addressing resumption performance. This experiment did not use a measure of diversion. However results of the modality hypothesis suggest the utility of such a measure to determine the role of interruption modality in initially alerting an operator. A measure of diversion would capture the operator’s initial awareness of the annunciation stimulus and may potentially be best obtained through EEG or oculometer measures for visual annunciation stimuli.

8. Conclusions

The immediate benefits of this research are a set of empirically-derived results describing interruption management on specific task factors in a relatively realistic operational context, and the interruption management model. I provide a summary of results which draws together the most significant empirical findings and inferences from the simulation experiment. Experimental results contribute to a better understanding of interruption management and suggest modifications to the interruption management model. This revised model is presented. I discuss the implications of interruption management research to improving flightdeck performance as an example of a complex multi-tasking domain. Finally, in broader context, I discuss future possibilities for exploring interruption management.

Summary of Experimental Results

Results from the simulation experiment indicate several general conclusions, offering both empirical insight into flightdeck interruption management performance and observations which inform the methodology of investigating interruptions on the flightdeck. General empirical findings include: (1) In a relatively realistic task context, even simple, routine interruptions significantly, and operationally degrade performance of an ongoing procedure and appear to motivate compensatory strategies. (2) Significant effects on interruption performance, on procedure performance, and on the ensemble task set performance are dependent, in most cases, on both subject variability and experimental manipulations of task factors and reflected primarily in measures of interrupting task initiation time, procedure performance errors, and interruption acknowledgment time. (3) Although not always in the expected directions, experimental conditions associated with modality, coupling-strength, and environmental stress showed the most prevalent significant effects on interruption management measures. (4) Significant interactions of individual subject differences and task factors provide a foundation for investigating operator characteristics associated with interruption management performance. (5) Although individual differences were significant in most cases, several significant task factor effects appeared to be consistent across subjects. In particular, these, more stable, effects warrant closer inspection of the contextual variables involved and relationships between independent and dependent variables.

Methodological observations include: (1) A flight simulation environment can be used effectively to experimentally investigate the effects of interruptions as an experimental platform which allows some degree of experimental control and preserves some aspects of the actual environment for increased generalizability. (2) Alternative explanations for unanticipated significant effects rely on reconsidering the nature of experimental conditions in other terms; primarily, the degree of memory load imposed. (3) Measures of interruption management demonstrated that the hypothesized factors, given the experimental conditions, most significantly affected early model stages. (4) Failure to demonstrate hypothesized effects in many disruption measures, particularly temporal effects, may be attributed to lack of experimental power and/or compensatory behavior in response to interruptions.

Interruption Management Model Development

Previously, I proposed a model of interruption management that defines information processing stages associated with processing an interruption. Processing an interruption through these stages results in specific effects on the interrupted procedure, defined as diverting, distracting, disturbing, and disruptive effects. The model identified factors hypothesized to affect interruption management and experimentally investigated in this simulation study. The model also defined effects of interruptions. These effects then were translated into dependent measures which characterized flightdeck interruption management in the present experiment.

Experimental results suggest two elaborations to this model. First, experimental results indicated that measures of disturbance dissociate, and that the model does not consider an important distinction, that is the time course of performance. One example of this dissociation is found in the coupling results. The first elaboration refines the notion of disturbance as an interruption effect in response to these results. Secondly, experimental results also indicated that patterns of behavior emerge over the measures of interruption management. These patterns suggest that subjects may employ strategies for interruption handling based on many situational characteristics; for example, the degree to which the interruption point is externally represented. An explicit example of this is found in results of modality conditions, where performance decrements were highly specified to the task/interruption modality conditions. This result indicates that subjects behave qualitatively differently in response to these different interaction conditions. Coupling and goal-level results also suggest this phenomenon, although through interpretation rather than empirically demonstrated. In all, experimental observations emphasize the importance of considering a spectrum of interruption effects, such as those suggested by the interruption management model. Further, these observations suggest that more qualitative approaches to studying interruption management strategies may be appropriate, particularly in realistic, more variable experimental environments. Toward this end, I extend the model of interruption management to describe five potential interruption integration strategies and consider these strategies in terms of the previously defined interruption effects.

Expanding "Disturbance" as an Effect of Interruptions

Disturbance, as originally defined, describes the effects of an interruption associated with integrating it into ongoing performance. Initially, measures for the sub-stages of integration; procedure preemption, interruption performance/scheduling, and procedure resumption, were grouped and assumed to reflect similar performance effects. Experimental results indicated that in fact independent factors appear to affect these sub-stages differently, requiring a finer definition of this construct. The interruption management model, then, is reconstructed to distinguish between these constituents of disturbance; preemptive disturbances, performance disturbances, and resumptive disturbances. This distinction provides a finer framework for identifying effects of task factors on interruption management performance.

It also provides for better definition of interruption management performance measures. These correspond directly to the measures used in this experiment. Preemptive disturbance was measured by initiation time. Performance disturbance was measured by interruption performance errors. Resumptive disturbance was measured by resumption time and the amount of unnecessary activity during the resumption interval. Explicitly defining these intervals suggests other potentially useful measures: unnecessary activity during the preemption interval, a time measure of interruption performance, and probability of resuming at the departure point.

Interruption Management Behaviors

The extended model of interruption management illustrates five possible behaviors an operator may exhibit when an interruption occurs. This section describes the *Oblivious Dismissal, Unintentional Dismissal, Intentional Dismissal, Preemptive Integration,* and *Intentional Integration* behaviors in terms of the model stages. These behaviors are illustrated as five paths in the interruption management model (Figures 8.1 through 8.5).

Detection and *Oblivious Dismissal*

The initial conditions of the model state that operators are engaged in an ongoing procedure, composed of a sequence of tasks. The interruption is introduced by an annunciation stimulus. If the annunciation stimulus is not salient enough to be *detected*, given available perceptual resources, the operator has no awareness of the stimulus. This interruption is *obliviously dismissed* by the operator (path 1, Figure 8.1). The operator does not perform the interruption in this iteration of the model and, unless continued presence of the annunciation stimulus prompts reiteration of the interruption management process, this interruption will not be addressed.

Interpretation and *Unintentional Dismissal*

Given that the operator detects the annunciation stimulus, providing a sensory representation of this stimulus, it is then incumbent upon him/her to interpret this annunciation in terms of the performance requirements of the interrupting task. If interpretation does not occur, the operator does not have a representation of performance requirements and therefore is not compelled to and cannot perform the associated task. This interruption is *unintentionally dismissed* and the operator does not perform the interruption in this iteration of the model (path 2, Figure 8.2). However, a representation of the annunciation stimulus remains for a short time in the sensory store. This interruption will not be addressed in future iterations unless continuation of the annunciation stimulus prompts reiteration of the interruption management process, or the sensory store representation induces intentional perceptual sampling and some other indication of the interrupting task is evident.

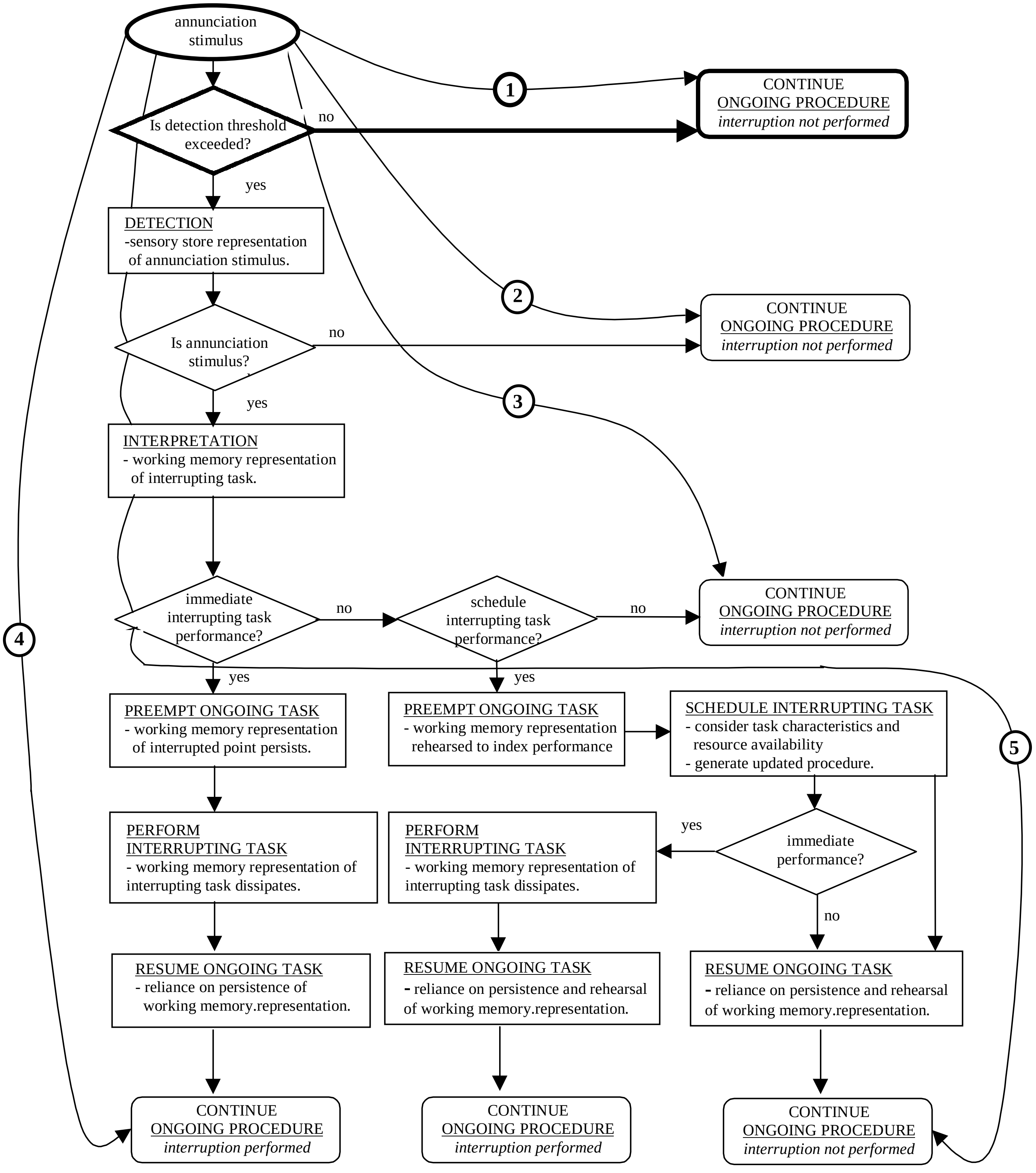


Figure 8.1. The Interruption Management Model & Oblivious Dismissal

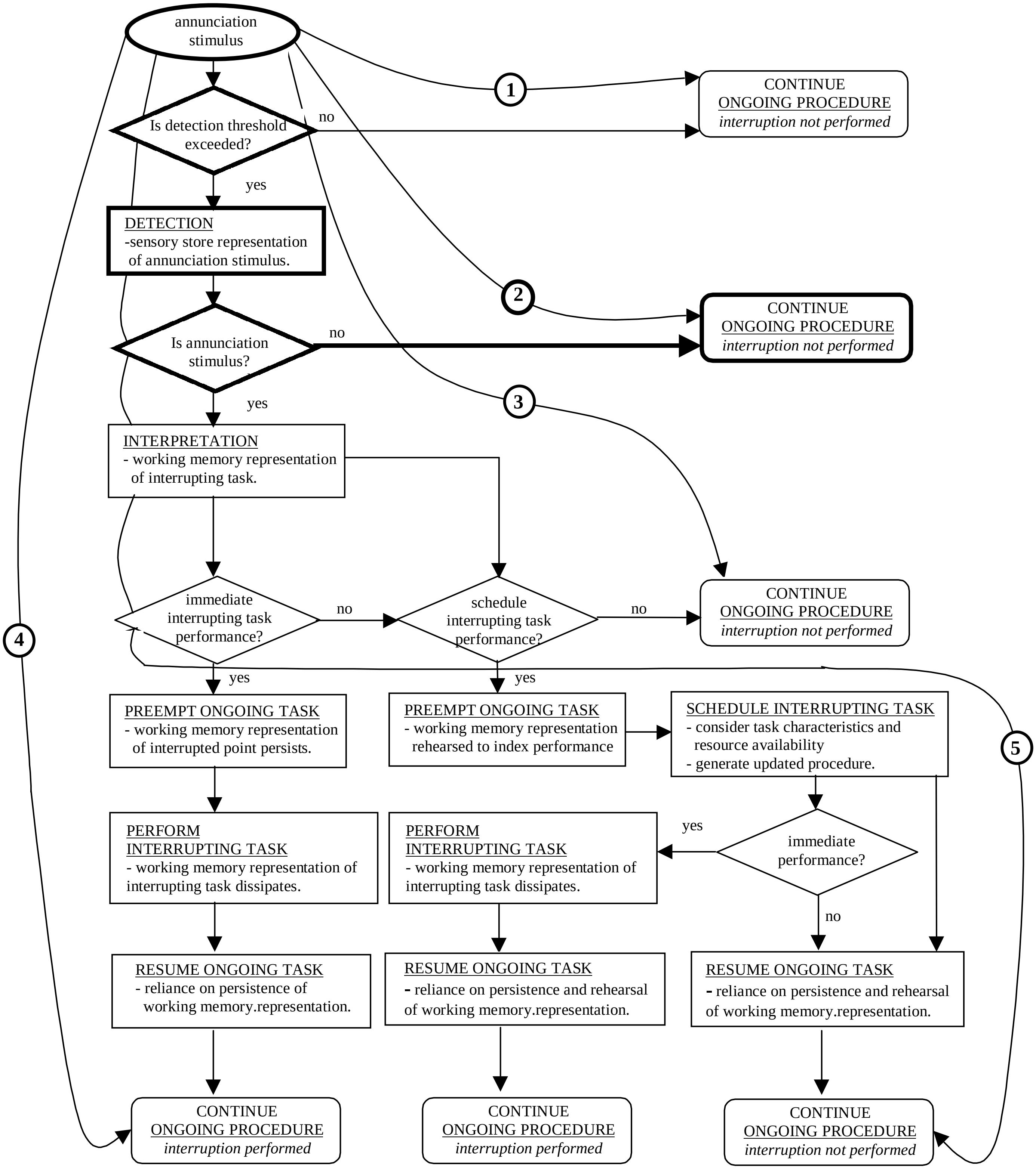


Figure 8.2. The Interruption Management Model & Unintentional Dismissal.

Integration and *Intentional Dismissal*

Given that the annunciation stimulus is correctly interpreted in terms of the interrupting task's performance requirements, the next stage requires integration of these additional performance requirements with those previously defined by the ongoing procedure. If, aware of the interruption's performance requirements, the operator elects to continue performing the ongoing task without performing or considering when to perform the interrupting task, the interruption is *intentionally dismissed* and the operator does not perform the interruption in this iteration of the model (path 3, Figure 8.3). The representation of the interruption in working memory gradually dissipates. Intentionally dismissed interruptions will not be re-addressed unless: (1) Continuation of the annunciation stimulus prompts reiteration of the interruption management process. (2) The working memory representation prompts performance at a later time. The likelihood of this recovery depends on the probability that the annunciation stimulus re-primes, and/or the ability of the operator to rehearse, and maintain activation of, the working memory representation.

Integration and *Preemptive Integration*

Alternatively, operators may integrate the interrupting task by immediately preempting the ongoing task to perform the interrupting task without considering the implications of performing it at that point. This is the *preemptive integration* behavior of interruption management (path 4, Figure 8.4). Upon preemption, available working memory representations associated with the ongoing task persist as an intention to rejoin this task. As the operator executes the interrupting task performance requirements, this information becomes most activated in working memory. After completing the interrupting task, it is not necessary to retain the interruption memory representation and, in the absence of rehearsal, it gradually dissipates. Continued presence of the interrupted task's working-memory representation prompts the operator to resume the interrupted task and continue the remainder of the ongoing procedure following performance of the interrupting task.

In the former description of *preemptive integration*, the interrupting task is completely performed before resuming the ongoing procedure. If one considers concurrent processes, interrupting task performance may itself be interrupted by other salient external stimuli, highly activated internal working memory representations, or additional annunciation stimuli. In this way, interruption integration, although initially preemptive, may also be opportunistically interleaved with ongoing procedure performance over several iterations.

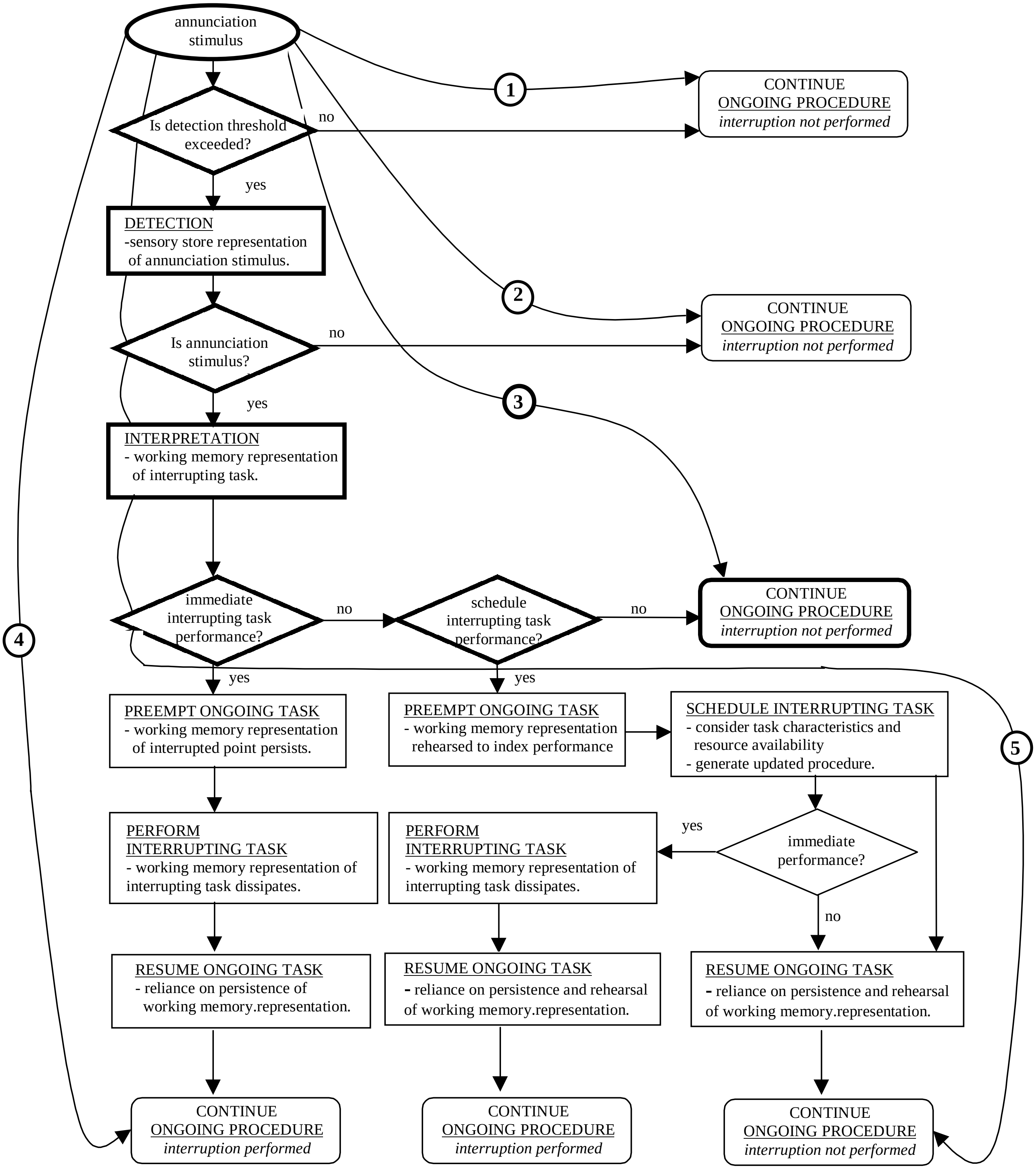


Figure 8.3. The Interruption Management Model & Intentional Dismissal

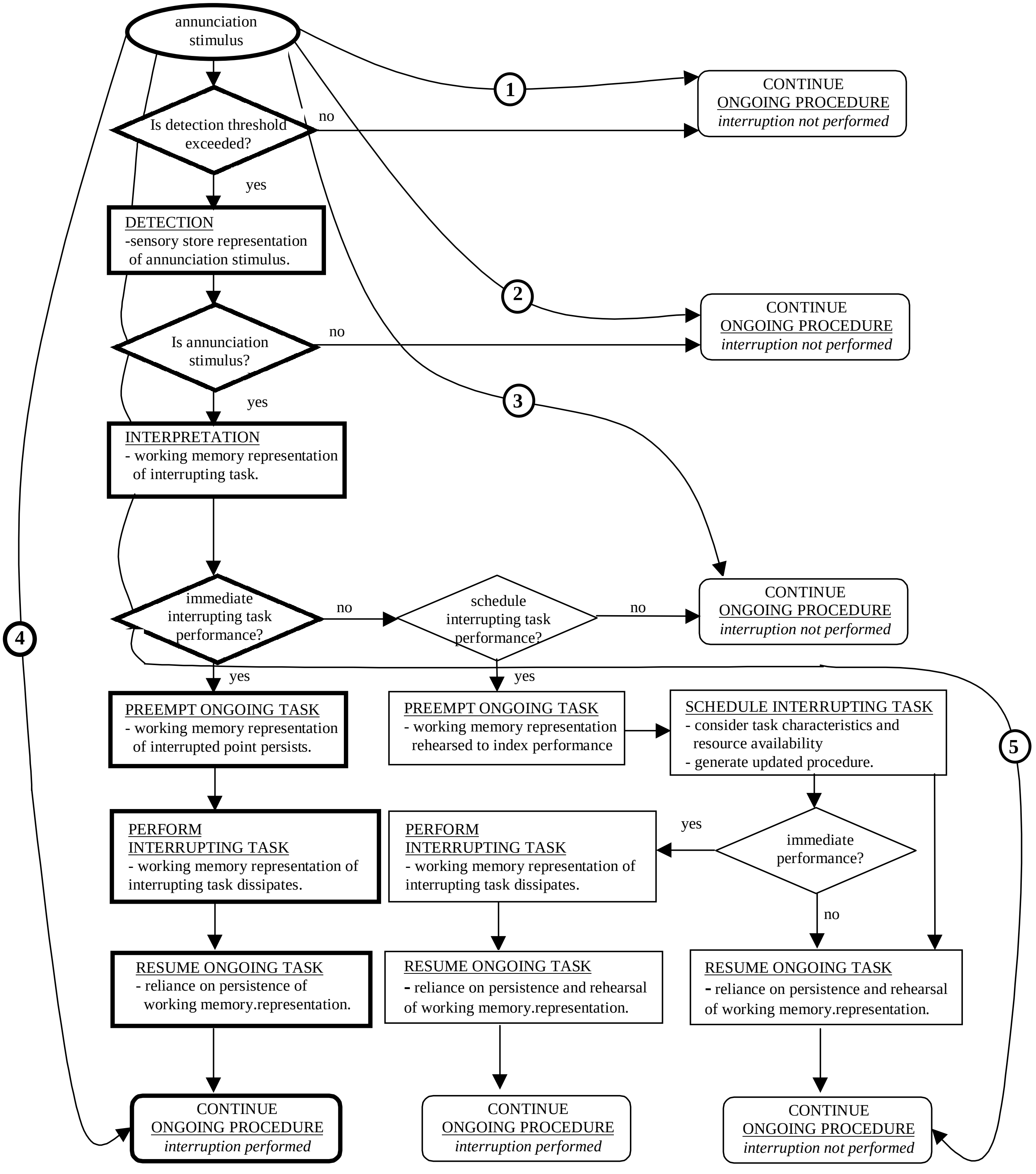


Figure 8.4. The Interruption Management Model & Preemptive Integration

Integration and *Intentional Integration*

In preemptive integration, the operator continues directly from interpreting the annunciation stimulus to performing the interrupting task, by preempting the ongoing procedure. Alternatively, the operator may actively consider when to integrate this new task, or may elect to not integrate the new task. *Intentional integration* interruption management involves explicit, intentional, strategic integration of the interruption into the ongoing procedure (path 5, Figure 8.5). Intentional integration is more likely than preemptive integration to the degree that the environment is predictable and controllable by the operator, and to the degree that the consequences of not explicitly considering task integration outweighs the effort required to determine this integration (Scholnick and Friedman 1993).

To consider this scheduling problem, the operator preempts the currently ongoing task. Upon preemption, available working memory representations associated with the ongoing task persist as an intention to rejoin this task after determining how to integrate the interruption. Normatively, this process involves rationally evaluating the interrupting task's resource requirements, projecting resource requirements of future procedural tasks', and considering task priority characteristics. However, due to imperfect information, under-specified objectives, and cognitive biases and limitation, the scheduling process is likely to be less optimal. Regardless of the generating process, the result of the scheduling stage is a revised procedure that includes the interrupting task.

After consideration, the operator may conclude that the interrupting task should be performed immediately. In this case, the sequence of overt actions will be identical to that of the operator who engages in preemptive integration. The distinctions between preemptive integration and this special case of intentional integration are presumably reflected in the relative speed of preemptive integration, and the relative optimality and cognitive load associated with intentional integration. Once performed or integrated into the future schedule, the working-memory representations associated with the interruption dissipate. Having completed the scheduling stage, the intention, or working-memory representations associated with the interrupted task prompts the operator to resume that task and continue performance of the newly developed schedule. The operator performs the interrupting task in the course of executing the revised procedure.

In the former description of intentional integration, the operator schedules the interrupting task as a complete unit in the ongoing procedure. Alternatively, the operator may parse performance requirements of the interrupting task and strategically schedule these components amid performance requirements of the remaining procedural tasks. This scheduling process is more computationally extensive.

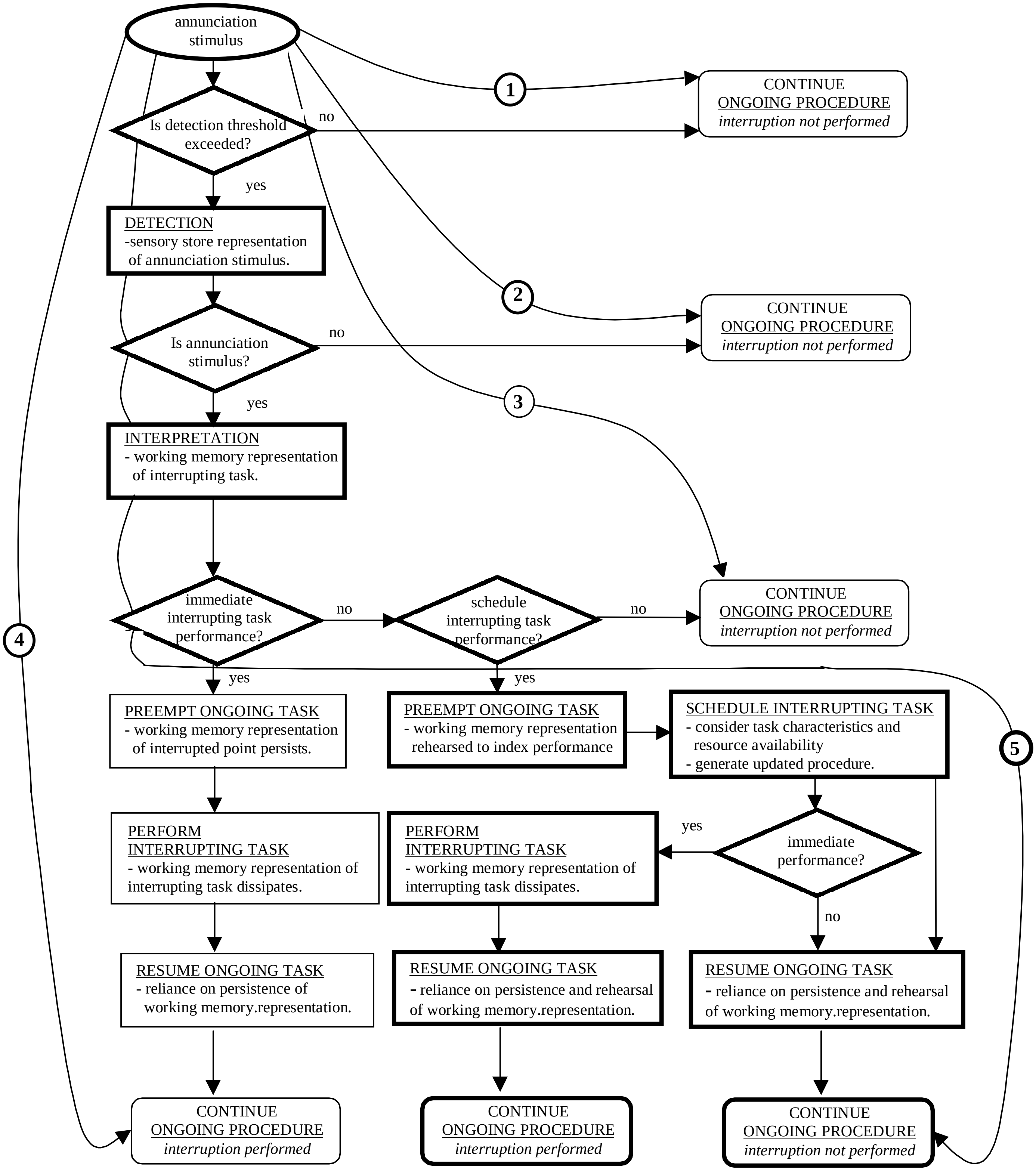


Figure 8.5. The Interruption Management Model & Intentional Integration

Interruption Management Behaviors in Terms of Interruption Effects

Interruption management behaviors can be described in terms of their potential effects on the ongoing procedure. Obliviously-dismissed interruptions do not divert the operator from the ongoing task, and therefore also are not capable of inducing disruptions. Unintentionally dismissed interruptions divert, but do not distract the operator from the ongoing task. In this case, disruptions can only propagate from diversion effects. Intentionally dismissed interruptions divert attention and distract the operator from the ongoing task. Propagation effects of both diversion and distraction may disrupt performance on the remainder of the ongoing procedure. Finally, disturbances to the ongoing procedure result from both Preemptive Integration and Intentional Integration. Detecting, interpreting and integrating an interruption into an ongoing procedure results in initial diversion and distraction from and ensuing disturbance to the procedure. Disruptions to the post-interruption procedure may propagate from any of these effects.

Implications for the Flightdeck

This research experimentally induced the deleterious effects of interruptions previously indicated in aviation incidents and accidents. Prior to this investigation, research on interruptions was limited to investigations of datalink implementations and a linguistic investigation of interrupted checklist performance. This research provides additional data on response and recovery times to datalink vs. radio interruptions. Conflicting evidence of interruption recovery times with previous datalink research (*i.e.*, Williams 1995) suggests differences between dedicated and task-shared implementations of datalink requiring further investigation. This research extends the investigation of modality effects associated with datalink implementations beyond traditional measures (*cf.* Kerns 1990) to include effects on performing the interrupting task and disruptive effects on post-interruption performance.

Previous basic research has indicated several task factors that affect interruption management performance, these were not experimentally manipulated in the flightdeck environment. Other research, investigating flightdeck performance, indicates that pilots respond to contextual information in selecting flightpath management modes and to manage their own workload. However, these lines of research have not been extended to consider the contextual factors which mediate interruption management on the flightdeck. This research extends basic research to the operational environment and demonstrates significant performance effects attributable to interrupting and interrupted task characteristics.

This research also demonstrates statistically significant, if not, in many cases, operationally profound, effects of realistic interruptions in a relatively realistic simulated commercial flightdeck. Even modest effects are noteworthy, as they show that even simple, well-practiced, routine, and, to some degree, expected interruptions reliably affect performance on the flightdeck. Further, because accidents typically result from an amalgam of, what would be in isolation minimally deleterious events, this research suggests that interruptions are likely to significantly contribute to performance degradation in more realistic, less rigid flightdeck operations. These results confirm empirically, then, what has been shown most dramatically in summaries of aviation incident reports (*e.g.*, Monan 1979; Barnes and Monan 1990; Turner and Huntley 1991) and aviation accident analyses (*e.g.*, Chou and Funk 1993; NTSB 1988); interruptions on the flightdeck pose an under-appreciated hazard to crew performance on the commercial flightdeck. While the present study provides empirical data and a theoretical framework towards understanding mechanisms and effects of interruptions on the flightdeck it is but a precursor to solving this problem.

Interruption Management Intervention on the Flightdeck

The benefit of understanding factors influencing interruption management is that it provides a foundation for identifying means by which to mitigate the deleterious effects of interruptions on the flightdeck. Previous research and results of this study may be extended to the design of interface features, intelligent aiding, and training programs for minimizing effects of interruptions on flightdeck performance.

Interfaces Features for Interruption Management

Kreifeldt and McCarthey (1981) propose *interruption resistance* as a specific interface design criterion. To this proposal, I add that interfaces should not necessarily always be interruption resistant, as sometimes interruptions are important, but should always be *interruption resilient*. Based on previous literature and the results of this research, one can postulate several interface features to reduce the deleterious effects of interruptions. First, the advantages of referenceable interrupting task information were evident in the modality results. Presenting ATC calls via datalink provides one solution to this problem, however it creates other concerns. In addition, auditory communication will likely be the primary means of communication among agents in the aviation system for some time. Flightdeck performance may be enhanced by providing a referenceable version of aurally-presented interrupting tasks. This could be accomplished in several ways. For example, a playback feature may provide pilots with the ability to rapidly confirm their interpretation of interrupting task annunciations. Additionally, if a datalink system is aboard, radio communications might, through speech recognition technology, be referenceable as a visual playback feature. Second, several studies have demonstrated the potential benefits of providing an externalized marker to the interrupted task. In particular, pilots interrupted in the middle of a checklist frequently mark the interruption point. Thus, interruption positions should be externally indicated. However, the degree to which markers of the interrupted task are useful depends on the degree to which the ongoing task set is proceduralized. Theoretically, interruptions to inflexible task sets should be more destructive than interruptions to procedural task sets (*e.g.*, Adams, Tenney, and Pew 1995). In this situation, interfaces could provide historical information of tasks performed to improve interruption resiliency.

Intelligent Aiding for Interruption Management

Intelligent aiding approaches to interruption management are distinguished from interface features by their reliance on some intelligence about those characteristics of the interrupting and interrupted tasks and task set, the environment, and the operator which influence interruption management. Intelligent interruption aiding is a subset of the larger issue of supporting multiple task management on the flightdeck. Several forms of task management aids have been developed for the flightdeck (Funk and Linde 1992), primarily focusing on the coordination of multiple ongoing tasks. While the more sophisticated of these approaches include mechanisms for detecting interruptions and providing pilots with useful resumption prompts, they do not attempt to more sensitively introduce interruptions to the ongoing task set. The sensitive introduction of interruptions requires another level of aiding.

An interruption integration aid (IIA) would serve as a protective membrane surrounding ongoing flightdeck tasks. The instantaneous permeability of this membrane to interruptions would be defined by a model of task, operator, and environment characteristics found to influence interruption management performance. An IIA would actively manage the introduction of interruptions in two ways: (1) by determining when an interruption should intervene on flightdeck performance, and (2) by determining how an interruption intervenes on flightdeck performance, *i.e.*, by defining the characteristics of the annunciation stimulus. While much more research is required to fully determine the interaction of pertinent characteristics, this research provides some preliminary suggestions. For example, if temporal constraints are not a concern, an IIA can reduce the potential for interruption and procedural performance errors by not allowing an interruption to occur between physically-proximal adjacent procedural tasks. Timesharing research would provide a foundation for determining how an IIA might reformulate annunciation stimulus modality to minimize interference with ongoing tasks.

Training for Interruption Management

Previous research finds that subjects in often-interrupted task sets eventually adopt mechanisms that reduce the deleterious effects of interruptions (Hess and Detweiler 1994). However, research on the flightdeck suggests that pilots do not adhere to rules dictating behavior in response to interrupting tasks during checklist performance (Linde and Goguen 1987). The factors enabling acclimation to interruptions and the circumstances which compel subjects to contradict explicit interrupt-handling instructions require more extensive examination to determine the efficacy of training operators for interrupted task management. Training could not only take to form of defining conditions under which to avoid interruption, but might extend to training pilots strategies for ensuring resumption of interrupted tasks.

Finally, training for interruption management does not extend only to the individual pilot but may have implications for Cockpit Resource Management (CRM) or training a crew to manage interruptions most capably. Pilots appear to adhere to pre-defined roles to determine who should respond to interruptions (Williams 1995). However, at times the entire crew can become engaged in addressing an interrupting task. Prior research indicates that resumption time is longest when both crew members are involved in an interruption (Williams 1995). Further, this whole-crew preoccupation with an interrupting task has lead to disregard for other aviation tasks and resulted in at least one disastrous accident (NTSB 1973). Recognizing what task, environment, and personality characteristics most predispose a pilot to be interrupted serve as guidelines for CRM compensatory techniques.

Directions for Future Work

Research explicitly directed to studying interruption management is relatively scarce. Therefore many opportunities for advancement exist. This research presents a basic stage model to describe interruption management. This model requires a great deal of embellishment to become a predictive instrument. One effort could begin with cataloging potentially-important factors associated with detection, interpretation and working memory representations, task switching, and human scheduling. The model would benefit from controlled, laboratory investigations of these factors in an interruption management paradigm and would suggest factors for experimentation in more realistic simulation experiments. The study of interruption management would also benefit tremendously from a more analytical approach to field studies. This might begin by using the interruption management behaviors derived in this research as a classification scheme for observed behavior and the interruption management model as a classification scheme for noting important situational characteristics. Simulation research should continue to bridge the gap between the laboratory and the field by demonstrating operational viability of factors identified by theory or laboratory experiments, and by demonstrating the robustness of and mechanisms behind field observations.

Results of the present simulation experiment suggest, in particular, several research issues. How are quantitative measures of interruption management performance associated with interruption management behaviors? Under what task conditions do operators tend to exhibit the various interruption management behaviors? What is the role of individual factors (processing capacities, perceptual biases, motivational characteristics) in interruption management? How does interruption management change in multi-agent situations? Finally, more applied research is required to develop and test prototype interruption resilient interfaces, interruption aiding devices, and interruption handling training regimes.

Appendix

Appendix 5.1

Summary of Subject Characteristics.

Characteristics of Test Subjects.

Subject	Age	Seat	Years Commercial	Years Military	Total Hours	Hours in Command	Current Aircraft	700 Series
3	48	C	21	5	16000	6000	757	3
4	35	FO	4	12	5000	2500	767	3
5	51	C	22	9	12000	8000	757/767	3
6	38	FO	*	*	5000	1500	767/757	1
7	54	C	25	5	20000	10000	767	2
8	52	C	23	6	25000	12000	757/767	2
9	35	FO	12	0	7500	1373	767	2
10	37	C	11	0	15500	7000	737-300/400	3
11	53	FO	8	21	10000	4200	767	2
12	42	FO	15	18	16000	8000	747-400	2
13	38	FO	6	10	5000	2000	767	2
14	49	C	17	0	10000	8000	767/757	4
15	56	C	30	6	17000	9000	767	3
16	56	C	28	28	20000	12000	767	4

C = Captain, FO = First Officer, * data not provided

Appendix 5.2

TSRV-IIC Simulation Code[16]

---

[16] Simulation specifications were programmed for the TSRV by Mrs. Wei Anderson and Mrs. Arlene Guenther of the Unisys Simulation, Programming, and Analysis group at NASA Langley.

```
*deck w1cond.f
      SUBROUTINE W1COND
*
** perform window 1 interruption task
*
*call cncdkey.com
*call console.com
*call cvoice.com
*call realtim.com
*call cparam.com
*call cradio.com
*call dataout.com
*call dnlink.com
*call iicdisc.com
*call iicvar.com
*call intcomm.com
*call setup.com
*call sysvar.com
*call trigger.com
      if (runcond .eq. '11.01' .or. runcond .eq. '21.01') then
        if (ip2dn .eq. 1 .and. dtogonm .le. dgonm1) then
          itrun = 1
          ittyp = 1
          if (runcond .eq. '11.01') then
            ivd = 1
            ivdata(ivd) = 20   ! run# 11
            iicflg1 = iicflg1 + 2**0
          else
            iicflg1 = iicflg1 + 2**11
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.02' .or. runcond .eq. '21.02') then
        if (abs(dtk).le.5. .and. tphideg.ge.timphi .and. tpast1.gt.0. .or.
     *      tpast1 .ge. 18.) then
          itrun = 1
          ittyp = 1
          ivd = 1
          ivdata(ivd) = 24      ! run# 19,27
          if (runcond .eq. '11.02') then
            iicflg1 = iicflg1 + 2**1
          else
            iicflg1 = iicflg1 + 2**12
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.03' .or. runcond .eq. '21.03') then
        if (comprc .eq. compfc ) then
          itrun = 1
          ittyp = 1
          ivd = 1
          ivdata(ivd) = 22      ! run# 24,22,30/22,30
          if (runcond .eq. '11.03') then
            iicflg1 = iicflg1 + 2**2
          else
            iicflg1 = iicflg1 + 2**13
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.04' .or. runcond .eq. '21.04') then
        if (atisrc .eq. atisfc) then
          itrun = 1
          ittyp = 1
```

```
          if (runcond .eq. '11.04') then
            ivd = 1
            ivdata(ivd) = 20   ! run# 13
            iicflg1 = iicflg1 + 2**3
          else
            iicflg1 = iicflg1 + 2**14
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.05' .or. runcond .eq. '21.05') then
        if (tatis .ge. 5.) then
          itrun = 1
          ittyp = 1
          ivd = 1
          ivdata(ivd) = 29      ! run# 18,26
          if (runcond .eq. '11.05') then
            iicflg1 = iicflg1 + 2**4
          else
            iicflg1 = iicflg1 + 2**15
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.06' .or. runcond .eq. '21.06') then
        if (tatis .ge. 5.) then
          itrun = 1
          ittyp = 2
          ivd = 1
          ivdata(ivd) = 1      ! run# 20,28
          ixmenu(2) = 1
          dpi(2) = mmsg
          if (runcond .eq. '11.06') then
            idnmsg = 5
            iicflg1 = iicflg1 + 2**5
          else
            idnmsg = 3
            iicflg1 = iicflg1 + 2**16
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.07' .or. runcond .eq. '21.07') then
        if (towrrc .eq. towrfc) then
          itrun = 1
          ittyp = 2
          ixmenu(2) = 1
          dpi(2) = mmsg
          idnmsg = 1
          if (runcond .eq. '11.07') then
            ivd = 1
            ivdata(ivd) = 1   ! run# 12
            iicflg1 = iicflg1 + 2**6
          else
            iicflg1 = iicflg1 + 2**17
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.08' .or. runcond .eq. '21.08') then
        if (wincdu .eq. initflc(1) .and. wincdu .ne. wincdup) then
          itrun = 1
          ittyp = 2
          ivd = 1
          ivdata(ivd) = 1      ! run# 16,24,32/24,32
          ixmenu(2) = 1
          dpi(2) = mmsg
          if (runcond .eq. '11.08') then
```

```
            idnmsg = 2
            iicflg1 = iicflg1 + 2**7
          else
            idnmsg = 4
            iicflg1 = iicflg1 + 2**18
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.09' .or. runcond .eq. '21.09') then
        if (wincdu .eq. initflc(1) .and. wincdu .ne. wincdup) then
          itrun = 1
          ittyp = 1
          ivd = 1
          ivdata(ivd) = 22      ! run# 17,25
          if (runcond .eq. '11.09') then
            iicflg1 = iicflg1 + 2**8
          else
            iicflg1 = iicflg1 + 2**19
          endif
          wevt1 = .true.
        endif
      else if (runcond .eq. '11.10' .or. runcond .eq. '21.10') then
*
** elapse time since 1R (STATUS>) of init/ref page was pushed
*
        if (tinit1r .gt. 0.) then
          tinit1r = tinit1r + h
*
** start timer when 1R (STATUS) was selected while on init/ref page
*
        else if (wincdu .eq. initflc(1) .and. keycdu .eq. 6) then
          tinit1r = h
        endif
*
        if (wincdup .eq. initflc(8) .and. wincdu .ne. wincdup .or.
     *      tinit1r .ge. 30.) then
          itrun = 1
          ittyp = 1
          ivd = 1
          if (runcond .eq. '11.10') then
            ivdata(ivd) = 24
            iicflg1 = iicflg1 + 2**9
          else
            ivdata(ivd) = 28
            iicflg1 = iicflg1 + 2**20
          endif
          wevt1 = .true.
        endif
      else if ((runcond .eq. '11.11' .or. runcond .eq. '11.12' .or.
     *         runcond .eq. '21.11' .or. runcond .eq. '21.12') .and.
     *        .not. nevt1 .and. ip2dn .eq. 1 .and. dtogonm .le. 4.8) then
        iicflg1 = iicflg1 + 2**10
        nevt1 = .true.
      endif
*
      return
      end
*deck w2cond.f
      SUBROUTINE W2COND
*
** perform window 2 interruption task
*
*call cncdkey.com
*call console.com
*call cvoice.com
*call realtim.com
*call cparam.com
*call cradio.com
*call dataout.com
*call iicdisc.com
*call intcomm.com
*call setup.com
*call trigger.com
      if (rcond2 .eq. '12.01' .or. rcond2 .eq. '22.01') then
        if (tpast2 .ge. 45.) then
          ittyp = 1
          if (rcond2 .eq. '12.01') then
            itrun = 2
            iicflg1 = iicflg1 + 2**21
          else
            itrun = 3
            iicflg1 = iicflg1 + 2**32
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.02' .or. runcond .eq. '22.02') then
        if (abs(dtk).le.5. .and. tphideg.ge.timphi .and. tpast3.gt.0. .or.
     *      tpast3 .ge. 38.) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '12.02') then
            itrun = 2
            ivdata(ivd) = 9   ! run# 16,24,32
            iicflg1 = iicflg1 + 2**22
          else
            itrun = 3
            ivdata(ivd) = 4   ! run# 16,24
            iicflg1 = iicflg1 + 2**33
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.03' .or. runcond .eq. '22.03') then
        if (wincdu .eq. initflc(1) .and. keycdu .eq. 13) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '12.03') then
            itrun = 4
            ivdata(ivd) = 2   ! run# 23,31
            iicflg1 = iicflg1 + 2**23
          else
            itrun = 5
            ivdata(ivd) = 3   ! run# 15,23,31
            iicflg1 = iicflg1 + 2**34
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.04' .or. runcond .eq. '22.04') then
        if (wincdu .eq. initflc(1) .and. keycdu .eq. 13) then
          ittyp = 1
          if (runcond .eq. '12.04') then
            itrun = 2
            iicflg1 = iicflg1 + 2**24
          else
            itrun = 3
            ivd = 1
            ivdata(ivd) = 4   ! run# 12
            iicflg1 = iicflg1 + 2**35
          endif
          wevt2 = .true.
        endif
```

```
      else if (runcond .eq. '12.05' .or. runcond .eq. '22.05') then
*
** elapse time since 3R (BAR) of perf/init page was pushed
*
        if (tperf3r .gt. 0.) then
          tperf3r = tperf3r + h
*
** start timer when 3R (BAR) was selected while on perf/init page
*
        else if (wincdu .eq. initflc(4) .and. keycdu .eq. 14) then
          tperf3r = h
        endif
*
        if (wincdup .eq. initflc(4) .and. wincdu .ne. wincdup .or.
     *      tperf3r .ge. 3.) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '12.05') then
            itrun = 2
            ivdata(ivd) = 12   ! run# 20,28
            iicflg1 = iicflg1 + 2**25
          else
            itrun = 3
            ivdata(ivd) = 6    ! run# 20,28
            iicflg1 = iicflg1 + 2**36
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.06' .or. runcond .eq. '22.06') then
        if (seatblt) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '12.06') then
            itrun = 2
            ivdata(ivd) = 9   ! run# 19,27
            iicflg1 = iicflg1 + 2**26
          else
            itrun = 3
            ivdata(ivd) = 4   ! run# 19,27
            iicflg1 = iicflg1 + 2**37
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.07' .or. runcond .eq. '22.07') then
        if (landlgt) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '12.07') then
            itrun = 2
            ivdata(ivd) = 12   ! run# 18,26
            iicflg1 = iicflg1 + 2**27
          else
            itrun = 3
            ivdata(ivd) = 6    ! run# 18,26
            iicflg1 = iicflg1 + 2**38
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.08' .or. runcond .eq. '22.08') then
        if (ixautb .ne. 4) then
          ittyp = 1
          itrun = 3
          if (runcond .eq. '12.08') then
            ivd = 1
            ivdata(ivd) = 4   ! run# 13
```

```
            iicflg1 = iicflg1 + 2**28
          else
            iicflg1 = iicflg1 + 2**39
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.09' .or. runcond .eq. '22.09') then
        if (ixmenu(1) .eq. 16) then
          ittyp = 1
          if (runcond .eq. '12.09') then
            itrun = 2
            ivd = 1
            ivdata(ivd) = 8   ! run# 11
            iicflg1 = iicflg1 + 2**29
          else
            itrun = 3
            ivdata(ivd) = 4
            iicflg1 = iicflg1 + 2**40
          endif
          wevt2 = .true.
        endif
      else if (runcond .eq. '12.10' .or. runcond .eq. '22.10') then
        if (ixmenu(1) .eq. 31 .or.
     *      ixmenu(1) .eq. 39 .and. nxmenu(1) .eq. 14) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '12.10') then
            itrun = 2
            ivdata(ivd) = 11   ! run# 22,30
            iicflg1 = iicflg1 + 2**30
          else
            itrun = 3
            ivdata(ivd) = 6    ! run# 14,22,30
            iicflg1 = iicflg1 + 2**41
          endif
          wevt2 = .true.
        endif
      else if ((runcond .eq. '12.11' .or. runcond .eq. '12.12' .or.
     *         runcond .eq. '22.11' .or. runcond .eq. '22.12') .and.
     *        .not. nevt2 .and. ip2dn .eq. 3 .and. dtogonm .le. 3.0) then
        iicflg1 = iicflg1 + 2**31
        nevt2 = .true.
      endif
*
      return
      end
*deck w3cond.f
      SUBROUTINE W3COND
*
** perform window 3 interruption task
*
*call cadcs.com
*call cncdkey.com
*call console.com
*call cvoice.com
*call realtim.com
*call cparam.com
*call cradio.com
*call dataout.com
*call iicdisc.com
*call intcomm.com
*call setup.com
*call trigger.com
      if (runcond .eq. '13.01' .or. runcond .eq. '23.01') then
        if (ip2dn .eq. 6 .and. dtogonm .le. dgonm3) then
```

```
          ittyp = 1
          if (runcond .eq. '13.01') then
            itrun = 2
            iicflg1 = iicflg1 + 2**42
          else
            itrun = 3
            iicflg1 = iicflg1 + 2**53
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.02' .or. runcond .eq. '23.02') then
        if (abs(dtk).le.5. .and. tphideg.ge.timphi .and. tpast6.gt.0. .or.
     *      tpast6 .ge. 17.) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '13.02') then
            itrun = 2
            ivdata(ivd) = 12   ! run# 18,26
            iicflg1 = iicflg1 + 2**43
          else
            itrun = 3
            ivdata(ivd) = 6    ! run# 18,26
            iicflg1 = iicflg1 + 2**54
          endif
          wevt3 = .true.
        endif
      else if (runcond.eq.'13.03' .or. runcond.eq.'23.03') then
        if (wincdu .eq. n1limit .and. wincdu .ne. wincdup) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '13.03') then
            itrun = 4
            ivdata(ivd) = 2   ! run# 16,24,32
            iicflg1 = iicflg1 + 2**44
          else
            itrun = 5
            ivdata(ivd) = 3   ! run# 24,32
            iicflg1 = iicflg1 + 2**55
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.04' .or. runcond .eq. '23.04') then
        if (wincdu .eq. n1limit .and. wincdu .ne. wincdup) then
          ittyp = 1
          if (runcond .eq. '13.04') then
            itrun = 2
            iicflg1 = iicflg1 + 2**45
          else
            itrun = 5
            ivd = 1
            ivdata(ivd) = 3   ! run# 11
            iicflg1 = iicflg1 + 2**56
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.05' .or. runcond .eq. '23.05') then
*
** elapse time since 1L (GA) on n1limit page is pushed
*
        if (tlmt1l .gt. 0.) then
          tlmt1l = tlmt1l + h
*
** start timer when 1L (GA) was depressed while on n1limit page
*
        else if (wincdu .eq. n1limit .and. keycdu .eq. 5) then
```

```
          tlmt1l = h
        endif
*
        if (wincdup .eq. n1limit .and. wincdu .ne. wincdup .or.
     *      tlmt1l .ge. 3.) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '13.05') then
            itrun = 2
            ivdata(ivd) = 11   ! run# 17,25
            iicflg1 = iicflg1 + 2**46
          else
            itrun = 3
            ivdata(ivd) = 6    ! run# 17,25
            iicflg1 = iicflg1 + 2**57
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.06' .or. runcond .eq. '23.06') then
        if (nosmoke) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '13.06') then
            itrun = 2
            ivdata(ivd) = 18   ! run# 14,22,30
            iicflg1 = iicflg1 + 2**47
          else
            itrun = 3
            ivdata(ivd) = 7    ! run# 22,30
            iicflg1 = iicflg1 + 2**58
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.07' .or. runcond .eq. '23.07') then
        if (ixspb .ne. 0 .and. gearc .eq. 1) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '13.07') then
            itrun = 2
            ivdata(ivd) = 15   ! run# 20,28
            iicflg1 = iicflg1 + 2**48
          else
            itrun = 3
            ivdata(ivd) = 7    ! run# 20,28
            iicflg1 = iicflg1 + 2**59
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.08' .or. runcond .eq. '23.08') then
        if (flapc .eq. 25.) then
          ittyp = 1
          if (runcond .eq. '13.08') then
            itrun = 2
            ivd = 1
            ivdata(ivd) = 13   ! run# 12
            iicflg1 = iicflg1 + 2**49
          else
            itrun = 3
            iicflg1 = iicflg1 + 2**60
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.09' .or. runcond .eq. '23.09') then
        if (ixmenu(1) .eq. 17) then
          ittyp = 1
```

```
          if (runcond .eq. '13.09') then
            itrun = 2
            iicflg1 = iicflg1 + 2**50
          else
            itrun = 5
            ivd = 1
            ivdata(ivd) = 2   ! run# 13
            iicflg1 = iicflg1 + 2**61
          endif
          wevt3 = .true.
        endif
      else if (runcond .eq. '13.10' .or. runcond .eq. '23.10') then
        if (ixmenu(1).eq.32 .or.
     *      ixmenu(1).eq.39 .and. nxmenu(1).eq.15) then
          ittyp = 1
          ivd = 1
          if (runcond .eq. '13.10') then
            itrun = 2
            ivdata(ivd) = 16   ! run# 15,23,31
            iicflg1 = iicflg1 + 2**51
          else
            itrun = 3
            ivdata(ivd) = 5    ! run# 23,31
            iicflg1 = iicflg1 + 2**62
          endif
          wevt3 = .true.
        endif
      else if ((runcond .eq. '13.11' .or. runcond .eq. '13.12' .or.
     *         runcond .eq. '23.11' .or. runcond .eq. '23.12') .and.
     *        .not. nevt3 .and. ip2dn .eq. 6 .and. dtogonm .le. 2.5) then
        iicflg1 = iicflg1 + 2**52
        nevt3 = .true.
      endif
*
      return
      end
*deck procevt.f
      SUBROUTINE PROCEVT
*
** This routine is the proprocess the *.evt file
*
*call cadcs.com
*call cguid.com
*call cinput.com
*call cmiscel.com
*call cncdkey.com
*call console.com
*call graphx.com
*call intcomm.com
*call lguid.com
*call pages.com
*call realtim.com
*call sysvar.com
*call cparam.com
*call cradio.com
*call dataout.com
*call dparam.com
*call iicdisc.com
*call setup.com
      parameter ( itfr1=1,   itfr2=2,   ifrq1=3,   ifrq2=4,   ilisten=5,
     * itrns=6,   ikeyon=7,  ikeyoff=8, itchbeg=9, itchend=46,itchrtn=47,
     * ifms=48,   ifmsp=49,  isbelt=50, ismoke=51, illgt=52,  iskid=53,
     * iabrk=54,  igear=55,  iflap=56,  ispb=57,   ithrc=58,  istickp=59,
     * istickr=60,itmarka=61,itmarkb=62,ipwpt=63,
ialtbug=64,ispdbug=65,
```

```
     * i2miles=66,itidle=67, ialtrst=68,ispdrst=69,ieventm=70,ialtdev=71,
     * ispddev=72,ihddev=73, iack=74,
imodelp=75,imodelt=76,ilink=77,
     * ialtlev=78,ilatdev=79)
      parameter  ( nline = 80, nchar = 80 )
      integer      itext(nline), imenu1p, imenu2p, ip2dnp, ncdcvt(80), ixspbp
      character*2  thrc, thrcp
      character*3  autbsw(4),itwinc(3),ittypc(2),iasc,iascp
      character*5  altc, altcp, clevoff
      character*7  itrunc(5)
      character*8  apprrcp,towrrcp,comprcp,comprc0,atisrcp,cdukys(79)
      character*10 service(4),talk,talkp
      character*12 ptrmark(7)
      character*16 touchm
      character*24 audio, audiop
      character*80 text(nline), textln, chklist(20), msglist(3)
      logical      eventmp, tfr1p, tfr2p, pmikep, cmikep, passflg
      logical      puodp, pdodp, rlodp,  rrodp
      logical      wevt1p, wevt2p, wevt3p
      logical      todbeg, todend, k18beg, k18end, fafbeg, fafend
      logical      tdeparr, farr, trthpat, frthpat, leg1nm1, leg1nm6, leg1exc
      logical      irunack, irunbeg, irunend, nrunack, nrunbeg, nrunend
      logical      hpatack, hpatbeg, hpatend, achgack, achgbeg, achgend
      logical      schgack, schgbeg, schgend
      logical      ivisack, ivisbeg, ivisend
      logical      sbeltp, nsmkp, llgtp, skdp, gearcp, sackp
      integer*1    jpagel(14*3*8), mpagel(14*3*8)
      character*24 mpagec(14)
      equivalence (jpagel(1), pagel(1))
      equivalence (mpagel(1), mpagec(1))
*
      data ntrate  / '***Data_32samples/second***' /
      data itwinc  / 'TOD', 'K18', 'FAF' /
      data itrunc  / 'INITRUN', 'NEWRUN ', 'HOLDPAT', 'CHGALT ',
'CHGSPD ' /
      data ittypc  / 'AUD', 'VIS' /
      data autbsw  / 'MIN', 'MED', 'MAX', 'OFF' /
      data service / 'APPROACH  ', 'TOWER     ', 'COMPANY   ', 'ATIS
' /
      data ptrmark / 'PROC-TOD    ', 'NON-PROC-HI ', 'PROC-18K    ',
     *              'NON-PROC-LO1', 'NON-PROC-LO2', 'PROC-FAF    ',
     *              'NON-PROC-APP' /
      data touchm  / '<CHECKLIST>:FROM' /
*
** ixmenu(1): denotes touch screen page transition
*
* 0 = blank
*
* 1 = <CHECKLIST>:FROM(MM)-To(Pre-Flight&Taxi-Out-Menu)
* 2 = <CHECKLIST>:FROM(MM)-To(Take-Off&Climb-Menu)
* 3 = <CHECKLIST>:FROM(MM)-To(Cruise(empty))
* 4 = <CHECKLIST>:FROM(MM)-To(Approach&Descent-Menu)
* 5 = <CHECKLIST>:FROM(MM)-To(Landing&Taxi-In-Menu)
*
* 6 = <CHECKLIST>:FROM(Pre-Flight&Taxi-Out-Menu)-To(MM)
* 7 = <CHECKLIST>:FROM(Pre-Flight&Taxi-Out-Menu)-To(Cockpit-
Prep-Cklst(empty))
* 8 = <CHECKLIST>:FROM(Pre-Flight&Taxi-Out-Menu)-To(Engine-
Start-Cklst(empty))
* 9 = <CHECKLIST>:FROM(Pre-Flight&Taxi-Out-Menu)-To(After-
Start-Cklst(empty))
*10 = <CHECKLIST>:FROM(Pre-Flight&Taxi-Out-Menu)-To(Taxi-
Out-Cklst(empty))
*
*11 = <CHECKLIST>:FROM(Take-Off&Climb-Menu)-To(MM)
```

```
*12 = <CHECKLIST>:FROM(Take-Off&Climb-Menu)-To(Before-
Take-Off-Cklst(empty))
*13 = <CHECKLIST>:FROM(Take-Off&Climb-Menu)-To(After--
Take-Off-Cklst(empty))
*14 = <CHECKLIST>:FROM(Take-Off&Climb-Menu)-To(Climb-
Cklst(empty))
*
*15 = <CHECKLIST>:FROM(Approach&Descent-Menu)-To(MM)
*16 = <CHECKLIST>:FROM(Approach&Descent-Menu)-
To(Approach-Cklst)
*17 = <CHECKLIST>:FROM(Approach&Descent-Menu)-To(Descent-
Cklst)
*18 = <CHECKLIST>:FROM(Approach&Descent-Menu)-To(Go-
Around-Cklst)
*
*19 = <CHECKLIST>:FROM(Landing&Taxi-In-Menu)-To(MM)
*20 = <CHECKLIST>:FROM(Landing&Taxi-In-Menu)-To(Taxi-In-
Cklst(empty))
*21 = <CHECKLIST>:FROM(Landing&Taxi-In-Menu)-To(Parking-
Cklst(empty))
*22
*23
*
*24 = <CHECKLIST>:FROM(Cockpit-Prep-Cklst(empty))-To(MM)
*25 = <CHECKLIST>:FROM(Engine-Start-Cklst(empty))-To(MM)
*26 = <CHECKLIST>:FROM(After-Start-Cklst(empty))-To(MM)
*27 = <CHECKLIST>:FROM(Taxi-Out-Cklst(empty))-To(MM)
*
*28 = <CHECKLIST>:FROM(Take-Off-Cklst(empty))-To(MM)
*29 = <CHECKLIST>:FROM(Rejected-Take-Off-Cklst(empty))-
To(MM)
*30 = <CHECKLIST>:FROM(Climb-Cklst(empty))-To(MM)
*
*31 = <CHECKLIST>:FROM(Approach-Cklst)-To(MM)
*32 = <CHECKLIST>:FROM(Descent-Cklst)-To(MM)
*33 = <CHECKLIST>:FROM(Go-Around-Cklst)-To(MM)
*
*34 = <CHECKLIST>:FROM(Landing-Cklst)-To(MM)
*35 = <CHECKLIST>:FROM(Landing-Roll-Cklst(empty))-To(MM)
*36 = <CHECKLIST>:FROM(Taxi-In-Cklst(empty))-To(MM)
*37 = <CHECKLIST>:FROM(Parking-Cklst(empty))-To(MM)
*
*38 = <CHECKLIST>:FROM(Cruise(empty))-To(MM)
*
*39 = TIME-OUT-FROM=(curent menu)-TO=(MM)
*
** ixmenu(2): denotes touch screen page transition
*
* 0 = blank
*
* 1 = FROM=(MM)-TO=(MESSAGE)
* 2 = FROM=(MESSAGE)-TO=(MM-ROGER)
* 3 = FROM=(MESSAGE)-TO=(MM-UNABLE)
*
      data chklist /
    *'<CHECKLIST>:TIME-OUT-FROM=(Main-Menu)-TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Pre-Flight&Taxi-Out-
Menu)-TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Take-Off&Climb-Menu)-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Cruise(empty))-TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Approach&Descent-Menu)-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Landing&Taxi-In-Menu)-
TO=(MM)',
```

```
    *'<CHECKLIST>:TIME-OUT-FROM=(Cockpit-Prep-Cklst(empty))-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Engine-Start-Cklst(empty))-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(After-Start-Cklst(empty))-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Taxi-Out-Cklst(empty))-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Before-Take-Off-
Cklst(empty))-TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(After-Take-Off-
Cklst(empty))-TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Climb-Cklst(empty))-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Approach-Cklst)-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Descent-Cklst)-TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Go-Around-Cklst)-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Landing-Cklst)-TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Landing-Roll-Cklst(empty))-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Taxi-In-Cklst(empty))-
TO=(MM)',
    *'<CHECKLIST>:TIME-OUT-FROM=(Parking-Cklst(empty))-
TO=(MM)' /
      data msglist / '<DATALINK>:FROM=(MM)-TO=(MESSAGE)',
    *             '<DATALINK>:FROM=(MESSAGE)-TO=(MM-
ROGER)',
    *             '<DATALINK>:FROM=(MESSAGE)-TO=(MM-
UNABLE)' /
      data cdukys /
    *      '--', '--', '--', '2L', '1L', '1R', '2R', '1',   '4',
    * '7',  ' .', '4L', '3L', '3R', '4R', '2',  '5',  '8',   '0',
    * '6L', '5L', '5R', '6R', '3',  '6',  '9',  '+/-','NXT PAGE', 'FIX',
    * 'DIR INTC','INIT REF','K', 'P', 'U','Z', 'F', 'A', 'LEGS', 'RTE',
    * 'L',  'Q',  'V',  'BLK','G',  'B',  'DEP ARR','CLB','M',   'R',
    * 'W',  'DEL','H',  'C',  'HOLD','CRZ','N',  'S',  'X',   '/',
    * 'I',  'D',  'PRG','DES','0',  'T',  'Y',  '--', 'J',   'E',
    * 'EXC','--', 'CL1','PRV PAGE','N1 LIMIT','CL2','--','--','--','--' /
*
** NCDU HEX          0,   1,   2,   3,   4,   5,   6,   7,   8,   9,
** WANTED SYMBOL     --,  --,  --,  --,  2L,  1L,  1R,  2R,  #1,  #4,
      DATA NCDCVT /  00,  00,  00,  00,  02,  01,  07,  08,  49,  52,
*
** NCDU HEX          A,   B,   C,   D,   E,   F,  10,  11,  12,  13,
** WANTED SYMBOL     #7,   .,  4L,  3L,  3R,  4R,  #2,  #5,  #8,  #0,
    *             55,  46,  04,  03,  09,  10,  50,  53,  56,  48,
*
** NCDU HEX         14,  15,  16,  17,  18,  19,  1A,  1B,  1C,  1D,
** WANTED SYMBOL     6L,  5L,  5R,  6R,  #3,  #6,  #9, +/-, NXP,
FIX,
    *             06,  05,  11,  12,  51,  54,  57,  43,  33,  15,
*
** NCDU HEX         1E,  1F,  20,  21,  22,  23,  24,  25,  26,  27,
** WANTED SYMBOL    DIN, INT,   K,   P,   U,   Z,   F,   A, LEG,
RTE,
    *             60,  39,  75,  80,  85,  90,  70,  65,  41,  40,
*
** NCDU HEX         28,  29,  2A,  2B,  2C,  2D,  2E,  2F,  30,  31,
** WANTED SYMBOL      L,   Q,   V, BLK,   G,   B, DAR, CLB,   M,
R,
    *             76,  81,  86,  32,  71,  66,  61,  34,  77,  82,
*
** NCDU HEX         32,  33,  34,  35,  36,  37,  38,  39,  3A,  3B,
```

```
** WANTED SYMBOL     W, DEL,   H,   C, HLD, CRZ,   N,   S,   X,   /,
    *             87,  59,  72,  67,  62,  35,  78,  83,  88,  47,
*
** NCDU HEX          3C,  3D,  3E,  3F,  40,  41,  42,  43,  44,  45,
** WANTED SYMBOL     I,   D, PRG, DES,   O,   T,   Y,  --,   J,   E,
    *             73,  68,  63,  36,  79,  84,  89,  00,  74,  69,
*
** NCDU HEX          46,  47,  48,  49,  4A,  4B,  4C,  4D,  4D,  4F,
** WANTED SYMBOL   EXC,  --, CL1, PVP, N1L, CL2,  --,  --,  --,  --,
    *             58,  00,  13,  42,  64,  14,  00,  00,  00,  00 /
*
* initialize and reset values once per run
*
      if (t .eq. 0.0) then
        tfr1p   = .not. comtfr1
        tfr2p   = .not. comtfr2
        apprrcp = ' '  ! force "matched...appr" event posting
        towrrcp = towrrc
        comprcp = comprc
        comprc0 = comprc
        atisrcp = atisrc
        audiop  = '*'
        talkp   = '*'
        ixtrnsp = -1
        pmikep  = .not. pmike
        cmikep  = .not. cmike
        imenu1p = -1
        imenu2p = -1
        sbeltp  = .not. seatblt
        nsmkp   = .not. nosmoke
        llgtp   = .not. landlgt
        skdp    = .not. antiskd
        sackp   = .false.
        ixautbp = -1
        gearcp  = -1
        flapcp  = -1
        ixspbp  = -1
        thrcp   = '*'
        tpitch  = 0.
        troll   = 0.
        puodp   = .false.
        pdodp   = .false.
        rlodp   = .false.
        rrodp   = .false.
        eventmp = .false.
        ip2dnp  = -1
        ip2dp   = -2
        altcmdp = 0.
        i18301 = 0
        i12301 = 0
        i10301 = 0
        i8301 = 0
        i4301 = 0
        cmsptp  = 0.
        altc    = ' '
        altcp   = ' '
        altp    = alt
        iasc    = ' '
        iascp   = ' '
        iasp    = ias
        wevt1p  = .false. !itmarker events window run type
        wevt2p  = .false.
        wevt3p  = .false.
```

```
        todbeg  = .false. !modeling events for each procedural interval
        todend  = .false.
        k18beg  = .false.
        k18end  = .false.
        fafbeg  = .false.
        fafend  = .false.
        itt2atc = 0       !modeling events for each intervening task
        irunack = .false.
        irunbeg = .false.
        irunend = .false.
        nrunack = .false.
        nrunbeg = .false.
        nrunend = .false.
        hpatack = .false.
        hpatbeg = .false.
        hpatend = .false.
        achgack = .false.
        achgbeg = .false.
        achgend = .false.
        schgack = .false.
        schgbeg = .false.
        schgend = .false.
        ivisack = .false.
        ivisbeg = .false.
        ivisend = .false.
*
        do i = itchbeg, itchend
          text(i) = touchm
        enddo
*
        text(itchbeg)(17:nchar)='=(MM)-TO=(Pre-Flight&Taxi-Out-Menu)'
        text(itchbeg+1)(17:nchar)='=(MM)-TO=(Take-Off&Climb-Menu)'
        text(itchbeg+2)(17:nchar)='=(MM)-TO=(Cruise(empty))'
        text(itchbeg+3)(17:nchar)='=(MM)-TO=(Approach&Descent-Menu)'
        text(itchbeg+4)(17:nchar)='=(MM)-TO=(Landing&Taxi-In-Menu)'
        text(itchbeg+5)(17:nchar)='=(Pre-Flight&Taxi-Out-Menu)-TO=(MM)'
        text(itchbeg+6)(17:nchar)=
     *   '=(Pre-Flight&Taxi-Out-Menu)-TO=(Cockpit-Prep-Cklst(empty))'
        text(itchbeg+7)(17:nchar)=
     *   '=(Pre-Flight&Taxi-Out-Menu)-TO=(Engine-Start-Cklst(empty))'
        text(itchbeg+8)(17:nchar)=
     *   '=(Pre-Flight&Taxi-Out-Menu)-TO=(After-Start-Cklst(empty))'
        text(itchbeg+9)(17:nchar)=
     *   '=(Pre-Flight&Taxi-Out-Menu)-TO=(Taxi-Out-Cklst(empty))'
        text(itchbeg+10)(17:nchar)='=(Take-Off&Climb-Menu)-TO=(MM)'
        text(itchbeg+11)(17:nchar)=
     *   '=(Take-Off&Climb-Menu)-TO=(Take-Off-Cklst(empty))'
        text(itchbeg+12)(17:nchar)=
     *   '=(Take-Off&Climb-Menu)-TO=(Rejected-Take-Off-Cklst(empty))'
        text(itchbeg+13)(17:nchar)=
     *   '=(Take-Off&Climb-Menu)-TO=(Climb-Cklst(empty))'
        text(itchbeg+14)(17:nchar)='=(Approach&Descent-Menu)-TO=(MM)'
        text(itchbeg+15)(17:nchar)=
     *   '=(Approach&Descent-Menu)-TO=(Approach-Cklst)'
        text(itchbeg+16)(17:nchar)=
     *   '=(Approach&Descent-Menu)-TO=(Descent-Cklst)'
        text(itchbeg+17)(17:nchar)=
     *   '=(Approach&Descent-Menu)-TO=(Go-Around-Cklst)'
```

```
        text(itchbeg+18)(17:nchar)='=(Landing&Taxi-In-Menu)-TO=(MM)'
        text(itchbeg+19)(17:nchar)=
     *   '=(Landing&Taxi-In-Menu)-TO=(Taxi-In-Cklst(empty))'
        text(itchbeg+20)(17:nchar)=
     *   '=(Landing&Taxi-In-Menu)-TO=(Parking-Cklst(empty))'
        text(itchbeg+21)(17:nchar)=' '
        text(itchbeg+22)(17:nchar)=' '
        text(itchbeg+23)(17:nchar)='=(Cockpit-Prep-Cklst(empty))-TO=(MM)'
        text(itchbeg+24)(17:nchar)='=(Engine-Start-Cklst(empty))-TO=(MM)'
        text(itchbeg+17)(17:nchar)='=(After-Start-Cklst(empty))-TO=(MM)'
        text(itchbeg+26)(17:nchar)='=(Taxi-Out-Cklst(empty))-TO=(MM)'
        text(itchbeg+27)(17:nchar)='=(Take-Off-Cklst(empty))-TO=(MM)'
        text(itchbeg+28)(17:nchar)=
     *   '=(Rejected-Take-Off-Cklst(empty))-TO=(MM)'
        text(itchbeg+29)(17:nchar)='=(Climb-Cklst(empty))-TO=(MM)'
        text(itchbeg+30)(17:nchar)='=(Approach-Cklst)-TO=(MM)'
        text(itchbeg+31)(17:nchar)='=(Descent-Cklst)-TO=(MM)'
        text(itchbeg+32)(17:nchar)='=(Go-Around-Cklst)-TO=(MM)'
        text(itchbeg+33)(17:nchar)='=(Landing-Cklst)-TO=(MM)'
        text(itchbeg+34)(17:nchar)='=(Landing-Roll-Cklst(empty))-TO=(MM)'
        text(itchbeg+35)(17:nchar)='=(Taxi-In-Cklst(empty))-TO=(MM)'
        text(itchbeg+36)(17:nchar)='=(Parking-Cklst(empty))-TO=(MM)'
        text(itchbeg+37)(17:nchar)='=(Cruise(empty))-TO=(MM)'
*
      endif
*
** Clear flags used to signal the occurance of events
*
      do i=1, nline
        itext(i) = 0
      enddo
*
** toggling a TFR on a COM channel for a radio event
*
      if (comtfr1 .ne. tfr1p) then
        itext(itfr1) = 1
        text(itfr1)(1:13) = '<RADIO>:COM1='
        if (comtfr1) then
          ix = 1
        else
          ix = 2
        endif
        text(itfr1)(14:nchar) = service(ix)
        tfr1p = comtfr1
      endif
      if (comtfr2 .ne. tfr2p) then
        itext(itfr2) = 1
        text(itfr2)(1:13) = '<RADIO>:COM2='
        if (comtfr2) then
          ix = 3
        else
          ix = 4
        endif
        text(itfr2)(14:nchar) = service(ix)
        tfr2p = comtfr2
      endif
*
** detecting a radio frequency change caused by tuning
*
      if (apprrc .ne. apprrcp) then
```

```
        itext(ifrq1) = 1
        text(ifrq1)(1:8) = '<RADIO>:'
        if (apprrc .eq. apprfc) then
          text(ifrq1)(9:21) = 'MATCHED-FREQ-'
          text(ifrq1)(22:29) = service(1)
          text(ifrq1)(30:30) = '='
          text(ifrq1)(31:nchar) = apprrc
        else
          text(ifrq1)(9:20) = 'TUNING-FREQ-'
          text(ifrq1)(21:28) = service(1)
          text(ifrq1)(29:29) = '='
          text(ifrq1)(30:nchar) = apprrc
        endif
        apprrcp = apprrc
      else if (towrrc .ne. towrrcp) then
        itext(ifrq1) = 1
        text(ifrq1)(1:8) = '<RADIO>:'
        if (towrrc .eq. towrfc) then
          text(ifrq1)(9:21) = 'MATCHED-FREQ-'
          text(ifrq1)(22:26) = service(2)
          text(ifrq1)(27:27) = '='
          text(ifrq1)(28:nchar) = towrrc
        else
          text(ifrq1)(9:20) = 'TUNING-FREQ-'
          text(ifrq1)(21:25) = service(2)
          text(ifrq1)(26:26) = '='
          text(ifrq1)(27:nchar) = towrrc
        endif
        towrrcp = towrrc
      endif
*
      if (comprc .ne. comprcp) then
        itext(ifrq2) = 1
        text(ifrq2)(1:8) = '<RADIO>:'
        if (comprc .eq. compfc) then
          text(ifrq2)(9:21) = 'MATCHED-FREQ-'
          text(ifrq2)(22:28) = service(3)
          text(ifrq2)(29:29) = '='
          text(ifrq2)(30:nchar) = comprc
        else
          text(ifrq2)(9:20) = 'TUNING-FREQ-'
          text(ifrq2)(21:27) = service(3)
          text(ifrq2)(28:28) = '='
          text(ifrq2)(29:nchar) = comprc
        endif
        comprcp = comprc
      else if (atisrc .ne. atisrcp) then
        itext(ifrq2) = 1
        text(ifrq2)(1:8) = '<RADIO>:'
        if (cfrqc2 .eq. atisfc) then
          text(ifrq2)(9:21) = 'MATCHED-FREQ-'
          text(ifrq2)(22:25) = service(4)
          text(ifrq2)(26:26) = '='
          text(ifrq2)(27:nchar) = atisrc
        else
          text(ifrq2)(9:20) = 'TUNING-FREQ-'
          text(ifrq2)(21:24) = service(4)
          text(ifrq2)(25:25) = '='
          text(ifrq2)(26:nchar) = atisrc
        endif
        atisrcp = atisrc
      endif
*
      if (vbatch .eq. 0) then
        if (t .eq. 0.) then
```

```
          do i = 1, lpa
            listen(i) = .true.
          enddo
        else
          listen(2) = .true.
          listen(5) = .true.
        endif
      endif
*
      if (vbatch .eq. 0) then
        if (t .eq. 0.) then
          trnsmit(3) = .true.
        else if (amod(t, 1.) .eq. 0.) then
          trnsmit(3) = .false.
          trnsmit(4) = .true.
        endif
      endif
*
** detect any listen or transmit switch changes for a radio event
*
** a total of 5 listen switches, more than one switch can be selected
** at a time: 1=int, 2=vhf1, 3=vhf2, 4=vhf3, 5=pa
*
** a total of 6 transmit channels controled by a knob, only 1 channel can
** be selected at a time: 1=servint, 2=int, 3=vhf1, 4=vhf2, 5=vhv3, 6=pa
*
      do i = 1, ixpa
        if (trnsmit(i)) ixtrns = i
      enddo
*
      ix = 2
      jx = 2
*
      audio = '('
      if (listen(1) .or. ixtrns .eq. 2) then ! int listen or transmit
        jx = ix + 3
        audio(ix:jx-1) = 'INT'
        ix = jx
      endif
      if (listen(2) .or. ixtrns .eq. 3) then ! vhf1 listen or transmit
        if (audio .ne. '(') then
          audio(ix:ix) = ';'
          ix = ix + 1
        endif
        if (comtfr1) then
          is = 8
          id = 1
        else
          is = 5
          id = 2
        endif
        jx = ix + is
        audio(ix:jx-1) = service(id)
        ix = jx
      endif
      if (listen(3) .or. ixtrns .eq. 4) then ! vhf2 audio or vhf2 talk
        if (audio .ne. '(') then
          audio(ix:ix) = ';'
          ix = ix + 1
        endif
        if (comtfr2) then
          is = 7
          id = 3
        else
          is = 4
          id = 4
        endif
        jx = ix + is
        audio(ix:jx-1) = service(id)
        ix = jx
      endif
      if (listen(5)) then
        if (audio .ne. '(') then
          audio(ix:ix) = ';'
          ix = ix + 1
        endif
        jx = ix + 2
        audio(ix:jx-1) = 'PA'
        ix = jx
      endif
      audio(ix:ix) = ')'
*
      if (audio .ne. audiop) then
        itext(ilisten) = 1
        text(ilisten)(1:18) = '<RADIO>:LISTEN-TO='
        text(ilisten)(19:nchar) = audio
        audiop = audio
      endif
*
      if (ixtrns .eq. 1) then
        talk = 'SERVINT'
      else if (ixtrns .eq. 2) then
        talk = 'INT'
      else if (ixtrns .eq. 3) then
        if (comtfr1) then
          talk = service(1)
        else
          talk = service(2)
        endif
      else if (ixtrns .eq. 4) then
        if (comtfr2) then
          talk = service(3)
        else
          talk = service(4)
        endif
      else if (ixtrns .eq. 5) then
        talk = 'VHF3'
      else if (ixtrns .eq. 6) then
        talk = 'PA'
      endif
*
      if (talk .ne. talkp) then
        itext(itrns) = 1
        text(itrns)(1:24) = '<RADIO>:TRANSMIT-SELECT='
        text(itrns)(25:nchar) = talk
        talkp = talk
      endif
*
** microphone keyed on/off for radio events
*
      if (vbatch .eq. 0) then
        if (t .eq. h*5) then
           pmike = .true.
        else if (t .eq. h*10) then
           pmike = .false.
        else if (t .gt. 1.) then
           cmike = .true.
        endif
      endif
*
```

```
cc      if (pmike .ne. pmikep .or. cmike .ne. cmikep) then
cc         if (pmike .or. cmike) then
      if (pmike .ne. pmikep) then
        if (pmike) then
          itext(ikeyon) =1
          text(ikeyon)(1:16) = '<RADIO>:TALK-TO='
          text(ikeyon)(17:nchar) = talk
          if (talk .eq. service(1) .or. talk .eq. service(2)) itt2atc = 1
        else if (t .ne. 0.) then
          itext(ikeyoff) = 1
          text(ikeyoff)(1:24) = '<RADIO>:TALK-STOPPED-TO='
          text(ikeyoff)(25:nchar) = talk
          itt2atc = 0
        endif
        pmikep = pmike
        cmikep = cmike
      endif
*
** touchscreen event
*
      if (imenu1p .ne. ixmenu(1) .and. ixmenu(1) .ne. 0) then
        if (ixmenu(1) .eq. 39) then
          itext(itchrtn) = 1
          text(itchrtn) = chklist(nxmenu(1))
        else
          itext(ixmenu(1)+itchbeg-1) = 1
        endif
        imenu1p = ixmenu(1)
      endif
*
** fms key events
*
      mpag = kmax0(modpag, 1)                 ! init value = 0
      if (keycdu .eq. 200) keycdu = 75        ! TCLR -> CL2
      if (keycdu .gt. 0 .and. keycdu .lt. 80) then ! test via auto keys
        icduseq = ncdcvt(keycdu + 1)
        if (icduseq .ge. 1 .and. icduseq .le. 12) then
          itext(ifms) = 1
          text(ifms)(1:12) = '<FMS>:PAGE=('
          text(ifms)(13:20) = wincdu
          text(ifms)(21:21) = ';'
          call i2ch(int8(mpag), 1)
          text(ifms)(22:22) = charray(1:1)
          text(ifms)(23:36) = ')-LINESELECT=('
          if (icduseq .le. 6) then
            lsk = icduseq * 2
            ii = 1
            jj = 12
          else
            lsk = (icduseq - 6) * 2
            ii = 13
            jj = 24
          endif
          do j = 1, 312
            mpagel(j) = jpagel(j) .and. '7f'x
            if ((mpagel(j) .ge. 16) .and. (mpagel(j) .le. 25)) then
              mpagel(j) = mpagel(j) + 32
            endif
          enddo
          text(ifms)(37:48) = mpagec(lsk)(ii:jj)
          text(ifms)(49:49) = ';'
          text(ifms)(50:61) = mpagec(lsk+1)(ii:jj)
          text(ifms)(62:62) = ';'
          text(ifms)(63:64) = cdukys(keycdu)
          text(ifms)(65:nchar) = ')'
```

```
        else if (icduseq .ge. 43 .and. icduseq .le. 57 .or. !+/-, ., /, 0-9
     *           icduseq .ge. 65 .and. icduseq .le. 90 .or. !-Z
     *           icduseq .eq. 32 .or. icduseq .eq. 58 .or.  !BLANK,EXEC
     *           icduseq .eq. 59 .or. icduseq .eq. 13 .or.  !DEL, CLR char
     *           icduseq .eq. 14) then                      !CLR line
          itext(ifms) = 1
          text(ifms)(1:12) = '<FMS>:PAGE=('
          text(ifms)(13:20) = wincdu
          text(ifms)(21:21) = ';'
          call i2ch(int8(mpag), 1)
          text(ifms)(22:22) = charray(1:1)
          text(ifms)(23:31) = ')-TYPED=('
          text(ifms)(32:34)= cdukys(keycdu)
          text(ifms)(35:nchar) = ')'
        else if (icduseq .eq. 15 .or.                       !FIX
     *           icduseq .ge. 33 .and. icduseq .le. 36 .or.
!NXP,CLB,CRZ,DES
     *           icduseq .ge. 39 .and. icduseq .le. 42 .or.
!INIT,RTE,LEGS,PVG
     *           icduseq .ge. 60 .and. icduseq .le. 64)
then!DINT,DAR,HLD,PRG
          itext(ifms) = 1
          text(ifms)(1:12) = '<FMS>:FROM=('
          text(ifms)(13:20) = wincdu
          text(ifms)(21:21) = ';'
          call i2ch(int8(mpag), 1)
          text(ifms)(22:22) = charray(1:1)
          text(ifms)(23:28) = ')-TO=('
          text(ifms)(29:36) = cdukys(keycdu)
          if (icduseq .eq. 33 .or. icduseq .eq. 42) then
            text(ifms)(37:nchar) = ')'
          else
            text(ifms)(37:37) = ';'
            text(ifms)(38:38) = charray(1:1)
            text(ifms)(39:nchar) = ')'
          endif
        endif
      endif
*
** overhead panel event for Seat Belt
*
      if (seatblt .ne. sbeltp) then
        itext(isbelt) = 1
        text(isbelt) = '<OVERHEAD>:SEATBELT-SIGN='
        if (seatblt) then
          text(isbelt)(26:nchar) = 'ON'
        else
          text(isbelt)(26:nchar) = 'OFF'
        endif
        sbeltp = seatblt
      endif
*
** overhead panel event for No Smoking Sign
*
      if (nosmoke .ne. nsmkp) then
        itext(ismoke) = 1
        text(ismoke) = '<OVERHEAD>:NO-SMOKING-SIGN='
        if (nosmoke) then
          text(ismoke)(28:nchar) = 'ON'
        else
          text(ismoke)(28:nchar) = 'OFF'
        endif
        nsmkp = nosmoke
      endif
*
```

```
** overhead panel event for landing light
*
      if (landlgt .ne. llgtp) then
        itext(illgt) = 1
        text(illgt) = '<OVERHEAD>:LANDING-LIGHTS-SIGN='
        if (landlgt) then
          text(illgt)(32:nchar) = 'ON'
        else
          text(illgt)(32:nchar) = 'OFF'
        endif
        llgtp = landlgt
      endif
*
** overhead panel event for anti-skid
*
      if (antiskd .ne. skdp) then
        itext(iskid) = 1
        text(iskid) = '<OVERHEAD>:ANTI-SKID='
        if (antiskd) then
          text(iskid)(22:nchar) = 'ON'
        else
          text(iskid)(22:nchar) = 'OFF'
        endif
        skdp = antiskd
      endif
*
** overhead panel event for autobrakes
*
      if (ixautb .ne. ixautbp) then
        itext(iabrk) =  1
        text(iabrk)(1:22) = '<OVERHEAD>:AUTOBRAKES='
        text(iabrk)(23:nchar) = autbsw(ixautb)
        ixautbp = ixautb
      endif
*
** energy control event for landing gear
*
      if (gearc .ne. gearcp) then
        itext(igear) = 1
        text(igear) = '<ENERGY-CTRL>:GEAR='
        if (gearc .eq. 0.) then
          text(igear)(20:nchar) = 'UP'
        else
          text(igear)(20:nchar) = 'DOWN'
        endif
        gearcp = gearc
      endif
*
** energy control event for flaps
*
      if (flapc .ne. flapcp) then
*
        itext(iflap) = 1
        call i2ch(int8(flapc), 2)
        text(iflap)(1:20) = '<ENERGY-CTRL>:FLAPS='
        if (charray(1:1) .eq. '0') then
          text(iflap)(21:nchar) = charray(2:2)
        else
          text(iflap)(21:nchar) = charray(1:2)
        endif
        flapcp = flapc
      endif
*
** energy control for speed brakes
*
```

```
      if (ixspb .ne. ixspbp) then
        itext(ispb) = 1
        text(ispb)(1:26)  = '<ENERGY-CTRL>:SPEEDBRAKES='
        call i2ch(ixspb, 1)
        text(ispb)(27:nchar) = charray(1:1)
        ixspbp = ixspb
      endif
*
** energy control for throttle
*
      if (thrc .ne. '0 ' .and. throtlc .le. 0. .or.
     *   thrc .eq. '0 ' .and. throtlc .le. 1.) then
        thrc = '0 '
      else if (thrc .ne. '1 ' .and. throtlc .le. 5. .or.
     *        thrc .eq. '1 ' .and. throtlc .le. 6.) then
        thrc = '1 '
      else if (thrc .ne. '2 ' .and. throtlc .le. 10. .or.
     *        thrc .eq. '2 ' .and. throtlc .le. 11.) then
        thrc = '2 '
      else if (thrc .ne. '3 ' .and. throtlc .le. 15. .or.
     *        thrc .eq. '3 ' .and. throtlc .le. 16.) then
        thrc = '3 '
      else if (thrc .ne. '4 ' .and. throtlc .le. 20. .or.
     *        thrc .eq. '4 ' .and. throtlc .le. 21.) then
        thrc = '4 '
      else if (thrc .ne. '5 ' .and. throtlc .le. 25. .or.
     *        thrc .eq. '5 ' .and. throtlc .le. 26.) then
        thrc = '5 '
      else if (thrc .ne. '6 ' .and. throtlc .le. 30. .or.
     *        thrc .eq. '6 ' .and. throtlc .le. 31.) then
        thrc = '6 '
      else if (thrc .ne. '7 ' .and. throtlc .le. 35. .or.
     *        thrc .eq. '7 ' .and. throtlc .le. 36.) then
        thrc = '7 '
      else if (thrc .ne. '8 ' .and. throtlc .le. 40. .or.
     *        thrc .eq. '8 ' .and. throtlc .le. 41.) then
        thrc = '8 '
      else if (thrc .ne. '9 ' .and. throtlc .le. 45. .or.
     *        thrc .eq. '9 ' .and. throtlc .le. 46.) then
        thrc = '9 '
      else if (thrc .ne. '10' .and. throtlc .le. 50. .or.
     *        thrc .eq. '10' .and. throtlc .le. 51.) then
        thrc = '10'
      else if (thrc .ne. '11' .and. throtlc .le. 55. .or.
     *        thrc .eq. '11' .and. throtlc .le. 56.) then
        thrc = '11'
      else
        thrc = '12'
      endif
*
      if (thrcp .ne. thrc) then
        itext(ithrc) = 1
        text(ithrc)(1:23)  = '<ENERGY-CTRL>:THROTTLE='
        text(ithrc)(24:nchar) = thrc
        thrcp = thrc
*
** flight path marker for throttle moved to idle
*
        if (throtlc .lt. 5.) then
          itext(itidle) = 1
          text(itidle) = '<FLIGHT-PATH>:THROTTLE-POSITION=IDLE'
        endif
      endif
*
** energy control for columc
```

```
*
** calculate the total time that the columc is out of detend
*
    if (tpitch .gt. 0.) tpitch = tpitch + h
*
    if (puod .ne. puodp .or. pdod .ne. pdodp) then
      itext(istickp) = 1
      if (puod) then
        text(istickp)(1:nchar) = '<ENERGY-CTRL>:STICK-PITCH-UP'
        tpitch = h
      else if (pdod) then
        text(istickp)(1:nchar) = '<ENERGY-CTRL>:STICK-PITCH-
DOWN'
        tpitch = h
      else
        text(istickp)(1:32) = '<ENERGY-CTRL>:STICK-PITCH-
TOTAL='
        call i2ch(int8(tpitch * 1000), 6)
        text(istickp)(33:35) = charray(1:3)
        text(istickp)(36:36) = '.'
        text(istickp)(37:nchar) = charray(4:6)
        tpitch = 0.
      endif
    endif
    puodp = puod
    pdodp = pdod
*
** energy control for wheelc
*
** calculate the total time that the wheelc is out of detend
*
    if (troll .gt. 0.) troll = troll + h
*
    if (rlod .ne. rlodp .or. rrod .ne. rrodp) then
      itext(istickr) = 1
      if (rlod) then
        text(istickr)(1:nchar) = '<ENERGY-CTRL>:STICK-ROLL-
LEFT'
        troll = h
      else if (rrod) then
        text(istickr)(1:nchar) = '<ENERGY-CTRL>:STICK-ROLL-
RIGHT'
        troll = h
      else
        text(istickr)(1:31) = '<ENERGY-CTRL>:STICK-ROLL-
TOTAL='
        call i2ch(int8(troll * 1000), 6)
        text(istickr)(32:34) = charray(1:3)
        text(istickr)(35:35) = '.'
        text(istickr)(36:nchar) = charray(4:6)
      endif
    endif
    rlodp = rlod
    rrodp = rrod
*
** Test for even marker to be printed
*
    if (eventm .and. eventm .ne. eventmp) then
      itext(ieventm) = 1
      text(ieventm) = '<E-MARKER>:EEEEE'
    endif
    eventmp = eventm
*
** event triggered by passing a waypoint
*
```

```
    if (ip2dn .ne. ip2dnp) then
*
** intervening task markers (use current wpt pointer)
*
      iptr = ip2dn - 1
      if (iptr .ge. 1 .and. iptr .le. 8) then
        itext(itmarka) = 1
        text(itmarka)(1:24)  = '<IT-MARKER>:WINDOW-TYPE='
        text(itmarka)(25:nchar) = ptrmark(iptr)
      endif
*
** flight path marker for waypoint just passed (current pointer)
*
      itext(ipwpt) = 1
      text(ipwpt)(1:27)  = '<FLIGHT-PATH>:PASSING-WYPT='
      call i2ch(ip2dn-1, 1)
      text(ipwpt)(28:nchar) = charray(1:1)
*
** performance measure for alt dev
*
      altdev = alt - altcmdp
      if (vbatch .eq. 0) altdev = 850.69
      call i2ch(int8(abs((altdev+.005) * 100)), 7)
      itext(ialtdev) = 1
      text(ialtdev)(1:32)  = '<PERF-MEASURE>:WYPT-ALT(ft)-
DEV='
      text(ialtdev)(33:37) = charray(1:5)
      text(ialtdev)(38:38) = '.'
      text(ialtdev)(39:nchar) = charray(6:7)
      if (altdev .lt. 0.) then
        textln = text(ialtdev)
        text(ialtdev)(34:41) = textln(33:40)
        text(ialtdev)(33:33) = '-'
      endif
*
** performance measure for spd dev
*
      casdev = ias - cmsptp
      if (vbatch .eq. 0) casdev = 101.45
      call i2ch(int8(abs((casdev+.005) * 100)), 5)
      itext(ispddev) = 1
      text(ispddev)(1:34)  = '<PERF-MEASURE>:WYPT-SPEED(kn)-
DEV='
      text(ispddev)(35:37) = charray(1:3)
      text(ispddev)(38:38) = '.'
      text(ispddev)(39:nchar) = charray(4:5)
      if (casdev .lt. 0.) then
        textln = text(ispddev)
        text(ispddev)(36:41) = textln(35:40)
        text(ispddev)(35:35) = '-'
      endif
*
** performance measure for head dev
*
      if (vbatch .eq. 0) tke = -101.45
      call i2ch(int8(abs((tke+.005) * 100)), 5)
*
      itext(ihddev) = 1
      text(ihddev)(1:32)  = '<PERF-MEASURE>:WYPT-HEAD(d)-
DEV='
      text(ihddev)(33:35) = charray(1:3)
      text(ihddev)(36:36) = '.'
      text(ihddev)(37:nchar) = charray(4:5)
      if (tke .lt. 0.) then
        textln = text(ihddev)
```

```
          text(ihddev)(33:33) = '-'
          text(ihddev)(34:39) = textln(33:38)
        endif
*
** performance measure for latitude dev
*
        if (vbatch .eq. 0) xtk = -261.45
        call i2ch(int8(abs((xtk+.005) * 100)), 5)
*
        itext(ilatdev) = 1
        text(ilatdev)(1:32)  = '<PERF-MEASURE>:WYPT-LAT(ft)-
DEV='
        text(ilatdev)(33:35) = charray(1:3)
        text(ilatdev)(36:36) = '.'
        text(ilatdev)(37:nchar) = charray(4:5)
        if (xtk .lt. 0.) then
          textln = text(ilatdev)
          text(ilatdev)(33:33) = '-'
          text(ilatdev)(34:39) = textln(33:38)
        endif
*
        ip2dnp = ip2dn
*
      endif
*
** itmarker events for window run type, the positions for interruption
** tasks
*
*
      if (wevt1.ne.wevt1p .or. wevt2.ne.wevt2p .or. wevt3.ne.wevt3p) then
        itext(itmarkb) = 1
        text(itmarkb)(1:12) = '<IT-MARKER>:'
        text(itmarkb)(13:15) = itwinc(itwin)
        text(itmarkb)(16:16) = '-'
        if (itrun .eq. 1 .or. itrun .eq. 3) then
          text(itmarkb)(17:23) = itrunc(itrun)
          text(itmarkb)(24:24) = '-'
          text(itmarkb)(25:nchar) = ittypc(ittyp)
        else
          text(itmarkb)(17:22) = itrunc(itrun)
          text(itmarkb)(23:23) = '-'
          text(itmarkb)(24:nchar) = ittypc(ittyp)
        endif
        wevt1p = wevt1
        wevt2p = wevt2
        wevt3p = wevt3
      endif
*
** flight path marker for altitude bug change
*
      altcmd = altcmd + .0005
      if (altcmd .lt. 100.) altcmd = altcmdp
      if (altcmd .ne. altcmdp) then
        itext(ialtbug) = 1
        text(ialtbug)(1:38)  = '<FLIGHT-PATH>:ALT-CHANGE-
INDICATED-TO='
        if (altcmd .ge. 10000.) then
          kbit = 8
        else if (altcmd .ge. 1000.) then
          kbit = 7
        else if (altcmd .ge. 100.) then
          kbit = 6
        else if (altcmd .ge. 10.) then
          kbit = 5
        else
          kbit = 4
        endif
        call i2ch(int8(altcmd * 1000), kbit)
        text(ialtbug)(39:39+kbit-4) = charray(1:kbit-3)
        text(ialtbug)(39+kbit-3:39+kbit-3) = '.'
        text(ialtbug)(39+kbit-2:nchar) = charray(kbit-2:kbit)
        altcmdp = altcmd
      endif
*
** flight path marker for start to level of to altitude (18000, 12000,
** 10000, 8000, 4000)
*
      if (alt .le. 18301. .and. alt .ge. 18250. .and. i18301 .eq. 0 .or.
     *    alt .le. 12301. .and. alt .ge. 12250. .and. i12301 .eq. 0 .or.
     *    alt .le. 10301. .and. alt .ge. 10250. .and. i10301 .eq. 0 .or.
     *    alt .le. 8301.  .and. alt .ge. 8250.  .and. i8301  .eq. 0 .or.
     *    alt .le. 4301.  .and. alt .ge. 4250.  .and. i4301  .eq. 0) then
        itext(ialtlev) = 1
        text(ialtlev)(1:33) = '<FLIGHT-PATH>:START-LEVEL-OFF-
TO='
        if (alt .le. 18301. .and. i18301 .eq. 0 ) then
          i18301 = 1
          clevoff = '18000'
        else if (alt .le. 12301. .and. i12301 .eq. 0) then
          i12301 = 1
          clevoff = '12000'
        else if (alt .le. 10301. .and. i10301 .eq. 0) then
          i10301 = 1
          clevoff = '10000'
        else if (alt .le. 8301.  .and. i8301  .eq. 0) then
          i8301 = 1
          clevoff = '8000'
        else if (alt .le. 4301.  .and. i4301  .eq. 0) then
          i4301 = 1
          clevoff = '4000'
        endif
        text(ialtlev)(34:nchar) = clevoff
      endif
*
** flight path marker for speed bug change
*
      casmspt = casmspt + .0005
      if (casmspt .lt. 100.) casmspt = cmsptp
      if (casmspt .ne. cmsptp) then
        itext(ispdbug) = 1
        text(ispdbug)(1:40)  = '<FLIGHT-PATH>:SPEED-CHANGE-
INDICATED-TO='
        if (casmspt .ge. 100.) then
          kbit = 6
        else if (casmspt .ge. 10.) then
          kbit = 5
        else
          kbit = 4
        endif
        call i2ch(int8(casmspt * 1000), kbit)
        text(ispdbug)(41:41+kbit-4) = charray(1:kbit-3)
        text(ispdbug)(41+kbit-3:41+kbit-3) = '.'
        text(ispdbug)(41+kbit-2:nchar) = charray(kbit-2:kbit)
        cmsptp = casmspt
      endif
*
** flight path marker for 2 miles in front of a wpt (next pointer)
*
      if (ip2d.ne.ip2dp .and. dtogonm.le.ralciic .and. ralciic.ne.0.) then
         itext(i2miles) = 1
```

```
          text(i2miles)(1:34) = '<FLIGHT-PATH>:TURN-IN-9.999-
MILES='
          call i2ch(int8((ralciic+.0005) * 1000), 4)
          text(i2miles)(23:23) = charray(1:1)
          text(i2miles)(25:27) = charray(2:4)
          call i2ch(ip2dn, 1)
          text(i2miles)(35:nchar) = charray(1:1)
          ip2dp = ip2d
      endif
*
** flight path marker for achieved an altitude restriction
*
      if (altp .ge. 19000. .and. alt .le. 19000.) then
        altc = '19000'
      else if (altp .ge. 18000. .and. alt .le. 18000.) then
        altc = '18000'
      else if (altp .ge. 12000. .and. alt .le. 12000.) then
        altc = '12000'
      else if (altp .ge. 10000. .and. alt .le. 10000.) then
        altc = '10000'
      else if (altp .ge. 8000. .and. alt .le. 8000.) then
        altc = '8000 '
      else if (altp .ge. 4000. .and. alt .le. 4000.) then
        altc = '4000 '
      endif
      altp = alt
*
      if (altcp .ne. altc) then
        itext(ialtrst) = 1
        text(ialtrst)(1:33)  = '<FLIGHT-PATH>:ALT-RESTR-
ACHIEVED='
        text(ialtrst)(34:nchar) = altc
        altcp = altc
      endif
*
** flight path marker for achieved speed restriction
*
      if (iasp .ge. 290. .and. ias .le. 290.) then
        iasc = '290'
      else if (iasp .ge. 240. .and. ias .le. 240.) then
        iasc = '240'
      else if (iasp .ge. 180. .and. ias .le. 180.) then
        iasc = '180'
      else if (iasp .ge. 150. .and. ias .le. 150.) then
        iasc = '150'
      else if (iasp .ge. 140. .and. ias .le. 140.) then
        iasc = '140'
      endif
      iasp = ias
*
      if (iascp .ne. iasc) then
        itext(ispdrst) = 1
        text(ispdrst)(1:35)  = '<FLIGHT-PATH>:SPEED-RESTR-
ACHIEVED='
        text(ispdrst)(36:nchar) = iasc
        iascp = iasc
      endif
*
** performance measure for selcal acknowledged
*
      if (selcack .and. selcack .ne. sackp) then
        itext(iack) = 1
        call i2ch(int8(sindt * 1000), 9)
        text(iack)(1:30)  = '<PERF-MEASURE>:ACKNOWLEDGE-RT='
        text(iack)(31:36) = charray(1:6)
```

```
        text(iack)(37:37) = '.'
        text(iack)(38:nchar) = charray(7:9)
        sackp = selcack
      endif
*
** modeling events for each procedural interval
*
      if (itwin .eq. 1) then
        if (.not. todbeg .and. comprc .ne. comprc0) then
          itext(imodelp) = 1
          text(imodelp) = '<MODEL>:TOD-PI=START'
          todbeg = .true.
        else if (.not. todend .and.
     *           wincdup .eq. initflc(8) .and. wincdu .ne. wincdup) then
          itext(imodelp) = 1
          text(imodelp) = '<MODEL>:TOD-PI=END'
          todend = .true.
        endif
      else if (itwin .eq. 2) then
        if (.not. k18beg .and.
     *      wincdu .eq. initflc(1) .and. wincdu .ne. wincdup) then
          itext(imodelp) = 1
          text(imodelp) = '<MODEL>:K18-PI=START'
          k18beg = .true.
        else if (.not. k18end .and. (ixmenu(1).eq.31 .or.
     *           ixmenu(1).eq.39 .and. nxmenu(1).eq.14)) then
          itext(imodelp) = 1
          text(imodelp) = '<MODEL>:K18-PI=END'
          k18end = .true.
        endif
      else if (itwin .eq. 3) then
        if (.not. fafbeg .and.
     *      wincdu .eq. n1limit .and. wincdu .ne. wincdup) then
          itext(imodelp) = 1
          text(imodelp) = '<MODEL>:FAF-PI=START'
          fafbeg = .true.
        else if (.not. fafend .and. (ixmenu(1).eq.32 .or.
     *     ixmenu(1).eq.39 .and. nxmenu(1).eq.15)) then
          itext(imodelp) = 1
          text(imodelp) = '<MODEL>:FAF-PI=END'
          fafend = .true.
        endif
      endif
*
** modeling events for each intervening task
*
      if (wincdu .eq. idaiflc(1) .and. wincdu .ne. wincdup) then
        tdeparr = .true.
      else
        tdeparr = .false.
      endif
      if (wincdup .eq. idaiflc(3) .and. wincdu .ne. wincdup) then
        farr = .true.
      else
        farr = .false.
      endif
      if (wincdu .eq. rtehold .and. wincdu .ne. wincdup) then
        trthpat = .true.
      else
        trthpat = .false.
      endif
      if (wincdup .eq. rtehold .and. wincdu .ne. wincdup) then
        frthpat = .true.
      else
        frthpat = .false.
```

```
      endif
      if (wincdu .eq. ilegflc(1) .and. icduseq .eq. 54) then
        leg1nm6 = .true.
      else
        leg1nm6 = .false.
      endif
      if (wincdu .eq. ilegflc(1) .and. icduseq .eq. 49) then
        leg1nm1 = .true.
      else
        leg1nm1 = .false.
      endif
      if (wincdu .eq. ilegflc(1) .and. icduseq .eq. 58) then
        leg1exc = .true.
      else
        leg1exc = .false.
      endif
*
** ACKN: <RADIO>:TALK-TO={approach or tower} - ittyp = 1
**      <DATALINK>:FROM=(MESSAGE)-TO=(MM-{ROGER or
STANDBY}) - ittyp = 2
** START: <FMS>:FROM=(anypage;x)-TO-(DEP/ARR:1)
** END: <FMS>:FROM=(ARR;1)-TO=(anypage;x)
*
      if (itrun .eq. 1) then
        if (.not. irunack .and. (ittyp .eq. 1 .and. itt2atc .ne. 0 .or.
     *     ittyp. eq. 2  .and. ixmenu(2) .gt. 1)) then
          itext(imodelt) = 1
          if (ittyp .eq. 1) then
            text(imodelt) = '<MODEL>:TOD-INITRUN-AUD=ACKN'
          else
            text(imodelt) = '<MODEL>:TOD-INITRUN-VIS=ACKN'
          endif
          irunack = .true.
        else if (.not. irunbeg .and. tdeparr) then
          itext(imodelt) = 1
          if (ittyp .eq. 1) then
            text(imodelt) = '<MODEL>:TOD-INITRUN-AUD=START'
          else
            text(imodelt) = '<MODEL>:TOD-INITRUN-VIS=START'
          endif
          irunbeg = .true.
        else if (.not. irunend .and. farr) then
          itext(imodelt) = 1
          if (ittyp .eq. 1) then
            text(imodelt) = '<MODEL>:TOD-INITRUN-AUD=END'
          else
            text(imodelt) = '<MODEL>:TOD-INITRUN-VIS=END'
          endif
          irunend = .true.
        endif
      else if (itrun .eq. 2) then
        if (.not. nrunack .and. itt2atc .ne. 0) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-NEWRUN-AUD=ACKN'
          text(imodelt)(9:11) = itwinc(itwin)
          nrunack = .true.
        else if (.not. nrunbeg .and. tdeparr) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-NEWRUN-AUD=START'
          text(imodelt)(9:11) = itwinc(itwin)
          nrunbeg = .true.
        else if (.not. nrunend .and. farr) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-NEWRUN-AUD=END'
          text(imodelt)(9:11) = itwinc(itwin)
          nrunend = .true.
        endif
      else if (itrun .eq. 3) then
        if (.not. hpatack .and. itt2atc .ne. 0) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-HOLDPAT-AUD=ACKN'
          text(imodelt)(9:11) = itwinc(itwin)
          hpatack = .true.
        else if (.not. hpatbeg .and. trthpat) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-HOLDPAT-AUD=START'
          text(imodelt)(9:11) = itwinc(itwin)
          hpatbeg = .true.
        else if (.not. hpatend .and. frthpat) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-HOLDPAT-AUD=END'
          text(imodelt)(9:11) = itwinc(itwin)
          hpatend = .true.
        endif
      else if (itrun .eq. 4) then
        if (.not. achgack .and. itt2atc .ne. 0) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-CHGALT-AUD=ACKN'
          text(imodelt)(9:11) = itwinc(itwin)
          achgack = .true.
        else if (.not. achgbeg .and. leg1nm6) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-CHGALT-AUD=START'
          text(imodelt)(9:11) = itwinc(itwin)
          achgbeg = .true.
        else if (.not. achgend .and. leg1exc) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-CHGALT-AUD=END'
          text(imodelt)(9:11) = itwinc(itwin)
          achgend = .true.
        endif
      else if (itrun .eq. 5) then
        if (.not. schgack .and. itt2atc .ne. 0) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-CHGSPD-AUD=ACKN'
          text(imodelt)(9:11) = itwinc(itwin)
          schgack = .true.
        else if (.not. schgbeg .and. leg1nm1) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-CHGSPD-AUD=START'
          text(imodelt)(9:11) = itwinc(itwin)
          schgbeg = .true.
        else if (.not. schgend .and. leg1exc) then
          itext(imodelt) = 1
          text(imodelt) = '<MODEL>:win-CHGSPD-AUD=END'
          text(imodelt)(9:11) = itwinc(itwin)
          schgend = .true.
        endif
      endif
*
** datalink events related to messages
*
      if (imenu2p .ne. ixmenu(2) .and. ixmenu(2) .ne. 0) then
        itext(ilink) = 1
        text(ilink) = msglist(ixmenu(2))
      endif
      imenu2p = ixmenu(2) ! imenu2p also used in model event
*
      do i = 1, nline
        if (itext(i) .eq. 1) then
```

```
          write(52,10) t, text(i), wtime1, wtime2, wtime3, runcond
          if (i .eq. ifms)
     *     write(53,10) t, text(i), wtime1, wtime2, wtime3, runcond
        endif
      enddo
```

```
      if (t .eq. 0.) write(52,'(a40)') ntrate
10    format(f11.2, ',  ', a80, 3(',  ', f11.3), ',  ', '<', a5, '>')
return
      end
```

Appendix 5.3

Data Compression Code[17]

---

[17] Data compression specifications were programmed by Mr. John Barry of Lockheed-Martin Technical Services at NASA Langley.

```
BEGIN{
 waypt = 0;
 wyptpass = ":PASSING-WYPT=";
 todstartstr = "<MODEL>:TOD-PI=START";
 todendstr = "<MODEL>:TOD-PI=END";
 k18startstr = "<MODEL>:K18-PI=START";
 k18endstr = "<MODEL>:K18-PI=END";
 fafstartstr = "<MODEL>:FAF-PI=START";
 fafendstr = "<MODEL>:FAF-PI=END";
 atisstr = "ATIS-frequency";
 towerstr = "Tower-frequency";
 companystr = "Company-frequency";
 altimstr = "Altimeter";
 gaeprstr = "GA-EPR";
 runwaystr = "Initial-runway";
 newrwystr = "K18-IT";
 newmafresstr = "FAF-IT";
 atisfreq = 0;
 companyfreq = 0;
 towerfreq = 0;
 estarttime = -1;
 altimeter = 0;
 gaepr = 0;
 runway = "";
 newrunway = "";
 eendtime = -1;
 passwayptstart = 0;
 passwayptstop = 0;
# used to flag bad procedure onset and performance times
 errornumber = -555;
 init = 0;
 todarrsize = 0;
 k18arrsize = 0;
 fafarrsize = 0;
 resumarrsize = 0;
# legs arrays are used to indicate whether or not the
# procedure and interrupt arrays have been fixed
 legs["tod"] = 0;
 legs["k18"] = 0;
 legs["faf"] = 0;
 legs["irv"] = 0;
 legs["nra"] = 0;
 itackntime = -1;
 itstarttime = -1;
 itperftime = -1;
 itstarttoend = errornumber;
 itackntostart = errornumber;
 itmarkererror = 0;
 iterrorno = -1;
 itaorder = -1;
 itaomit = -1;
 itsomit = -1;
 itfomit = -1;
 itleftintearly = -1;
 inttwolegs = 0;
 procresumetime = -1;
 inttoprocfpmcount = 0;
# EXC has been pushed
 startexctoendcount = -1;
 exctoendcount = errornumber;
 resumevclass = -1;
# EXC has been pushed
 intexc = -1;
```

```
# time from it marker EXC
 itmarktoexc = -1;
# time from EXC to first proc event
 excresumetime = -1;
# class of first event after EXC
 excresumeclass = -1;
# number of fpm events between EXC and next proc event
 exctoprocfpmcount = -1;
 ensembleonsettime = errornumber;
 ensembleperftime = errornumber;
 ensemblestarttime = -1;
 ensembleendtime = -1;
 ensemblettd = errornumber;
 count1 = 0;
 count2 = 0;
 ensemblefpmcount = -1;
 itirasize = 0;
 itirvsize = 0;
 itnrasize = 0;
 ithpasize = 0;
 itcsasize = 0;
 itcaasize = 0;
 intsize = 0;
# number of conditions in condition map file
 conditioncnt = 0;
 procarrsize = 0;
# counter and index into the current interruption array
 intcount = 0;
# counter and index into the current procedure  array
 proccount = 0;
# counter and index into the extraneous event array
 extranproccount = 0;
# true if in INITRUN-VIS first message was ROGER
 rogernotrequired = 0;
# true if an procedural or interruption event occurs more than once
 duplicate = 0;
# variables used to check the typed in epr value
# tempepr  is a buffer to hold the typed in epr value
 eprdigits = 5;
 eprcnt = 0;
 eprok = 0;
 eprclrcnt = 0;
# variables used to check the typed in altimete value
# tempaltimeter is a buffer to hold the typed in altimeter value
 altimdigits = 5;
 altimcnt = 0;
 altimok = 0;
 altimclrcnt = 0;
# variables used to check the typed in altitude change value
# tempalt is a buffer to hold the typed in altitude change value
 altdigits = 4;
 altval = 6500;
 k18altcnt = 0;
 k18altok = 0;
 fafaltcnt = 0;
 fafaltok = 0;
 fafaltclrcnt = 0;
 k18altclrcnt = 0;
# variables used to check the typed in speed change value
# tempspd is a buffer to hold the typed in speed change value
 spddigits = 4;
 spdval = 160;
 k18spdcnt = 0;
 k18spdok = 0;
```

```
  fafspdcnt = 0;
  fafspdok = 0;
  fafspdclrcnt = 0;
  k18spdclrcnt = 0;
# error counters
  misscount = 0;
  prevevent = 0;
  ordercount = 0;
  valok = 1;
  itselerr = -1;
  itexerr = -1;
  iterrortotal = -1;
  totalerrcount = 0;
  totmissnt = 0;
  totordcnt = 0;
  totvalcnt = 0;
  totextcnt = 0;
#  printf("\nSubj, RList, Block, Seg,  Run#, Leg,  Cond,    POnset,
PPerf,  ");
#  printf("ITAckn,  ITInit,  ITPerf,  ITAtoI,  ITItoE, ITEfrm,   PRes,
ITRev,  EnsOt,  EnsPerf,   EnsTod, EnsFPM,  PResFPM,");
#  printf("  ErrCnt, OmErr, OrdErr, ValErr, ExErr,");
#  printf("  ITA-order,   ITA-omit,   ITS-omit,   ITF-omit,  ITSelErr,
ITExErr, ITErrTot,  ");
#  printf("Int,  Goal,  IPMod,  ItMod,  WDead,   Coup,    Rel,
ITMarkErr,  EXCtoEND,  ");
#  printf("  ITExOrder,   ITExc,  ExcResT,  ExcResE,  ExcResFpm,
INTTwoLegs,");
}
{if(init == 0)
   {
# get subject and run number from the
# filename variable sent in
     run_number  = substr(FF,6,2);
     subj_number = substr(FF,2,2);
     seg_number =  substr(FF,5,1);
     if (index(substr(FF,4,1),"A"))
       block_number = 10;
     else
       if (index(substr(FF,4,1),"B"))
        block_number = 20;
       else
        block_number = -99;
# runlist is 2 if subject number
# is even, 1 if odd
     runlist = subj_number;
     runlist %= 2;
     if (runlist == 0)
       runlist = 2;
     else
       runlist = 1;
     init = 1;
   }
 if (FILENAME == "condition_map.txt")
   if (NR > 1)
     {
       conditioncnt ++;
       intarr[$1]   = $2;
       goalarr[$1]  = $3;
       ipmodarr[$1] = $4;
       itmodarr[$1] = $5;
       wdeadarr[$1] = $6;
       couparr[$1]  = $7;
       relarr[$1]   = $8;
     }
```

```
if (FILENAME == "itiraevt.set"){
  itirasize += 1;
  ira[itirasize] = $0;
}
if (FILENAME == "itirvevt.set"){
  itirvsize += 1;
  irv[itirvsize] = $0;
}
if (FILENAME == "itcaaevt.set"){
  itcaasize += 1;
  if((index($0,"<FMS>:PAGE=(LEGS    ;2")) ||\
     (index($0,"<FMS>:FROM=(LEGS    ;2")))
    if  (index($0,"<FMS>:FROM=(LEGS    ;2")){
      split($0,arr,"2");
      cak[itcaasize] = arr[1]  "2" arr[2] "2" arr[3];
      caf[itcaasize] = arr[1]  "1" arr[2] "1" arr[3];
    }
    else{
      cak[itcaasize] = substr($0,1,21)  "2" substr($0,23);
      caf[itcaasize] = substr($0,1,21)  "1" substr($0,23);
    }
  else
    if (index($0,"{K18 or FAF}")){
      split($0,arr,"{");
      temp = arr[1];
      temp1 = arr[2];
      split(temp1,arr,"}");
      cak[itcaasize] = temp "K18" arr[2];
      caf[itcaasize] = temp "FAF" arr[2];
    }
    else
      if (index($0,"{APPROACH or TOWER}")){
       split($0,arr,"{");
       temp = arr[1];
       temp1 = arr[2];
       split(temp1,arr,"}");
       cak[itcaasize] = temp "APPROACH" arr[2];
       caf[itcaasize] = temp "TOWER" arr[2];
      }
      else{
       cak[itcaasize] = $0;
       caf[itcaasize] = $0;
       }
}

if (FILENAME == "itcsaevt.set"){
  itcsasize += 1;
  if(index($0,"<FMS>:PAGE=(LEGS    ;2")){
    csk[itcsasize] = substr($0,1,21)  "2" substr($0,23);
    csf[itcsasize] = substr($0,1,21)  "1" substr($0,23);
  }
  else
    if (index($0,"{K18 or FAF}")){
      split($0,arr,"{");
      temp = arr[1];
      temp1 = arr[2];
      split(temp1,arr,"}");
      csk[itcsasize] = temp "K18" arr[2];
      csf[itcsasize] = temp "FAF" arr[2];
    }
    else
      if (index($0,"{APPROACH or TOWER}")){
       split($0,arr,"{");
       temp = arr[1];
```

```
       temp1 = arr[2];
       split(temp1,arr,"}");
       csk[itcsasize] = temp "APPROACH" arr[2];
       csf[itcsasize] = temp "TOWER" arr[2];
      }
      else{
       csk[itcsasize] = $0;
       csf[itcsasize] = $0;
      }
}

if (FILENAME == "ithpaevt.set"){
  ithpasize += 1;
  if((index($0,"<FMS>:PAGE=(HOLD    ;2")) ||\
     (index($0,"<FMS>:FROM=(HOLD    ;2")))
    if  (index($0,"<FMS>:FROM=(HOLD    ;2")){
      split($0,arr,"2");
      hpk[ithpasize] = arr[1]  "2" arr[2] "2" arr[3];
      hpf[ithpasize] = arr[1]  "1" arr[2] "1" arr[3];
    }
    else{
      hpk[ithpasize] = substr($0,1,21)  "2" substr($0,23);
      hpf[ithpasize] = substr($0,1,21)  "1" substr($0,23);
    }
  else
    if (index($0,"{K18 or FAF}")){
      split($0,arr,"{");
      temp = arr[1];
      temp1 = arr[2];
      split(temp1,arr,"}");
      hpk[ithpasize] = temp "K18" arr[2];
      hpf[ithpasize] = temp "FAF" arr[2];
    }
    else
      if (index($0,"{APPROACH or TOWER}")){
       split($0,arr,"{");
       temp = arr[1];
       temp1 = arr[2];
       split(temp1,arr,"}");
       hpk[ithpasize] = temp "APPROACH" arr[2];
       hpf[ithpasize] = temp "TOWER" arr[2];
      }
      else{
       hpk[ithpasize] = $0;
       hpf[ithpasize] = $0;
      }
}

if (FILENAME == "itnraevt.set"){
  itnrasize += 1;
  if (index($0,"{K18 or FAF}")){
    split($0,arr,"{");
    temp = arr[1];
    temp1 = arr[2];
    split(temp1,arr,"}");
    nrk[itnrasize] = temp "K18" arr[2];
    nrf[itnrasize] = temp "FAF" arr[2];
  }
  else
    if (index($0,"{APPROACH or TOWER}")){
      split($0,arr,"{");
      temp = arr[1];
      temp1 = arr[2];
      split(temp1,arr,"}");
      nrk[itnrasize] = temp "APPROACH" arr[2];
```

```
      nrf[itnrasize] = temp "TOWER" arr[2];
    }
    else{
      nrk[itnrasize] = $0;
      nrf[itnrasize] = $0;
    }
}


if (FILENAME == "resumptive.set"){
  resumarrsize += 1;
  resumarr[resumarrsize] = $0;
}

if (FILENAME == "ptodevt.set"){
  todarrsize += 1;
  todarr[todarrsize] = $0;
}
else
  if (FILENAME == "pk18evt.set"){
    k18arrsize += 1;
    k18arr[k18arrsize] = $0;
  }
  else
    if (FILENAME == "pfafevt.set"){
      fafarrsize += 1;
      fafarr[fafarrsize] = $0;
    }
    else
      {
# working with the event file
# get frequencies
        if (index($0,towerstr))
          {
            split($0,arr,":");
            towerfreq = arr[2];
          }
        else
          if (index($0,companystr))
            {
              split($0,arr,":");
              companyfreq = arr[2];
            }
          else
            if (index($0,atisstr))
              {
               split($0,arr,":");
               atisfreq = arr[2];
              }
            else
# get the altimeter setting
              if (index($0,altimstr))
                {
                  split($0,arr,":");
                  altimeter = arr[2] * 100;
                }
              else
# get the ga-epr setting
               if (index($0,gaeprstr))
                 {
                   split($0,arr,":");
                   gaepr = arr[2] * 1000;
                 }
               else
# get the initial runway
```

```
# nb. assumption is that the setting value
# will have a space in front of it eg.
# Initial-runway: APP-28R-IN
             if (index($0,runwaystr))
              {
               split($0,arr," ");
               runway = arr[2];
               split(runway,arr,"-");
               if (length(arr[2]) == 2)
                arr[2] = "0" arr[2];
               runway = arr[2];
              }
# get the new runway from K18-IT: or FAF-IT: entry
# nb. assumption is that the setting value
# will have a space in front of it eg.
# K18-IT: RWY-10L
             else
              if ((index($0,newrwystr)) || (index($0,newmafresstr)))
               {
                if(index($0,"RWY"))
                  {
                   split($0,arr," ");
                   newrunway = arr[2];
                   split(newrunway,arr,"-");
                   if(length(arr[2]) == 2)
                     arr[2] = "0" arr[2];
                   newrunway = arr[2];
                  }
                }
# substitute the frequency values in the place of
# their stubs in the tod procedure array
      if (init == 1)
        {

# substitute the new runway for its stub in itnraevt.set
# which is in the nrk and nrf
         if((legs["nra"] == 0) && (newrunway != ""))
           {
            for(i=1;i<=itnrasize;i++)
             if(index(nrk[i],"{newrunway}"))
              {
               split(nrk[i],arr,"{");
               temp = arr[1];
               temp1 = arr[2];
               split(temp1,arr,"}");
               temp1 = arr[2];
               nrk[i] = temp newrunway temp1;
               nrf[i] = temp newrunway temp1;
              }
            legs["nra"] = 1;
           }
# substitute the runway for its stub in itirvevt.set
#  which is in the irv array and itiraevt.set which is
#  is in the ira array
         if((legs["irv"] == 0) && (runway != ""))
           {
            for(i=1;i<=itirvsize;i++)
             if(index(irv[i],"{initrwy}"))
              {
               split(irv[i],arr,"{");
               temp = arr[1];
               temp1 = arr[2];
               split(temp1,arr,"}");
               temp1 = arr[2];
               irv[i] = temp runway temp1;
```

```
               ira[i] = temp runway temp1;
              }
           legs["irv"] = 1;
          }

         if((legs["tod"] == 0) && (atisfreq != 0))
          {
           for (i=1; i<=todarrsize ; i++)
             if (index(todarr[i],"freq}"))
              {
               split(todarr[i],arr,"=");
               if(arr[2] == "{afreq}")
                todarr[i] = arr[1]"=" sprintf("%1.2f",atisfreq);
               else
                if(arr[2] == "{cfreq}")
                  todarr[i] = arr[1]"="  sprintf("%1.2f",companyfreq);
                else
                  if(arr[2] == "{tfreq}")
                   todarr[i] = arr[1]"=" sprintf("%1.2f",towerfreq);
              }
           legs["tod"] = 1;
          }
# substitute the altimeter value, the braking value and
# the altimeter digits in place of their stubs in the
# k18 procedure array
         if ((legs["k18"] == 0) && (altimeter != 0))
          {
           for (i=1; i<=k18arrsize ; i++)
             {
# not worried about to whom subject talks
              if (index(k18arr[i],"TALK-TO") || index(k18arr[i],"TALK-
STOPPED-TO"))
               {
                split(k18arr[i],arr,"=");
                k18arr[i] =  k18arr[i];
# may caused legitimate talk to be discounted ..arr[1];
               }
              else
               if (index(k18arr[i],"{altim}"))
                {


                  split(k18arr[i],arr,"{");
                  temp = arr[1];
                  split(arr[2],arr,"}");
                  k18arr[i] = temp (altimeter / 100.0) arr[2];
                }
#         else
#          if (index(k18arr[i],"{braking}")){
#            split(k18arr[i],arr,"=");
#            k18arr[i] = arr[1]"="braking;
#          }
               else
                if (index(k18arr[i],"{altdig"))
                  {
                   split(k18arr[i],arr,"{");
                   if(index(arr[2],"dig1"))
                     k18arr[i] = arr[1]sprintf("%d",(altimeter/1000))"  )";
                   else
                     if(index(arr[2],"dig2"))
                      k18arr[i] = arr[1]sprintf("%d",((altimeter/100)%10))"
)";
                     else
                      if(index(arr[2],"dig3"))
                        k18arr[i] =
arr[1]sprintf("%d",((altimeter%100)/10))"  )";
```

```
                         else
                           if(index(arr[2],"dig4"))
                             k18arr[i] =
arr[1]sprintf("%d",((altimeter%100)%10))"  )";
                        }
                  }
                legs["k18"] = 1;
               }
# substitute the epr value and the epr digits in the
# place of their stubs if the faf procedure array
          if ((legs["faf"] == 0) && (gaepr != 0))
            {
             for (i=1; i<=fafarrsize ; i++)
               {
                if (index(fafarr[i],"TALK-TO") || index(fafarr[i],"TALK-
STOPPED-TO"))
                  {
                   split(fafarr[i],arr,"=");
                   fafarr[i] =  fafarr[i];
#  may caused legitimate talk to be discounted ..arr[1];
                  }
                else
                  if (index(fafarr[i],"{ga-epr}"))
                    {
                     split(fafarr[i],arr,"{");
                     temp = arr[1];
                     split(arr[2],arr,"}");
                     fafarr[i] = temp (gaepr / 1000.0) arr[2];
                    }
                  else
                    if (index(fafarr[i],"{gadig"))
                      {
                       split(fafarr[i],arr,"{");
                       if(index(arr[2],"dig1"))
                         fafarr[i] = arr[1]sprintf("%d",(gaepr/1000))"  )";
                       else
                         if(index(arr[2],"dig2"))
                           fafarr[i] = arr[1]sprintf("%d",((gaepr/100)%10))"  )";
                         else
                           if(index(arr[2],"dig3"))
                             fafarr[i] = arr[1]sprintf("%d",((gaepr%100)/10))"  )";
                           else
                             if(index(arr[2],"dig4"))
                               fafarr[i] = arr[1]sprintf("%d",((gaepr%100)%10))"
)";
                      }
                legs["faf"] = 1;
               }
            }

          if ((legs["tod"] != 0) && (legs["k18"] != 0) && (legs["faf"] !=
0))
            {
             if (DBG == 0)
               {
                for (i=1; i<=resumarrsize ; i++)
                  printf("\n%s", resumarr[i] );
                printf("\nTOD events :");
                for (i=1; i<=todarrsize ; i++)
                  printf("\n%s", todarr[i] );
                printf("\n\nK18 events");
                for (i=1; i<=k18arrsize ; i++)
                  printf("\n%s", k18arr[i] );
                printf("\n\nFAF events");
                for (i=1; i<=fafarrsize ; i++)
```

```
                  printf("\n%s", fafarr[i] );
                printf("\nHold Path K18 interrupt events");
                for (i=1; i<=ithpasize;i++)
                  printf("\nhpk[%d] = %s",i,hpk[i]);
                printf("\nHold Path FAF interrupt events");
                for (i=1; i<=ithpasize;i++)
                  printf("\nhpf[%d] = %s",i,hpf[i]);
                printf("\nNew Ruway K18 interrupt events");
                for (i=1; i<=itnrasize;i++)
                  printf("\nnrk[%d] = %s",i,nrk[i]);
                printf("\nNew Ruway FAF interrupt events");
                for (i=1; i<=itnrasize;i++)
                  printf("\nnrf[%d] = %s",i,nrf[i]);
                printf("\nChange Speed K18 interrupt events");
                for (i=1; i<=itcsasize;i++)
                  printf("\ncsk[%d] = %s",i,csk[i]);
                printf("\nChange Speed FAF interrupt events");
                for (i=1; i<=itcsasize;i++)
                  printf("\ncsf[%d] = %s",i,csf[i]);
                printf("\nChange Altitude K18 interrupt events");
                for (i=1; i<=itcaasize;i++)
                  printf("\ncak[%d] = %s",i,cak[i]);
                printf("\nChange Altitude FAF interrupt events");
                for (i=1; i<=itcaasize;i++)
                  printf("\ncaf[%d] = %s",i,caf[i]);
                printf("\nInitRun AUD interrupt events");
                for (i=1; i<=itirasize;i++)
                  printf("\nira[%d] = %s",i,ira[i]);
                printf("\nInitRun VIS interrupt events");
                for (i=1; i<=itirvsize;i++)
                  printf("\nirv[%d] = %s",i,irv[i]);
               }
             init = 2;
            }
        }


# get time

      split($1,arr,",");
      time = arr[1];
      isprocevent = 0;
      isintevent = 0;
      if (index($0,wyptpass))
        {
         leg = 0;
# passing waypoint - get number
         split($0,arr,"=");
         waypt = arr[2];
         split(waypt,arr," ");
         waypt = arr[1];
# passing waypoints that begin a procedural leg
         if((waypt == 1) || (waypt == 3) || (waypt == 6))
           {
# set leg variable and set up procedural
# count array
             if(waypt == 1)
               {
                leg = 1;
                for (i=1; i<=todarrsize ; i++){
                               proccntarr[i] = -1;
                               procstrarr[i] = todarr[i];
                              }
                procarrsize = todarrsize;
               }
             else
```

```
                if(waypt == 3)
                  {
                    leg = 2;
                    for (i=1; i<=k18arrsize ; i++)
                     {
                       proccntarr[i] = -1;
                       procstrarr[i] = k18arr[i];
                     }
                    procarrsize = k18arrsize;
                  }
                else
                  if(waypt == 6)
                    {
                     leg = 3;
                     for (i=1; i<=fafarrsize ; i++)
                       {
                         proccntarr[i] = -1;
                         procstrarr[i] = fafarr[i];
                       }
                     procarrsize = fafarrsize;
                    }
              passwayptstart = time;
             }
           else

# passing waypoints that end a procedural leg
             if((waypt == 2) || (waypt == 4) || (waypt == 7))
              {
                passwayptstop = time;
# error checking here is basic : if the procedure is not
# demarcated with a start and end between waypoint
# boundaries, both the onset and performance times are flagged
                onsettime = errornumber;
                performancetime = errornumber;
                if (estarttime != -1)
                  {
                    if (passwayptstart <= estarttime)
                     {
# possible to calculate the onset time
                       onsettime = estarttime - passwayptstart;
                       ensembleonsettime = onsettime;
                       ensemblestarttime =  estarttime;
                     }
                    else
                     if ((DBG == 6) || (DBG == 7))
                       if (passwayptstart == 0)
                         printf("\nNo Passing Start Wypt Marker");
                       else
                         printf("\nStart Marker before Passing Wypt Marker ");
# possible to calculate the performance time
                    if ((eendtime >= estarttime) && (eendtime <=
passwayptstop))
                     {
                       performancetime = eendtime - estarttime;
                       ensembleendtime  = eendtime;
                       ensembleperftime = performancetime;
                     }
                    else
                     if ((DBG == 6) || (DBG == 7))
                       {
                         if (eendtime == -1)
                           printf("\nNo procedure END Marker");
                         else {
                               if (eendtime < estarttime)
                                 printf("\nProcedure END time < START time");
```

```
                              if (eendtime > passwayptstop)
                                printf("\nProcedure ended after stop waypt
crossed");
                            }
                        if (passwayptstop == 0)
                          printf("\nNo Passing End Wypt Marker");
                       }
                  }
                else
                  if ((DBG == 6) || (DBG == 7))
                    printf("\nNo procedure START marker");
# calculate the initialization, acknowledgement
# and performance time of leg interrupt

# first get the last two digits of the run condition
                split($6,arr,".");
                temp = arr[2];
                split(temp,arr,">");
                temp = arr[1];
# if there is no IT marker,set value of deltas
# according to to run condition
                if(interrupts["IT-MARK-TIME"] == 0)
                  {
                    if (eendtime != -1)
                     if (ensemblettd != errornumber)
                       ensemblettd = ensemblettd - eendtime;
                    iterrorno = -999;
                    if ((temp >= 11) && (temp <= 12))
                     {
                       resumevclass = -999;
                       procresumetime = -999;
                       itackntime = errornumber;
                       itstarttime = errornumber;
                       itperftime =  errornumber;
                       itmarktoexc = -999;
                       excresumetime = -999;
                       excresumeclass = -999;
                       exctoprocfpmcount = -999;
                     }
                    else{
                        itackntime = -999;
                        itstarttime = -999;
                        itperftime = -999;
                       }
                  }
                else
                  {
                    if (temp == 10)
                     {
                       resumevclass = -888;
                       procresumetime = -888;
                       inttoprocfpmcount = -888;
                       excresumetime = -888;
                       excresumeclass = -888;
                       exctoprocfpmcount = -888;
                     }
# calculate IT acknowledgement time
# use this clause to calculate the IT error
# form number per (2.8 of specs)
                    if(interrupts[interrupts["IT-MARK-EVENT"] "=ACKN"]
== 0)
                     {
                       itackntime = errornumber;
# did not acknowledge but started
```

```
                  if(interrupts[interrupts["IT-MARK-EVENT"]
"=START"] != 0)
                    {
# did not acknowledge but started and ended
                     if (interrupts[interrupts["IT-MARK-EVENT"]
"=END"] != 0)
                      {
# did not acknowledge but started and ended: start < finish
                        if (interrupts[interrupts["IT-MARK-EVENT"]
"=END"] > interrupts[interrupts["IT-MARK-EVENT"] "=START"])
                          iterrorno = 2;
                      }
# did not acknowledge started but did not finish
                     else
                      iterrorno = 6;
                    }
                  else
# did not acknowledge did not start or end
                   if (interrupts[interrupts["IT-MARK-EVENT"] "=END"]
== 0)
                     {
                      iterrorno = 7;
                      ensembleperftime = -888;
                     }
                 }
                else
                 {
                  itackntime = interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] - interrupts["IT-MARK-TIME"];
# acknowledged and started
                  if(interrupts[interrupts["IT-MARK-EVENT"]
"=START"] != 0)
                    {
# acknowledged and started but not ended : acknowledge < start
                     if(interrupts[interrupts["IT-MARK-EVENT"]
"=END"] == 0)
                      if (interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] < interrupts[interrupts["IT-MARK-EVENT"] "=START"])
                        iterrorno = 4;
                      else
# acknowledged and started but not ended : acknowledge > start
                        iterrorno = 5;
                     else
# acknowledged, started and ended: start < finish
                      if(interrupts[interrupts["IT-MARK-EVENT"]
"=END"] > interrupts[interrupts["IT-MARK-EVENT"] "=START"])
# acknowledged, started and ended: start < acknowledge  < finish
                        if(interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] > interrupts[interrupts["IT-MARK-EVENT"] "=START"])
                          iterrorno = 1;
                        else
# acknowledged, started and ended: acknowledged < start < finish
                          iterrorno = 0;
                    }
                  else
# acknowledged but did not start or finish
                   if(interrupts[interrupts["IT-MARK-EVENT"] "=END"]
== 0)
                     {
                      iterrorno = 3;
                      ensembleperftime = -888;
                     }
                 }
# calculate IT initiation time
```

```
                if (interrupts[interrupts["IT-MARK-EVENT"] "=START"]
== 0)
                 {
                  itstarttime = -888;
#                 ensembleperftime = -888;
                 }
                else
                 {
                  itstarttime = interrupts[interrupts["IT-MARK-EVENT"]
"=START"] - interrupts["IT-MARK-TIME"];
# calculate ensemble onset time if not already set
# or if interruption started before procedure
                  if (ensembleonsettime == errornumber)
                    {
                     if (temp == 2)
                      if ((interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] != 0) &&\
                          (interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] < \
                           interrupts[interrupts["IT-MARK-EVENT"]
"=START"]))
                        ensembleonsettime = interrupts[interrupts["IT-
MARK-EVENT"] "=ACKN"]  - passwayptstart;
                      else
                        ensembleonsettime = interrupts[interrupts["IT-
MARK-EVENT"] "=START"] - passwayptstart;
                    }
                  else
                    {
                     if ((interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] != 0) &&\
                        (interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] < \
                         interrupts[interrupts["IT-MARK-EVENT"]
"=START"]))
                      ensembleonsettime = interrupts[interrupts["IT-
MARK-EVENT"] "=ACKN"];
                     else
                      ensembleonsettime = interrupts[interrupts["IT-
MARK-EVENT"] "=START"];
                     if (estarttime <  ensembleonsettime)
                      ensembleonsettime = estarttime - passwayptstart;
                     else
                      ensembleonsettime -= passwayptstart;
                    }
                 }

# calculate IT performance time (end - mark)
                if (interrupts[interrupts["IT-MARK-EVENT"] "=END"] ==
0)
                 {
                  itperftime = -888;
#                 ensembleperftime = -888;
                  if (procresumetime == -1){
                    resumevclass = -777;
                    procresumetime = -777;
                  }
                  if (ensemblettd != errornumber)
                    if (eendtime != -1)
                      ensemblettd = ensemblettd - eendtime;
                 }
                else
                 {
                  itperftime = interrupts[interrupts["IT-MARK-EVENT"]
"=END"] - interrupts["IT-MARK-TIME"];
```

```
                    if (itstarttime != -888)
# calculate the ensemble performance time
                     if (ensembleperftime < 0)
                      {
# something was wrong with the procedural times
                       if ((interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] != 0) &&\
                          (interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] < \
                           interrupts[interrupts["IT-MARK-EVENT"]
"=START"]))
                         ensemblestarttime = interrupts[interrupts["IT-
MARK-EVENT"] "=ACKN"];
                       else
                         ensemblestarttime = interrupts[interrupts["IT-
MARK-EVENT"] "=START"];
                       ensembleendtime  = interrupts[interrupts["IT-MARK-
EVENT"] "=END"];
                       if (ensembletd != errornumber)
                        {
#           printf("\nwhen waypt is %d subtracting %1.3f from tod time
%1.3f",\
#                        waypt, ensembleendtime, ensembletd);
                          ensembletd = ensembletd - ensembleendtime;

                        }
                       else
#                   printf("\nwhen waypt is %d tod time= %1.3f",\
#                        waypt,ensembletd);
                      }
                     else
                      {
# get earlier start and later end
                       if ((interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] != 0) &&\
                          (interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] < \
                           interrupts[interrupts["IT-MARK-EVENT"]
"=START"]))
                         ensemblestarttime = interrupts[interrupts["IT-
MARK-EVENT"] "=ACKN"];
                       else
                         ensemblestarttime = interrupts[interrupts["IT-
MARK-EVENT"] "=START"];
                       if (estarttime <= ensemblestarttime)
                         ensemblestarttime = estarttime;
                       if (eendtime >= interrupts[interrupts["IT-MARK-
EVENT"] "=END"])
                         ensembleendtime = eendtime;
                       else
                         ensembleendtime = interrupts[interrupts["IT-MARK-
EVENT"] "=END"];
                       ensembleperftime =  ensembleendtime -
ensemblestarttime;
                       if (ensembletd != errornumber)
                        {
                          ensembletd = ensembletd - ensembleendtime;
#                  printf("\nwhen waypt is %d subtracting %1.3f from tod
time %1.3f",\
#                        waypt, ensembleendtime, ensembletd);
                        }
                       else
#                   printf("\nwhen waypt is %d tod time= %1.3f",\
#                        waypt,ensembletd);
                      }
```

```
                    } #else interrupt has ended
                     if ((itperftime > 0) && (itstarttime > 0))
                       itstarttoend = itperftime - itstarttime;
                   if ((itackntime > 0) && (itstarttime > 0))
                    itackntostart =itstarttime - itackntime;
                  } # else - there was an it-marker
                   if(waypt == 2)
                    prevleg = 1;
                   else
                    if(waypt == 4)
                     prevleg = 2;
                    else
                     if(waypt == 7)
                      prevleg = 3;
                 ensemblefpmcount = count1 + count2;
# if one or the other (proc or int) did not start reduce by count2
                 if (ensemblefpmcount > 0)
                  if ((estarttime == -1) || ((interrupts[interrupts["IT-MARK-
EVENT"] "=START"] == 0) && \
                             (interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"] == 0)))
                   ensemblefpmcount -=  count2;
                 if ((DBG == 14) || (DBG == 15) || (DBG == 16) || (DBG ==
17))
                  {
                   t[1] = estarttime;
                   l[1] = "Proc Start Time = ";
                   t[2] = eendtime;
                   l[2] = "Proc End Time   = ";
                   t[3] = interrupts[interrupts["IT-MARK-EVENT"]
"=START"];
                   l[3] = "Int Start Time  = ";
                   t[4] = interrupts[interrupts["IT-MARK-EVENT"]
"=END"];
                   l[4] = "Int End Time    = ";
                   t[5] = passwayptstart;
                   l[5] = "Proc Mark Time  = ";
                   t[6] = interrupts["IT-MARK-TIME"];
                   l[6] = "Int Mark Time   = ";
                   l[7] = "Proc Resum Time = ";
                   if (procresumetime == -1)
                    t[7] = procresumetime;
                   else
                    t[7] =  procresumetime + interrupts[interrupts["IT-MARK-
EVENT"] "=END"];
                   t[8] = interrupts[interrupts["IT-MARK-EVENT"]
"=ACKN"];
                   l[8] = "Int ACKN Time   = ";
                   t[9] = ensembleonsettime + passwayptstart;
                   l[9] = "Ens onset   = ";
                   l[10] = "Hit EXC    = ";
                   t[10] = intexc;

                   if (DBG == 17)
                    {
                     printf("\nensemble fpm count : %d + %d = %d",
count1,count2, count1 + count2);
                     if (ensemblefpmcount > 0)
                      if (estarttime == -1)
                       printf("\nshould reduce ensemble fpm by %d because
procedure did not start",\
                              count2);
                     if (interrupts[interrupts["IT-MARK-EVENT"]
"=START"] == 0)
```

```
               printf("\nshould reduce ensemble fpm by %d because
interruption did not start",\
                  count2);
            }
            printf("\nEnsemble Times");
            if (interrupts["IT-MARK-EVENT"] == 0)
            {
           printf("\nproc start time = %1.2f",estarttime);
           printf("\nProc End   time = %1.2f",eendtime);
           print("\nNo Interrupt");
          }
        else{
# sort arrays
            for(i=10;i>1;i--)
             for(j=1;j<i;j++)
              if (t[j] > t[j+1])
               {
                temp = t[j];
                t[j] = t[j+1];
                t[j+1] = temp;
                temp = l[j];
                l[j] = l[j+1];
                l[j+1] = temp;
               }
            for(i=1;i<=10;i++)
             printf("\n%s%1.2f",l[i],t[i]);
           }
        }
       if  ((DBG == 8) || (DBG == 9) ||  (DBG == 10) || (DBG == 11))
        if (interrupts["IT-MARK-EVENT"] != 0)
         {
          printf("\nFor event %s",interrupts["IT-MARK-EVENT"]);
          printf("\nMARKER time      = %1.3f\nACKNOWLEDGE
time = %1.3f",\
             interrupts["IT-MARK-TIME"],\
             interrupts[interrupts["IT-MARK-EVENT"] "=ACKN"]);
          printf("\nSTART time       = %1.3f\nEND time         =
%1.3f",\
             interrupts[interrupts["IT-MARK-EVENT"] "=START"],\
             interrupts[interrupts["IT-MARK-EVENT"] "=END"]);
         }
       split($6,arr,"<");
       temp = sprintf("<%s", arr[2]);
       totalerrcount = 0;
       totmissnt = 0;
       totordcnt = 0;
       totvalcnt = 0;
       totextcnt = 0;
       if (procarrsize > 0){
# output the procedural infomation
        if ((DBG == 18) || (DBG == 19) || (DBG == 20) || (DBG ==
22))
         printf("\n***Results for leg %d procedural events",prevleg);
        misscount = 0;
        prevevent = 0;
        ordercount = 0;
        valok = 1;
        if (prevleg == 2)
         valok = altimok;
        else
         if (prevleg == 3)
          valok = eprok;
        for(i=1;i<=procarrsize;i++){
         if ((prevleg == 2) || (prevleg == 3)){
```

```
          if ((index(procstrarr[i],"<RADIO>:TRANSMIT-
SELECT=COMPANY")) &&\
            ((index(procstrarr[i - 1],"<RADIO>:TRANSMIT-
SELECT=VHF3")) ||\
            (index(procstrarr[i + 1],"<RADIO>:TRANSMIT-
SELECT=VHF3"))) &&\
            (proccntarr[i] == -1))
           continue;
          else
           if (index(procstrarr[i],"<RADIO>:TRANSMIT-
SELECT=VHF3") &&\
             (proccntarr[i] == -1))
            continue;
         }
         if ((DBG == 18) || (DBG == 19) || (DBG == 20) || (DBG ==
22))
          printf("\npr[%-69s] = %3s should be %d",\
             procstrarr[i],proccntarr[i],i);
# if there are more than one entry just use the first
         if(length(proccntarr[i]) > 2){
          split(proccntarr[i],arr," ");
          temp1 = arr[1];
         }
         else
          temp1 = proccntarr[i];

         if (temp1 == -1)
          misscount +=1;
         else{
          if (temp1 < prevevent)
           ordercount += 1;
          prevevent = temp1;
         }
        }
        if ((DBG == 18) || (DBG == 19) || (DBG == 20) || (DBG ==
22)){
         printf("\nThere were %d omissions %d order errors %d value
errors and the following %d extraneous events:",\
            misscount,ordercount, 1 - valok,extranproccount);
         for(i=1;i<=extranproccount;i++)
          printf("\n%s",exprocarr[i]);
         printf("\n");
        }
        totalerrcount = misscount + ordercount + extranproccount;
        if (valok == 0)
         totalerrcount += 1;
        totmissnt = misscount;
        totordcnt = ordercount;
        totvalcnt = 1 - valok;
        totextcnt = extranproccount;
       }
       if (intsize == 0)
        {
         iterrortotal = errornumber ;
         itselerr = errornumber;
         itexerr = errornumber;
         itleftintearly = errornumber;
# intsize is 0 when there was no IT-MARKER
         if ((substr(temp,5,2) != 11) && (substr(temp,5,2) != 12))
          itmarkererror = 1;
         else
          itmarkererror = 0;

        }
       if (intsize != 0){
```

```
# output the interrupt infomation
        misscount = 0;
        prevevent = 0;
        ordercount = 0;
        valok = 1;
        if (prevleg == 3){
          if (index(interrupts["IT-MARK-EVENT"],"FAF-CHGSPD-
AUD"))
             valok = fafspdok;
          else
            if (index(interrupts["IT-MARK-EVENT"],"FAF-CHGALT-
AUD"))
             valok = fafaltok;
         }
        else
         if (prevleg == 2){
          if (index(interrupts["IT-MARK-EVENT"],"K18-CHGSPD-
AUD"))
             valok = k18spdok;
          else
            if (index(interrupts["IT-MARK-EVENT"],"K18-CHGALT-
AUD"))
             valok = k18altok;
         }
# set the itselerr from changes document 1/31/96
        if (index(interrupts["IT-MARK-EVENT"],"CHGSPD-AUD") ||\
           index(interrupts["IT-MARK-EVENT"],"CHGALT-AUD"))
         itselerr = 1 - valok;
# pilot went to legs page before typing EXC
        if (itleftintearly == -1){
         itleftintearly = 0;
         if (((intcntarr[intsize] != -1) && (intcntarr[intsize - 1] != -1))
&&\
            (intcntarr[intsize] < intcntarr[intsize - 1]))
          itleftintearly = 1;
         }
        if ((DBG == 18) || (DBG == 19) || (DBG == 20) || (DBG ==
22))
         printf("\n***Results for leg %d interrupt
%s",prevleg,interrupts["IT-MARK-EVENT"]);
        for(i=1;i<=intsize;i++){
         if ((index(interrupts["IT-MARK-EVENT"],"FAF-HOLDPAT-
AUD") ||\
            index(interrupts["IT-MARK-EVENT"],"FAF-CHGSPD-
AUD") ||\
            index(interrupts["IT-MARK-EVENT"],"FAF-CHGALT-
AUD")) &&\
            index(intstrarr[i],"NXT PAGE"))
           continue;
         if (index(interrupts["IT-MARK-EVENT"],"INITRUN-VIS")
&&\
            index(intstrarr[i],"<DATALINK>:FROM=(MESSAGE)-
TO=(MM-ROGER)") &&\
            (intcntarr[i] == -1) &&  rogernotrequired)
           continue;
# set flag for itexerr from changes  document  1/31/96
         if (index(intstrarr[i],"<FMS>:PAGE=(") &&\
            index(intstrarr[i],"-TYPED=(EXC"))
           if (intcntarr[i] == -1)
            itexerr = 1;
           else
            itexerr = 0;
         if ((DBG == 18) || (DBG == 19) || (DBG == 20) || (DBG ==
22))
           printf("\nir[%-69s] = %3s should be %d",\
                   intstrarr[i],intcntarr[i],i);
# if there are more than one entry just use the first
           if(length(intcntarr[i]) > 2){
             split(intcntarr[i],arr," ");
             temp1 = arr[1];
           }
           else
             temp1 = intcntarr[i];

           if (temp1 == -1)
             misscount +=1;
           else{
             if (temp1 < prevevent)
               ordercount += 1;
             prevevent = temp1;
            }
          }
          if ((DBG == 18) || (DBG == 19) || (DBG == 20) || (DBG ==
22)){
           printf("\nThere were %d omissions %d order errors and %d
value errors",\
                  misscount,ordercount,1 - valok);
           printf("\n");
          }

          totalerrcount += misscount + ordercount;
          if (valok == 0)
            totalerrcount += 1;
          totmissnt += misscount;
          totordcnt += ordercount;
          totvalcnt += 1 - valok;
        }

        if((interrupts[interrupts["IT-MARK-EVENT"] "=END"] == 0)
&&\
           (interrupts["IT-MARK-EVENT"] != 0))
           exctoendcount = -777;


# if itmarktoexc >  itperftime then the EXC is after
# the transition to LEGD - error
             if((intexc == -1) && (interrupts["IT-MARK-TIME"] !=0))
               itmarktoexc = errornumber;

             if ((itperftime > 0) && (itmarktoexc > itperftime))
               exctoendcount = -666;
# if there is no ending interrupt or ending
# procedural times - time to deadline is an error
             if ((interrupts[interrupts["IT-MARK-EVENT"] "=END"] ==
0) &&\
                 (eendtime == -1))
               ensemblettd = -666;


# reset array of interrupt times


        for (item in interrupts)
          interrupts[item] = 0;
# set interrupt acknowledge start and finish omission and order flags
        if (iterrorno == -1)
          {
           iterrorno = errornumber;
           itaorder  = errornumber;
           itaomit   = errornumber;
```

```
                itsomit   = errornumber;
                itfomit   = errornumber;
               }
            else
              if (iterrorno == 0)
                {
                  itaorder = 0;
                  itaomit = 0;
                  itsomit = 0;
                  itfomit = 0;
                }
              else
                if (iterrorno == 1)
                {
                  itaorder = 1;
                  itaomit = 0;
                  itsomit = 0;
                  itfomit = 0;
                }
                else
                  if (iterrorno == 2)
                    {
                      itaorder = 0;
                      itaomit = 1;
                      itsomit = 0;
                      itfomit = 0;
                    }
                  else
                    if (iterrorno == 3)
                      {
                       itaorder = 0;
                       itaomit = 0;
                       itsomit = 1;
                       itfomit = 1;
#                      itexerr = 0;
                      }
                    else
                      if (iterrorno == 4)
                       {
                         itaorder = 0;
                         itaomit = 0;
                         itsomit = 0;
                         itfomit = 1;
                       }
                      else
                       if (iterrorno == 5)
                         {
                           itaorder = 1;
                           itaomit = 0;
                           itsomit = 0;
                           itfomit = 1;
                         }
                       else
                         if (iterrorno == 6)
                           {
                             itaorder = 0;
                             itaomit = 1;
                             itsomit = 0;
                             itfomit = 1;
                           }
                         else
                           if (iterrorno == 7)
                             {
                              itaorder = 0;
                              itaomit = 1;
```

```
                              itsomit = 1;
                              itfomit = 1;
#                             itexerr = 0;
                             }
                           else
                             if (iterrorno == -999)
                              {
                                itaorder = -999;
                                itaomit = -999;
                                itsomit = -999;
                                itfomit = -999;
                              }
            if (iterrorno == -999)
              iterrortotal = -999;
            else
              {
                if ((iterrorno >= 0) || (itselerr >= 0) || (itexerr >= 0) ||
(itleftintearly >= 0))
                  {
                    iterrortotal = 0;
                    if (iterrorno >= 0)
                      iterrortotal = itaorder + itaomit + itsomit + itfomit;
                    if (itleftintearly >= 0)
                      iterrortotal += itleftintearly;
                    if (itselerr >= 0)
                      iterrortotal += itselerr;
                    if (itexerr >= 0)
                      iterrortotal += itexerr;
                  }
              }


            if (itselerr == -1)
              itselerr = errornumber;
            if (itexerr == -1)
              itexerr = errornumber;

            if (excresumetime < 0)
              if  (excresumetime == -1)
                {
                  excresumetime = errornumber;
                  excresumeclass = errornumber;
                  exctoprocfpmcount = errornumber;;
                }
              else
                {
                  excresumeclass =  excresumetime;
                  exctoprocfpmcount = excresumetime ;
                }

            if (procresumetime < 0)
              if  (procresumetime == -1)
                {
                  resumevclass =  errornumber;
                  inttoprocfpmcount = errornumber;
                  procresumetime = errornumber;
                }
              else
                {
                  resumevclass =  procresumetime;
                  inttoprocfpmcount = procresumetime;
                }
            if (ensembleperftime < 0)
              ensemblefpmcount = ensembleperftime;
```

```
             printf("\n%2d, %5d, %5d, %4d, %5d, %3d, %6s,   %7.2f,
%7.2f, %7.2f, %7.2f, %7.2f, %7.2f, %7.2f, %5d, %7.2f, %4d, %7.2f,
%8.2f, %8.2f, %4d, %6d,",\

subj_number,runlist,block_number,seg_number,run_number,prevleg,sub
str(temp,2,5),onsettime,performancetime,\

itackntime,itstarttime,itperftime,itackntostart,itstarttoend,iterrorno,procre
sumetime,resumevclass,\

ensembleonsettime,ensembleperftime,ensemblettd,ensemblefpmcount,int
toprocfpmcount);
         printf(" %8d, %6d, %5d, %6d, %6d,", totalerrcount,totmissnt,
totordcnt,totvalcnt,totextcnt);
         printf("  %7d,  %9d,  %9d, %11d, %9d, %8d, %7d,
",itaorder,itaomit,itsomit,itfomit,itselerr,itexerr,iterrortotal);
         printf(" %6s, %6s, %6s, %6s, %6s, %6s, %6s, %6d, %10d,
%12d,",\
              intarr[temp],\
              goalarr[temp],\
              ipmodarr[temp],\
              itmodarr[temp],\
              wdeadarr[temp],\
              couparr[temp],\
              relarr[temp],\
              itmarkererror,\
              exctoendcount,\
              itleftintearly);
             printf(" %10.2f, %7.2f, %8d, %8d, %8d,",\
                  itmarktoexc,\
                  excresumetime,\
                  excresumeclass,\
                  exctoprocfpmcount,\
                  inttwolegs );
# reset some times to default values
             inttwolegs = 0;
             inttoprocfpmcount = 0;
             procresumetime = -1;
             iterrorno = -1;
         resumevclass = -1;
         ensemblettd = errornumber;
         ensembleonsettime =  errornumber;
         count1 = 0;
         count2 = 0;
         countevent = 0;
         eendtime = -1;
         estarttime = -1;
         passwayptstart = 0;
         passwayptstop = 0;
         itselerr = -1;
         itexerr = -1;
         intsize = 0;
         intcount = 0;
         extranproccount = 0;
         startexctoendcount = -1;
         exctoendcount = errornumber;
         procarrsize = 0;
         proccount = 0;
         itmarkererror = 0;
         itaorder = -1;
         itaomit = -1;
         itsomit = -1;
         itfomit = -1;
         itleftintearly = -1;
```

```
         ensembleperftime = errornumber;
         ensemblestarttime = -1;
         ensembleendtime = -1;
         itstarttoend = errornumber;
         itackntostart = errornumber;
         itackntime = -1;
         itstarttime = -1;
         itperftime = -1;
         ensemblefpmcount = -1;
         iterrortotal = -1;
         intexc = -1;
         itmarktoexc = -1;
         excresumetime = -1;
         excresumeclass = -1;
             exctoprocfpmcount = -1;
           }
        }

      else
# this else refers to  if (index($0,wyptpass))
# ie. we are now not at a waypoint passing statement
        {
         if((leg == 1) || (leg == 2) || (leg == 3))
           {
# check for a procedural event, log it if found
# get a superset of all of the procedural events.
           if((index($0,"<RADIO>")) || (index($0,"<FMS>")) ||\
              (index($0,"<OVERHEAD>")) ||
(index($0,"<CHECKLIST>")) ||\
               ((index($0,"<ENERGY-CTRL>")) &&\
                !(index($0,"<ENERGY-CTRL>:STICK")) &&\
                !(index($0,"THROTTLE"))) ||\
              (index($0,"SPEEDBRAKES")) ||\
              (index($0,"<DATALINK>")))
              {
               found = 0;
               i = 1;
               duplicate = 0;

# bump the counter keeping track of events between typeing of EXC
# and end of interrupt
               if (startexctoendcount == 1)
                 if (!(index($0,"<ENERGY-CTRL>"))){
                  exctoendcount++;
#                printf("\ntime = %1.3f event = %s count = %d",time,$2,
exctoendcount);
               }

# get the go around epr value typed in
               if ((leg == 3) &&\
                  (index($2,"<FMS>:PAGE=(N1 LIMIT;1)-TYPED=(")))
                 {
                  split($2,arr,"(");
                  temp = arr[3];
                  split(temp,arr," ");
                  if (index(arr[1],"CL1"))
                    {
                     if (eprcnt >= 0)
                       {
                        eprcnt -= 1;
                        eprok = 0;
                       }
                    }
                  else
                    {
```

```
               eprcnt += 1;
               tempepr[eprcnt] = arr[1];
               if (eprcnt == eprdigits)
                {
# there are sufficient digits - check the value
                 temp =  tempepr[1];
                 for (j=2;j<=eprdigits;j++)
                  {
                   if(j == 2)
                    {
# the decimal point must be in position 2
                     if (tempepr[j] == ".")
                      eprok = 1;
                    }
                   temp = temp tempepr[j] ;
                  }
                 if (eprok && (temp != gaepr / 1000.0))
                  eprok = 0;
                 if (DBG == 21)
                  if (!eprok)
                   printf("\ntyped in wrong gaepr value = %s instead
of %1.3f",temp, gaepr / 1000.0);
                  else
                   printf("\ntyped in correct gaepr value = %s",temp);
                }
               else
                eprok = 0;
              }
            }
# get the altimeter value typed in
# value should equal altimeter value in file header
           if ((leg == 2) &&\
             (index($2,"<FMS>:PAGE=(PRF INIT;1)-TYPED=(")))
            {
             split($2,arr,"(");
             temp = arr[3];
             split(temp,arr," ");
             if (index(arr[1],"CL1"))
              {
               if (altimcnt >= 0)
                {
                 altimcnt -= 1;
                 altimok = 0;
                }
              }
             else{
                altimcnt += 1;
                tempaltimeter[altimcnt] = arr[1];
                if (altimcnt == altimdigits)
                 {
                  temp = tempaltimeter[1];
                  for (j=2;j<= altimdigits;j++)
                   {
                    if (j == 3)
                     {
                      if (tempaltimeter[j] == ".")
                       altimok = 1;
                     }
                    temp = temp tempaltimeter[j];
                   }
                  if (altimok && (temp != altimeter / 100.0))
                   altimok = 0;
                  if (DBG == 21)
                   if (!altimok)
```

```
                    printf("\ntyped in wrong altimeter value = %s
instead of %1.2f",temp, altimeter / 100.0);
                   else
                    printf("\ntyped in correct altimeter value = %s"
,temp);
                 }
                else
                 altimok = 0;
               }
            }

# get the change altitude value typed in
# value should be 6500
    if (index(interrupts["IT-MARK-EVENT"],"K18-CHGALT-AUD")
&&\
      index($2,"<FMS>:PAGE=(LEGS") &&\
      index($2,"-TYPED=("))
     {
      split($2,arr,"(");
      temp = arr[3];
      split(temp,arr," ");
      if (!index(arr[1],"EXC"))
       {
        if (index(arr[1],"CL1"))
         {
          if (k18altcnt >= 0)
           {
            k18altcnt -= 1;
            k18altok = 0;
           }
         }
        else
         {
          k18altcnt += 1;
          tempalt[k18altcnt] = arr[1];
          if (k18altcnt == altdigits)
           {
            temp = tempalt[1];
            for(j=2;j<=altdigits; j++)
             temp = temp tempalt[j];
            if (temp == altval)
             k18altok = 1;
            if (DBG == 21)
             if (!k18altok)
              printf("\ntyped in wrong alt chg value = %s instead of
%d",temp, altval);
             else
              printf("\ntyped in correct alt chg value = %s" ,temp);
           }
          else
           k18altok = 0;
         }
       }
     }


# get the change altitude value typed in
# value should be 6500
    if (index(interrupts["IT-MARK-EVENT"],"FAF-CHGALT-AUD")
&&\
      index($2,"<FMS>:PAGE=(LEGS") &&\
      index($2,"-TYPED=("))
     {

      split($2,arr,"(");
```

```
      temp = arr[3];
      split(temp,arr," ");
      if (!index(arr[1],"EXC"))
       {
        if (index(arr[1],"CL1"))
         {
          if (fafaltcnt >= 0)
           {
            fafaltcnt -= 1;
            fafaltok = 0;
           }
         }
        else {
           fafaltcnt += 1;
           tempalt[fafaltcnt] = arr[1];
           if (fafaltcnt == altdigits)
            {
             temp = tempalt[1];
             for(j=2;j<=altdigits; j++)
              temp = temp tempalt[j];
             if (temp == altval)
              fafaltok = 1;
             if (DBG == 21)
              if (!fafaltok)
               printf("\ntyped in wrong alt chg value = %s instead of
%d",temp, altval);
              else
               printf("\ntyped in correct alt chg value = %s" ,temp);
            }
           else
            fafaltok = 0;
          }
       }
     }

# get the change speed value typed in
# value should be "160/"
    if (index(interrupts["IT-MARK-EVENT"],"K18-CHGSPD-AUD")
&&\
      index($2,"<FMS>:PAGE=(LEGS") &&\
      index($2,"-TYPED=("))

     {
      split($2,arr,"(");
      temp = arr[3];
      split(temp,arr," ");
      if (!index(arr[1],"EXC"))
       {
        if (index(arr[1],"CL1"))
         {
          if (k18spdcnt >= 0)
           {
            k18spdcnt -= 1;
            k18spdok = 0;
           }
         }
        else
         {
          k18spdcnt += 1;
          tempspd[k18spdcnt] = arr[1];
          if (k18spdcnt == spddigits)
           {
            temp = tempspd[1]
             for(j=2;j<=spddigits; j++)
              if (j == spddigits)
               {
                if (tempspd[j] == "/")
                 k18spdok = 1;
               }
              else
               temp = temp  tempspd[j];
            if (k18spdok && (temp != spdval))
             k18spdok = 0;
            if(DBG == 21)
             if (!k18spdok)
              printf("\ntyped in wrong spd chg value = %s instead of
%d",temp, spdval);
             else
              printf("\ntyped in correct spd chg value = %s equals
%d",temp,spdval);
           }
          else
           k18spdok = 0;
         }
       }
     }

# get the change speed value typed in
# value should be "160/"
    if (index(interrupts["IT-MARK-EVENT"],"FAF-CHGSPD-AUD")
&&\
      index($2,"<FMS>:PAGE=(LEGS") &&\
      index($2,"-TYPED=("))
     {
#     (index($2,"<FMS>:PAGE=(LEGS    ;1)-TYPED=("))

      split($2,arr,"(");
      temp = arr[3];
      split(temp,arr," ");
      if (!index(arr[1],"EXC"))
       {
        if (index(arr[1],"CL1"))
         {
          if (fafspdcnt >= 0)
           {
            fafspdcnt -= 1;
            fafspdok = 0;
           }
         }
        else{
         fafspdcnt += 1;
         tempspd[fafspdcnt] = arr[1];
         if (fafspdcnt == spddigits)
          {
           temp = tempspd[1]
            for(j=2;j<=spddigits; j++)
             if (j == spddigits)
              {
               if (tempspd[j] == "/")
                fafspdok = 1;
              }
             else
              temp = temp  tempspd[j];
           if (fafspdok && (temp != spdval))
            fafspdok = 0;
           if (DBG == 21)
            if (!fafspdok)
             printf("\ntyped in wrong spd chg value = %s instead of
%d",temp, spdval);
            else
```

```
                  printf("\ntyped in correct spd chg value = %s equals
%d",temp,spdval);
               }
             else
               fafspdok = 0;
            }
          }
       }

      while ((!found) && (i<=procarrsize))
       {
# have to interject this typing code before the exact match
# code because it should supercede it ie. if pilot is typing
# and there are typing bins to fill, fill them
        if (((leg == 3) &&\
            (index($2,"<FMS>:PAGE=(N1 LIMIT;")) &&\
            (index($2,")-TYPED=(")) &&\
            (index(procstrarr[i],"<FMS>:PAGE=(N1 LIMIT;")) &&\
            (index(procstrarr[i],")-TYPED=(")) &&\
            (proccntarr[i] == -1)) ||\
            ((leg == 2) &&\
            (index($2,"<FMS>:PAGE=(PRF INIT;")) &&\
            (index($2,")-TYPED=(")) &&\
            (index(procstrarr[i],"<FMS>:PAGE=(PRF INIT;")) &&\
            (index(procstrarr[i],")-TYPED=(")) &&\
            (proccntarr[i] == -1)))
          {
# # check to see if thing that was typed is legal
            if (index($2,procstrarr[i]))
             found = 1;
            else {
# # for epr and altimeter values legal things are digits and decimal point
             split($2,arr,"(");
             temp = arr[3];
             split(temp,arr," ");
             temp = arr[1];
             split(procstrarr[i],arr,"(");
             temp1 = arr[3];
             split(temp1,arr," ");
             temp1 = arr[1];
             if (((temp == ".") ||  ((temp >= 0) &&  (temp <= 9))) &&\
                ((temp1 == ".") ||  ((temp1 >= 0) &&  (temp1 <= 9))))
               found = 1;
             if (DBG == 21)
               if (found)
                 printf("\n###PROCfilling up %s with a %s instead of
%s",procstrarr[i],temp,temp1);
            }
          }
        if (!found)
          if (index($2,procstrarr[i]))
          {
            found = 1;
            if(proccntarr[i] != -1)
             duplicate = 1;
          }
        else
          if ((leg == 1) &&\
             (index(procstrarr[i],"<FMS>:FROM=(STATUS")) &&\
             (index($2,"<FMS>:FROM=(STATUS")))
            {
             found = 1;
             if(proccntarr[i] != -1)
               duplicate = 1;
            }
```

```
          else
# dont care about page from when going to any
# INIT REF page in second procedural leg
           if ((leg == 2) &&\
              (index($2,"<FMS>:FROM=")) &&\
              (index(procstrarr[i],"<FMS>:FROM=")) &&\
              (index($2,"-TO=(INIT REF")) &&\
              (index(procstrarr[i],"-TO=(INIT REF")))
             {
              found = 1;
              if(proccntarr[i] != -1)
                duplicate = 1;
             }
           else
# dont care about page numbers in transition from
# PRF INIT to LEGS
            if ((leg == 2) &&\
               (index($2,"<FMS>:FROM=(PRF INIT")) &&\
               (index(procstrarr[i],"<FMS>:FROM=(PRF INIT")) &&\
               (index($2,"-TO=(LEGS")) &&\
               (index(procstrarr[i],"-TO=(LEGS")))
              {
               found = 1;
               if(proccntarr[i] != -1)
                 duplicate = 1;
              }
            else
# dont care from which INIT REF page PERF BARSET is selected
             if ((leg == 2) &&\
                (index($2,"<FMS>:PAGE=(INIT REF")) &&\
                (index(procstrarr[i],"<FMS>:PAGE=(INIT REF")) &&\
                (index($2,"<PERF/BARSET")) &&\
                (index(procstrarr[i],"<PERF/BARSET")))
               {
                found = 1;
                if(proccntarr[i] != -1)
                 duplicate = 1;
               }
             else
# dont care about page from unless page to is legs
              if ((index(procstrarr[i],"<FMS>:FROM=")) &&
(index($2,"<FMS>:FROM=")) && (!index($2,"TO=(LEGS")))
                {
                 n = index(procstrarr[i],"TO=");
                 temp = substr(procstrarr[i],n);
                 if (index($2,temp))
                   {
                    found = 1;
#           printf("\nin proc code trying to coalesce %s and
%s",$2,procstrarr[i]);
                    if (proccntarr[i] != -1)
                      duplicate = 1;
                   }
                }
                else
# check for speedbrake value > 0
                 if ((leg == 3) && (index(procstrarr[i],"SPEEDBRAKES"))
&& (index($0,"SPEEDBRAKES")))
                   {
                    split($2,arr,"=");
                    if (arr[2] > 0)
                      {
                       found = 1;
                       if(proccntarr[i] != -1)
                        duplicate = 1;
```

```
                    }
                  }
                 else
# not going to worry if the bar set value is that of the altimeter
                  if ((leg == 2) && (index($2,"bar set")) &&
(index(procstrarr[i],"bar set")))
                   {
                    found = 1;
                    if(proccntarr[i] != -1)
                      duplicate = 1;
                   }
                  else
# not worrying about matching the value of the go around epr
                   if ((leg == 3) &&\
                      (index($2,"<FMS>:PAGE=(N1 LIMIT;1)-
LINESELECT=(           ; GA")) &&\
                      (index(procstrarr[i],"<FMS>:PAGE=(N1 LIMIT;1)-
LINESELECT=(           ; GA")))
                    {
                      found = 1;
                      if(proccntarr[i] != -1)
                       duplicate = 1;
                    }
# k18 and faf procedures may end with a time out
                   else
                    if ((((leg == 3) &&\

(index(procstrarr[i],"<CHECKLIST>:FROM=(Descent-Cklst)-
TO=(MM)")) &&\
                         (index($2,"<CHECKLIST>:TIME-OUT-
FROM=(Descent-Cklst)-TO=(MM)")))) ||\
                       (((leg == 2) &&\

(index(procstrarr[i],"<CHECKLIST>:FROM=(Approach-Cklst)-
TO=(MM)")) &&\
                         (index($2,"<CHECKLIST>:TIME-OUT-
FROM=(Approach-Cklst)-TO=(MM)")))))
                      {
                       found = 1;
                       if(proccntarr[i] != -1)
                         duplicate = 1;
                      }

        if (found || duplicate)
          {
# this event even though perhaps extraneous may be the first procedural
event
# after an interruption or may extend the end of the procedure
            if (i == procarrsize)
              {
               eendtime = time;
               if ((DBG == 17) || (DBG == 22))
                 printf("\n#####loading last event of procedure %s at time =
%1.2f leg = %d####",\
                        procstrarr[i],eendtime,leg);
              }

            isprocevent = 1;

# code added to take resumptive proc event from EXC
            if ((excresumetime == -1) && (intexc > 0) && (interrupts["IT-
MARK-TIME"] != 0))
              {
               excresumetime = time - intexc;
```

```
#             printf("\nexresumetime = %1.3f event = %s",
excresumetime,$2)
               if (excresumeclass == -1)
                 excresumeclass = 1;
              }

            if(procresumetime == -1)
              if ((interrupts["IT-MARK-TIME"] != 0) &&
(interrupts[interrupts["IT-MARK-EVENT"] "=END"] != 0))
               {
                 if (resumevclass == -1)
                   {
                    resumevclass = 1;
                    if (DBG == 13)
                     printf("\nclassifying %s as %d\ncurrent time = %1.3f > int
ending %1.3f",\
                            $2,\
                            resumevclass,\
                            time,\
                            interrupts[interrupts["IT-MARK-EVENT"] "=END"]);
                   }
                 procresumetime = time - interrupts[interrupts["IT-MARK-
EVENT"] "=END"];
                 if (DBG == 12)
                   printf("\n in proc code in leg %d:\n resuming from
interruption %s\n with %s \n resumetime = %1.3f\n time = = %1.3f > int
ending = %1.3f\n",\
                          leg,interrupts["IT-MARK-EVENT"],procstrarr[i],\
                          procresumetime,time,interrupts[interrupts["IT-MARK-
EVENT"] "=END"]);
               }
#     printf("\nfound an event %s count is %d \nprocresumetime = %1.3f
\nint start = %1.3f int stop = %1.3f \n time = %1.3f\n",\
#            procstrarr[i],proccntarr[i],\
#            procresumetime,\
#            interrupts["IT-MARK-TIME"],\
#            interrupts[interrupts["IT-MARK-EVENT"] "=END"],time);

          } # if duplicate or found
# if the slot already has an entry, but the new entry is in
# order, put the new value in the slot  and mark the old value
# as extraneous
          if(duplicate)
            {
             if ((proccount == proccntarr[i - 1]))
               {
                if (DBG == 22)
                  {
                   printf("\nPROC:here we should insert %s in position
%d",procstrarr[i],proccount + 1);
                   printf("\nits current value is %d", proccntarr[i]);
                  }
                found = 1;
# if this is a duplicate TUNING pass it through
                if ((!(index($2,"TUNING"))) && (!(index($2,"MATCHED-
FREQ"))))
                  {
                   extranproccount += 1;
                   exprocarr[extranproccount] = $2;
                  }
               }
            else{
             found = 0;
             duplicate = 0;
            }
```

```
          }
        if(!found)
          i += 1;
       }# while not found

        if(found)
          {
            proccount += 1;
            proccntarr[i] = proccount;
# if this is the first procedural event then mark the time
            if (i == 1)
              {
               estarttime = time;
               if ((DBG == 17) || (DBG == 22))
                 printf("\n#####loading first event of procedure %s at time =
%1.3f leg = %d####",\
                        procstrarr[i],estarttime,leg);
              }
          }
        else{
          if (DBG == 22)
            printf("\nin leg %d %s\nis not a procedural event checking
interrupts",leg,$2);
# check the appropriate interruption events
          if ((intsize > 0))
            {
# intsize is non zero when marker has been crossed
              i = 1;
              duplicate = 0;
              rogernotrequired = 0;
              while ((!found) && (i<=intsize))
               {
# code added to count events between hitting EXC button
# and the end of the event
                 if (index($2,"-TYPED=(EXC)") &&\
                     index(intstrarr[i],"-TYPED=(EXC)"))
                   {
                     found = 1;
                     if (intcntarr[i] != -1)
                      duplicate = 1;
                     if (intexc == -1)
                      {
                        intexc = time;
                        exctoprocfpmcount = 0;
                        if (interrupts["IT-MARK-TIME"] == 0)
                          itmarktoexc = errornumber;
                        else
                          itmarktoexc = time - interrupts["IT-MARK-TIME"];
                      }

                     if (startexctoendcount == -1)
                      {
                        startexctoendcount = 1;
                        exctoendcount = 0;

                      }
                   }
# code changed to relax the page restrictions on
# the legs page both for speed and altitude changes
                 if (index(interrupts["IT-MARK-EVENT"],"CHGSPD-AUD")
||\
                     index(interrupts["IT-MARK-EVENT"],"CHGALT-
AUD"))
                   {
                     if (index($2,"<FMS>:FROM=(LEGS  ") &&\
                         index($2,"-TO=(LEGS") &&\
                         index(intstrarr[i],"<FMS>:FROM=(LEGS  ") &&\
                         index(intstrarr[i],"-TO=(LEGS"))
                       {
                         found = 1;
                         if (intcntarr[i] != -1)
                           duplicate = 1;
                       }
                     else
# on the legs page
                     if (index($2,"<FMS>:PAGE=(LEGS") &&\
                         index(intstrarr[i],"<FMS>:PAGE=(LEGS"))
# first the lineselect and typing EXC
                      if((index($2,"-TYPED=(EXC)") &&\
                          index(intstrarr[i],"-TYPED=(EXC)")) ||\
                         (index($2,"-LINESELECT=(m") &&\
                         index(intstrarr[i],"-LINESELECT=(m")))
                        {
                          found = 1;
                          if (intcntarr[i] != -1)
                            duplicate = 1;
                        }
                      else
# typing a speed change
                        if (index(interrupts["IT-MARK-EVENT"],"CHGSPD-
AUD") &&\
                            index($2,"-TYPED=(") &&\
                            index(intstrarr[i],"-TYPED=(") &&\
                            (intcntarr[i] == -1))
                          {
                            if (index($2,intstrarr[i]))
                             found = 1;
# legal chg spd values are digits and /
                            else
                             {
                               split($2,arr,"(");
                               temp = arr[3];
                               split(temp,arr," ");
                               temp = arr[1];
                               split(intstrarr[i],arr,"(");
                               temp1 = arr[3];
                               split(temp1,arr," ");
                               temp1 = arr[1];
                               if (((temp == "/") ||  ((temp >= 0) &&  (temp <= 9)))
&&\
                                   ((temp1 == "/") ||  ((temp1 >= 0) &&  (temp1 <=
9))))
                                 found = 1;
                            if (DBG == 21)
                             if (found)
                               {
                                 printf("\n###INT:filling up %s with a %s instead of
%s",intstrarr[i],temp,temp1);
                                 printf("\n%s\n%s",$2,intstrarr[i]);
                               }
                             }
                          }
                        else
# typing an altitude change
                          if (index(interrupts["IT-MARK-EVENT"],"CHGALT-
AUD") &&\
                             index($2,"-TYPED=(") &&\
                             index(intstrarr[i],"-TYPED=(") &&\
                             (intcntarr[i] == -1))
                            {
```

```
                        if (index($2,intstrarr[i]))
                         found = 1;
                        else
                         {
# legal chg alt values are digits
                          split($2,arr,"(");
                          temp = arr[3];
                          split(temp,arr," ");
                          temp = arr[1];
                          split(intstrarr[i],arr,"(");
                          temp1 = arr[3];
                          split(temp1,arr," ");
                          temp1 = arr[1];
                          if (((temp >= 0) &&  (temp <= 9)) &&\
                             ((temp1 >= 0) &&  (temp1 <= 9)))
                             found = 1;
                          if (DBG == 21)
                            if (found)
                             printf("\n###INT:filling up %s with a %s instead
of %s",intstrarr[i],temp,temp1);
                         }
                       }
               } # changing speed or altitude
            if(!found)
# in TOD-INITRUN-VIS interrupt - ignore content of message
              if ((interrupts["IT-MARK-EVENT"] == "TOD-INITRUN-
VIS") &&\
                 (index($2,"<DATALINK>:FROM=(MESSAGE)-
TO=(MM-")) &&\
                 (index(intstrarr[i],"{UNABLE or ROGER}")))
               {
                 found = 1;
                 n = index($2,"MM-");
                 temp = substr($2,n + 3);
                 split(temp,arr,")");
                 if(index(arr[1],"ROGER"))
                  rogernotrequired = 1;
                 else
                  rogernotrequired = 0;
                 if(intcntarr[i] != -1)
                  duplicate = 1;
               }

               else
# dont care about page from unless page to is legs
                 if ((index(intstrarr[i],"<FMS>:FROM=")) &&
(index($2,"<FMS>:FROM=")) && (!index($2,"TO=(LEGS")))
                  {
                   n = index(intstrarr[i],"TO=(");
                   temp = substr(intstrarr[i],n);
                   if (index($2,temp))
                     {
                      found = 1;
#             printf("\nin int code trying to coalesce %s and
%s",$2,intstrarr[i]);
                      if (intcntarr[i] != -1)
                       duplicate = 1;
                     }
                  }
              else
# ignore target of TALK AND TALK-STOPPED in all interruptions
               if ((((index(intstrarr[i],"TALK-TO")) &&
(index($2,"TALK-TO"))) ||\
            ((index(intstrarr[i],"TALK-STOPPED-TO")) &&
(index($2,"TALK-STOPPED-TO")))))
```

```
                 {
                  found = 1;
                  if (intcntarr[i] != -1)
                   duplicate = 1;
                 }
               else
# dont care about latitude of new waypoint in lineselect
                 if(((index(interrupts["IT-MARK-EVENT"],"FAF-
HOLDPAT-AUD")) &&\
                   (index(intstrarr[i],"<FMS>:PAGE=(HOLD    ;1)-
LINESELECT=( 328$  15.3n;MAFAT       ;1L)")) &&\
                   (index($2,"<FMS>:PAGE=(HOLD    ;1)-
LINESELECT=("))) ||\
                  ((index(interrupts["IT-MARK-EVENT"],"K18-
HOLDPAT-AUD")) &&\
                   (index(intstrarr[i],"<FMS>:PAGE=(HOLD    ;2)-
LINESELECT=( 328$  15.3n;MAFAT       ;1L)")) &&\
                   (index($2,"<FMS>:PAGE=(HOLD    ;2)-
LINESELECT=("))))
                  {
                   if (index($2,"MAFA"))
                     {
                      found = 1;
                      itselerr = 0;
                      if (intcntarr[i] != -1)
                       duplicate = 1;
                     }
                   else
                     if (itselerr == -1)
                      itselerr = 1;
                  }
                 else
# look only at initial runway value in lineselect on arrival page
                  if ((index(interrupts["IT-MARK-EVENT"],"TOD-
INITRUN")) &&\
                     (index(intstrarr[i],\
                          sprintf("<FMS>:PAGE=(ARR     ;1)-
LINESELECT=( runways    ;%s         ;1L)",runway))) &&\
                     (index($2,"<FMS>:PAGE=(ARR     ;1)-
LINESELECT")))
                   {
                     split($2,arr,"=");
                     temp = arr[3];
                     split(temp,arr,";");
                     temp = arr[2];
                     split(temp,arr," ");
                     temp = arr[1];
                     if (runway == temp)
                       {
                        found = 1;
                        itselerr = 0;
                        if (intcntarr[i] != -1)
                          duplicate = 1;
                       }
                     else
                       itselerr = 1;
                   }
                  else
# look only at new runway value in lineselect on arrival page
                   if ((index(interrupts["IT-MARK-EVENT"],"NEWRUN-
AUD")) &&\
                      (index(intstrarr[i],\
                           sprintf("<FMS>:PAGE=(ARR     ;1)-
LINESELECT=(            ;%s         ;2L)",newrunway))) &&\
```

```
                          (index($2,"<FMS>:PAGE=(ARR     ;1)-
LINESELECT")))
                         {
                          split($2,arr,"=");
                          temp = arr[3];
                          split(temp,arr,";");
                          temp = arr[2];
                          split(temp,arr," ");
                          temp = arr[1];
                          if (newrunway == temp)
                           {
                             found = 1;
                             itselerr = 0;
                             if (intcntarr[i] != -1)
                               duplicate = 1;
                           }
                          else
                           itselerr = 1;
                         }
                       else
                         if (index($2,intstrarr[i]))
                          {
                           found = 1;
                           if (intcntarr[i] != -1)
                             duplicate = 1;
                          }

              if(duplicate)
               if ((intcount == intcntarr[i - 1]))
                 {
                  if (DBG == 22)
                    {
                     printf("\nINT:here we should insert %s in position
%d",intstrarr[i],intcount + 1);
                     printf("\nits current value is %d", intcntarr[i]);
                    }
                  found = 1;
                  intcntarr[i] = -1;
# flag the condition where the end should be adjusted
                  if (i == intsize)
                    {
#                    printf("\nrefilling ending event %s at time =
%1.3f",$2,time);
                      inttwolegs = 1;
                    }

                  extranproccount += 1;
                  exprocarr[extranproccount] = $2;
                 }
              else
               {
                 found = 0;
                 duplicate = 0;
               }
            if (!found)
              i += 1;
          } # while loop
        }
        else
          if (DBG == 22)
            printf("\ninterrupt has not occurred yet");
        if(found)
          {
            intcount += 1;
            if (intcntarr[i] == -1)
```

```
               intcntarr[i] = intcount;
             else
               intcntarr[i] = sprintf("%s %d",intcntarr[i],intcount);
             if (DBG == 22)
               printf("\nin interrupt code event loading int[%s] = %d with :
\n%s at time = %1.3f",\
                     intstrarr[i],intcntarr[i],$2,time);
           }
         else{
           if (DBG == 22)
             printf("\n doing dont care filter on %s ",$2);

# filter out dont care events
# leg 3 any extra speedbrake value
# all talk stopped to events
# leg 2 any flap adjustments after the level off has started
# any leg listen to events
# ignore TUNING to some frequency other than INVALID
           extraneous = 0;
           if((!((leg == 3) && index($2,"SPEEDBRAKES"))) &&\
              (!index($2,"TALK-STOPPED-TO")) && \
              (!((leg == 2) && (index($2,"FLAPS") && (ensemblettd != 0))))
&&\
              (!index($2,"<RADIO>:LISTEN-TO=")) &&\
              (!(index($2,"TUNING"))) &&\
              (!(index($2,"MATCHED-FREQ"))) &&\
              (!(index($2,"<DATALINK>:FROM=(MM)-
TO=(MESSAGE)"))))
              {
               extraneous = 1;
# the next two filters are for the typing in of altimeter
# go around epr, changes of altitude and speed numbers
               if (((leg == 3) &&\
                   (index($2,"<FMS>:PAGE=(N1 LIMIT;1)-TYPED=("))) ||\
                    ((leg == 2) &&\
                     (index($2,"<FMS>:PAGE=(PRF INIT;1)-TYPED=("))))
                 {
                  split($2,arr,"(");
                  temp = arr[3];
                  split(temp,arr," ");
# if this is a legal digit and the number is not complete
                  if ((arr[1] == ".") || ((arr[1] >= 0) && (arr[1] <= 9)))
                    {
                     if (((leg == 3) && (eprcnt <= eprdigits)) ||\
                         ((leg == 2) && (altimcnt <= altimdigits)))
                                                     extraneous = 0;
                 }
# if this is a clear then make the first one extraneous
              else
               if (index(arr[1],"CL"))
                 if (leg == 2)
                   if (altimclrcnt == 0)
                     altimclrcnt += 1;
                   else
                     extraneous = 0;
                 else
                   if (eprclrcnt == 0)
                     eprclrcnt += 1;
                   else
                     extraneous = 0;
             }
            if (((index(interrupts["IT-MARK-EVENT"],"FAF-CHGSPD-
AUD")) &&\
                (index($2,"<FMS>:PAGE=(LEGS    ;1)-TYPED=("))) ||\
```

```
          ((index(interrupts["IT-MARK-EVENT"],"K18-CHGSPD-
AUD")) &&\
           (index($2,"<FMS>:PAGE=(LEGS    ;2)-TYPED=("))))
         {
          split($2,arr,"(");
          temp = arr[3];
          split(temp,arr," ");
# if this is a legal digit and the number is not complete
          if ((arr[1] == "/") || ((arr[1] >= 0) && (arr[1] <= 9)))
            {
              if(((leg == 2) && (k18spdcnt <= spddigits)) ||\
                 ((leg == 3) && (fafspdcnt <= spddigits)))
                                              extraneous = 0;
         }
         else
# if this is a clear make only the first one extraneous
          if (index(arr[1],"CL"))
            if (leg == 2)
              if (k18spdclrcnt == 0)
                k18spdclrcnt += 1;
              else
                extraneous = 0;
            else
              if (fafspdclrcnt == 0)
                fafspdclrcnt += 1;
              else
                extraneous = 0;
        }

        if (((index(interrupts["IT-MARK-EVENT"],"FAF-CHGALT-
AUD")) &&\
           (index($2,"<FMS>:PAGE=(LEGS    ;1)-TYPED=("))) ||\
          ((index(interrupts["IT-MARK-EVENT"],"K18-CHGALT-
AUD")) &&\
           (index($2,"<FMS>:PAGE=(LEGS    ;2)-TYPED=("))))
         {
          split($2,arr,"(");
          temp = arr[3];
          split(temp,arr," ");
# if this is a legal digit and the number is not complete
          if ((arr[1] == "/") || ((arr[1] >= 0) && (arr[1] <= 9)))
            {
              if(((leg == 2) && (k18altcnt <= altdigits)) ||\
                 ((leg == 3) && (fafaltcnt <= altdigits)))
                extraneous = 0;
            }
         else
# if this is a clear make only the first one extraneous
          if (index(arr[1],"CL"))
            if (leg == 2)
              if (k18altclrcnt == 0)
                k18altclrcnt += 1;
              else
                extraneous = 0;
            else
              if (fafaltclrcnt == 0)
                fafaltclrcnt += 1;
              else
                extraneous = 0;
        }
        if (extraneous == 1)
         {
          extranproccount += 1;
          exprocarr[extranproccount] = $2;
         }
```

```
        }# potentially extraneous
         if (DBG == 22)
           if(extraneous == 0)
             printf("\ndont care about %s",$2);
           else
             printf("\n%s is extraneous",$2);
      } # subject to filter
      } # not a procedural event
             } # potential procedural event

# collect interrupt event times for (2.4, 2.5 and 2.6 of specs)
    if (index($0,"<IT-MARKER>") && !(index($0,"WINDOW-
TYPE")))
      {
# IT-MARKER time
      split($0,arr,":");
      temp = arr[2];
      split(temp,arr," ");
      interrupts["IT-MARK-EVENT"] = arr[1];
      interrupts["IT-MARK-TIME"] = time;
#      printf("\n%s occured at time %1.3f",interrupts["IT-MARK-
EVENT"],interrupts["IT-MARK-TIME"]);
# load appropriate interrupt array into intcntarr
      if (arr[1] == "TOD-INITRUN-AUD")
        {
         for(i=1;i<=itirasize;i++)
           {
            intstrarr[i] = ira[i];
            if(index(intstrarr[i],"IT-MARKER"))
              {
               intcount += 1;
               intcntarr[i] = intcount;
              }
            else
              intcntarr[i] = -1;
           }
         intsize = itirasize;
#     for(i=1;i<=intsize;i++)
#       printf("\nintcntarr[%s] = %d",\
#            intstrarr[i],intcntarr[i]);
        }
      else
        if (arr[1] == "TOD-INITRUN-VIS")
          {
           for(i=1;i<=itirvsize;i++)
             {
              intstrarr[i] = irv[i];
              if(index(intstrarr[i],"IT-MARKER"))
                {
                 intcount += 1;
                 intcntarr[i] = intcount;
                }
              else
                intcntarr[i] = -1;
             }
           intsize = itirvsize;
#       for(i=1;i<=intsize;i++)
#         printf("\nintcntarr[%s] = %d",\
#              intstrarr[i],intcntarr[i]);
          }
        else
          if (arr[1] == "K18-HOLDPAT-AUD")
            {
             for(i=1;i<=ithpasize;i++)
               {
```

```
                     intstrarr[i] = hpk[i];
                     if(index(intstrarr[i],"IT-MARKER"))
                       {

                         intcount += 1;
                         intcntarr[i] = intcount;
                       }
                     else
                       intcntarr[i] = -1;
                   }
                 intsize = ithpasize;
#           for(i=1;i<=intsize;i++)
#             printf("\nintcntarr[%s] = %d",\
#                    intstrarr[i],intcntarr[i]);
                }
             else
               if (arr[1] == "FAF-HOLDPAT-AUD")
                 {
                   for(i=1;i<=ithpasize;i++)
                     {
                       intstrarr[i] = hpf[i];
                       if(index(intstrarr[i],"IT-MARKER"))
                         {
                           intcount += 1;
                           intcntarr[i] = intcount;
                         }
                       else
                         intcntarr[i] = -1;
                     }
                   intsize = ithpasize;
#             for(i=1;i<=intsize;i++)
#               printf("\nintcntarr[%s] = %d",\
#                      intstrarr[i],intcntarr[i]);
                 }
               else
                 if (arr[1] == "FAF-CHGSPD-AUD")
                   {
                     for(i=1;i<=itcsasize;i++)
                       {
                         intstrarr[i] = csf[i];
                         if(index(intstrarr[i],"IT-MARKER"))
                           {
                             intcount += 1;
                             intcntarr[i] = intcount;
                           }
                         else
                           intcntarr[i] = -1;
                       }
                     intsize = itcsasize;
#               for(i=1;i<=intsize;i++)
#                 printf("\nintcntarr[%s] = %d",\
#                        intstrarr[i],intcntarr[i]);
                   }
                 else
                   if (arr[1] == "K18-CHGSPD-AUD")
                     {
                       for(i=1;i<=itcsasize;i++)
                         {
                           intstrarr[i] = csk[i];
                           if(index(intstrarr[i],"IT-MARKER"))
                             {
                               intcount += 1;
                               intcntarr[i] = intcount;
                             }
                           else
                            intcntarr[i] = -1;
                         }
                       intsize = itcsasize;
#                    for(i=1;i<=intsize;i++)
#                      printf("\nintcntarr[%s] = %d",\
#                             intstrarr[i],intcntarr[i]);
                     }
                   else
                     if (arr[1] == "FAF-CHGALT-AUD")
                       {
                        for(i=1;i<=itcaasize;i++)
                          {
                            intstrarr[i] = caf[i];
                            if(index(intstrarr[i],"IT-MARKER"))
                              {
                               intcount += 1;
                               intcntarr[i] = intcount;
                              }
                            else
                              intcntarr[i] = -1;
                          }
                        intsize = itcaasize;
#                  for(i=1;i<=intsize;i++)
#                    printf("\nintcntarr[%s] = %d",\
#                           intstrarr[i],intcntarr[caf[i]]);
                       }
                     else
                       if (arr[1] == "K18-CHGALT-AUD")
                         {
                           for(i=1;i<=itcaasize;i++)
                             {
                               intstrarr[i] = cak[i];
                               if(index(intstrarr[i],"IT-MARKER"))
                                 {
                                   intcount += 1;
                                   intcntarr[i] = intcount;
                                 }
                               else
                                 intcntarr[i] = -1;
                             }
                           intsize = itcaasize;
#                    for(i=1;i<=intsize;i++)
#                      printf("\nintcntarr[%s] = %d",\
#                             intstrarr[i],intcntarr[i]);
                         }
                       else
                         if (arr[1] == "FAF-NEWRUN-AUD")
                           {
                             for(i=1;i<=itnrasize;i++)
                               {
                                intstrarr[i] = nrf[i];
                                if(index(intstrarr[i],"IT-MARKER"))
                                  {
                                    intcount += 1;
                                    intcntarr[i] = intcount;
                                  }
                                else
                                  intcntarr[i] = -1;
                               }
                             intsize = itnrasize;
#                      for(i=1;i<=intsize;i++)
#                        printf("\nintcntarr[%s] = %d",\
#                               intstrarr[i],intcntarr[i]);
                           }
                         else
```

```
                     if (arr[1] == "K18-NEWRUN-AUD")
                      {
                        for(i=1;i<=itnrasize;i++)
                         {
                          intstrarr[i] = nrk[i];
                          if(index(intstrarr[i],"IT-MARKER"))
                           {
                            intcount += 1;
                            intcntarr[i] = intcount;
                           }
                          else
                           intcntarr[i] = -1;
                         }
                        intsize = itnrasize;
#                      for(i=1;i<=intsize;i++)
#                        printf("\nintcntarr[%s] = %d",\
#                              intstrarr[i],intcntarr[i]);
                      }
      }
   else{
        if (intsize != 0)
         {
# load ACKN time
           if (interrupts[interrupts["IT-MARK-EVENT"] "=ACKN"] == 0)
            {
             if (index(interrupts["IT-MARK-EVENT"],"TOD-INITRUN-
VIS"))
               {
                if (index($2,"<DATALINK>:FROM=(MESSAGE)-
TO=(MM"))
                 interrupts[interrupts["IT-MARK-EVENT"] "=ACKN"] =
time;
               }
             else
              if (index($2,"<RADIO>:TALK-TO"))
                interrupts[interrupts["IT-MARK-EVENT"] "=ACKN"] =
time;
            }
# load START time
           if (interrupts[interrupts["IT-MARK-EVENT"] "=START"] ==
0)
            {
             if ((index(interrupts["IT-MARK-EVENT"],"TOD-
INITRUN")) ||\
                (index(interrupts["IT-MARK-EVENT"],"NEWRUN")))
               {
                if (index($2,"<FMS>:FROM=(") &&\
                   index($2,"-TO=(DEP ARR ;1)"))
                 interrupts[interrupts["IT-MARK-EVENT"] "=START"] =
time;
               }
             else
              if (index(interrupts["IT-MARK-EVENT"],"HOLDPAT-
AUD"))
                {
                 if (index($2,"<FMS>:FROM=(") &&\
                    index($2,"-TO=(HOLD"))
                   interrupts[interrupts["IT-MARK-EVENT"] "=START"] =
time;
                }
              else
               if (index(interrupts["IT-MARK-EVENT"],"CHGALT-
AUD"))
                 {
                  if (index($2,"<FMS>:PAGE=(LEGS") &&\
                     index($2,"-TYPED=(6"))
                    interrupts[interrupts["IT-MARK-EVENT"] "=START"]
= time;
                 }
                else
                 if (index(interrupts["IT-MARK-EVENT"],"CHGSPD-
AUD"))
                   {
                    if (index($2,"<FMS>:PAGE=(LEGS") &&\
                       index($2,"-TYPED=(1"))
                     interrupts[interrupts["IT-MARK-EVENT"] "=START"]
= time;
                   }
            }
# load end time
         if (interrupts[interrupts["IT-MARK-EVENT"] "=END"] == 0)
          {
           if ((index(interrupts["IT-MARK-EVENT"],"TOD-INITRUN"))
||\
              (index(interrupts["IT-MARK-EVENT"],"NEWRUN")))
             {
              if ((index($2,"<FMS>:FROM=(ARR")) &&\
                 (index($2,"-TO=(LEGS")))
               interrupts[interrupts["IT-MARK-EVENT"] "=END"] = time;
       }
         else
          if (index(interrupts["IT-MARK-EVENT"],"HOLDPAT-AUD"))
            {
            if (index($2,"<FMS>:FROM=(HOLD") &&\
              index($2,"-TO=(LEGS"))
             {
              split($2,arr,"(");
              if (!index(arr[3],"NXT PAGE"))
                interrupts[interrupts["IT-MARK-EVENT"] "=END"] = time;
             }
           }
          else
           if ((index(interrupts["IT-MARK-EVENT"],"CHGALT-AUD"))
||\
              (index(interrupts["IT-MARK-EVENT"],"CHGSPD-AUD")))
             if (index($2,"<FMS>:FROM=(LEGS  ") &&\
                 index($2,"-TO=(LEGS"))
                 interrupts[interrupts["IT-MARK-EVENT"] "=END"] =
time;
          if (interrupts[interrupts["IT-MARK-EVENT"] "=END"] != 0)
           {
            startexctoendcount = 0;
            if (exctoendcount > 0)
              exctoendcount--;
           }
          }
          }

       }
# mark the time to deadline times for all legs 1 2 and 3

              if (index($0,"<FLIGHT-PATH>:SPEED-CHANGE-
INDICATED-TO=240.000"))

                {
                 ensemblettd = time;
                 if (DBG == 16)
                   printf("\nin leg %d marking ensemble tod %s\n at time
=  %1.2f",leg,substr($2,0,50),ensemblettd);
                }
```

```
          if (index($0,"<FLIGHT-PATH>:START-LEVEL-OFF-
TO=12000"))
           {
             ensemblettd = time;
             if (DBG == 16)
               printf("\nin leg %d marking ensemble tod %s\n at time
=  %1.2f",leg,substr($2,0,50),ensemblettd);
           }

          if (index($0,"<FLIGHT-PATH>:START-LEVEL-OFF-
TO=4000"))
           {
             ensemblettd = time;
             if (DBG == 16)
               printf("\nin leg %d marking ensemble tod %s\nat time
= %1.2f",leg,substr($2,0,50),ensemblettd);
           }
# count the active flight path events of the ensemble event
          if((estarttime != -1) ||\
     (interrupts[interrupts["IT-MARK-EVENT"] "=ACKN"] != 0) ||\
     (interrupts[interrupts["IT-MARK-EVENT"] "=START"] != 0))
# if in ensemble either interruption or procedure are not done bump
count1
    if ((eendtime == -1) || (interrupts[interrupts["IT-MARK-EVENT"]
"=END"] == 0))
      {
# see if event qualifies
        if ((countevent != -1) && (eendtime != -1) &&
(interrupts[interrupts["IT-MARK-EVENT"] "=END"] == 0))
         {
# if procedural event is done but there is no
# interruption dont bother to inspect or count the event
          split($6,arr,".");
          temp = arr[2];
          split(temp,arr,">");
          temp = arr[1];

          if ((temp >= 11) && (temp <= 12))
           {
             countevent = -1;
             if (DBG == 17)
               printf("\nchecking condition = %s (no interrupt) proc ended
at %1.3f count = %d current time = %1.3f",\
                    temp,eendtime,count1,time);
           }
          else
           countevent = 0;
         }

        if (countevent == 0)
         {
# either procedure or interruption is incomplete and there is an
interruption
          if ((index($0,"<ENERGY-CTRL>:THROTTLE")) ||\
             (index($0,"<ENERGY-CTRL>:STICK-PITCH-UP")) ||\
             (index($0,"<ENERGY-CTRL>:STICK-PITCH-DOWN")) ||\
             (index($0,"<ENERGY-CTRL>:STICK-ROLL-LEFT")) ||\
             (index($0,"<ENERGY-CTRL>:STICK-ROLL-RIGHT")) ||\
             (index($0,"<ENERGY-CTRL>:FLAPS")) ||\
             (index($0,"<ENERGY-CTRL>:SPEEDBRAKES")))
         if (index($0,"SPEEDBRAKES"))
          {
           split($2,arr,"=");
           if (arr[2] > 4)
             countevent = 1;
            else
             countevent = 0;
          }
         else
          countevent = 1;
          if (countevent == 1)
           {
             if ((eendtime == -1) && (interrupts[interrupts["IT-MARK-
EVENT"] "=END"] == 0))
               {
                count1 += 1;
                if (DBG == 17)
                 printf("\nbumping count1 (neither ended) = %d with %36s
at time %1.3f",count1,substr($2,0,35),time);
               }
             else
               {
                count2 += 1;
# EXC has been typed but first resumptive event has not occured
                if ((excresumetime == -1) && (intexc > 0) &&
(interrupts["IT-MARK-TIME"] != 0))
                 exctoprocfpmcount++;
# interrupt has ended but proc has not
                if (interrupts[interrupts["IT-MARK-EVENT"] "=END"] !=
0)
# if this is before the proc has resumed count it in inttoproccount */
                 if (procresumetime == -1)
                  {
                   inttoprocfpmcount += 1;
                   if (DBG == 17)
                     printf("\n bumping intoproc (int ended) = %d with %36s
at time %1.3f",\
                          inttoprocfpmcount,substr($2,0,35),time);
                  }
                if (DBG == 17)
                 if (eendtime != -1)
                   printf("\nbumping count2 (proc ended) = %d with %36s at
time %1.3f",count2,substr($2,0,35),time);
                 else
                   printf("\nbumping count2 (int ended) = %d with %36s at
time %1.3f",count2,substr($2,0,35),time);
               }
             countevent = 0;
            }
          }
      }
# Code added to to resumptive event classification
# for eventf from Hitting EXC to first proc event
      if ((intexc > 0) && (excresumeclass == -1) &&\
         !(index($0,interrupts["IT-MARK-EVENT"])) &&\
         (time > intexc))
        {
         if (index($0,"SPEEDBRAKES"))
           {
# if this is a speedbrake value it must be > 4
            split($2,arr,"=");
            if (arr[2] > 4)
              excresumeclass = 2;
#             printf("\nin leg %d resuming with %s of class =
%d",leg,$2,excresumeclass);
           }
         else
           {
# it this is in the table of resumptive events it is
```

```
# of class 2 or 3
              i = 1;
              while ((excresumeclass == -1) && (i <= resumarrsize))
                if (index($2,resumarr[i]))
                  {
                    if(index($0,"<ENERGY-CTRL>"))
                      excresumeclass = 2;
                    else
                      excresumeclass = 3;
#              printf("\nin leg %d resuming with %s of class =
%d",leg,$2,excresumeclass);
                  }
                else
                  i += 1;
             }
           if (excresumeclass == -1)
             excresumeclass = 0;
           if (DBG == 13)
             printf("\nclassifying %s as %d\ncurrent time = %1.3f > int
ending %1.3f",\
                  $2,\
                   excresumeclass,\
                  time,\
                  interrupts[interrupts["IT-MARK-EVENT"] "=END"]);
         }


       if ((interrupts["IT-MARK-TIME"] != 0) &&
(interrupts[interrupts["IT-MARK-EVENT"] "=END"] != 0))
         {
# classify the first resumptive event (2.10 in specs)
           if ((resumevclass == -1) && !(index($0,interrupts["IT-MARK-
EVENT"])) &&\
              (time > interrupts[interrupts["IT-MARK-EVENT"] "=END"]))
             {
# not dealing with the event =END marker
              if (index($0,"SPEEDBRAKES"))
                {
# if this is a speedbrake value it must be > 4
                  split($2,arr,"=");
                  if (arr[2] > 4)
                    resumevclass = 2;
```

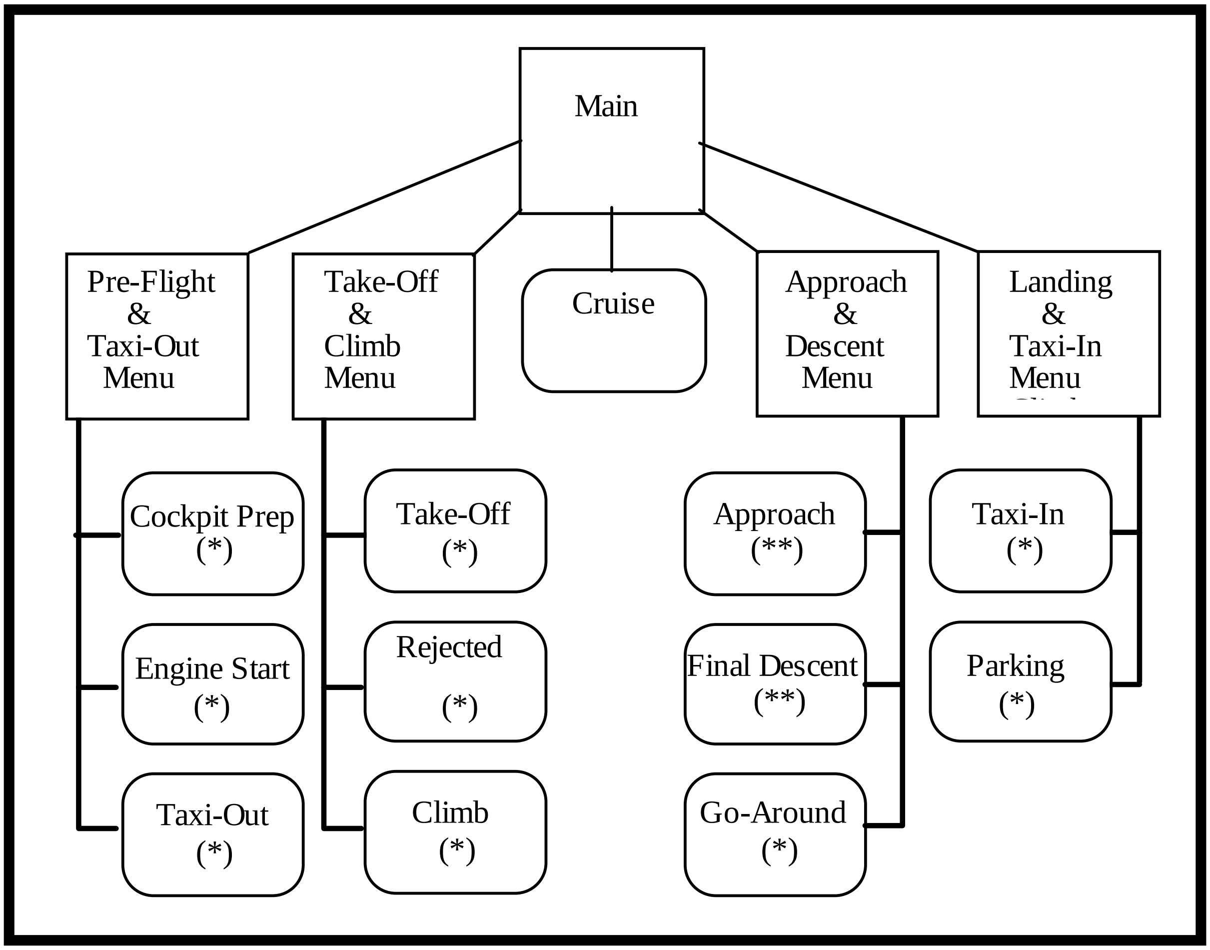
```
#            printf("\nin leg %d resuming with %s of class = %d",leg,$2,
resumevclass);
                 }
               else
                 {
# it this is in the table of resumptive events it is
# of class 2 or 3
                  i = 1;
                  while ((resumevclass == -1) && (i <= resumarrsize))
                    if (index($2,resumarr[i]))
                      {
                        if(index($0,"<ENERGY-CTRL>"))
                          resumevclass = 2;
                        else
                          resumevclass = 3;
#              printf("\nin leg %d resuming with %s of class = %d",leg,$2,
resumevclass);
                      }
                    else
                      i += 1;
                 }
               if (resumevclass == -1)
                 resumevclass = 0;
               if (DBG == 13)
                 printf("\nclassifying %s as %d\ncurrent time = %1.3f > int
ending %1.3f",\
                      $2,\
                      resumevclass,\
                      time,\
                      interrupts[interrupts["IT-MARK-EVENT"] "=END"]);

             }
         } # if interrupt has ended
             } # if((leg == 1) || (leg == 2) || (leg == 3))
         } # not at a waypoint crossing
      } # processing an event file

}
 END{
   printf("\n");
 }
```

Appendix 5.4

Checklist Menu Structure

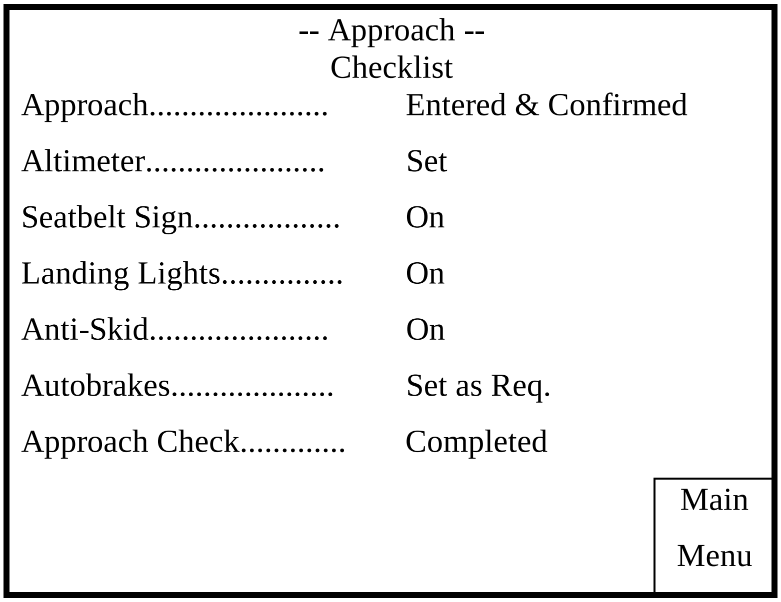


Legend
(*) not used in experimental scenario
(**) Approach and Final Descent checklists are displayed in Appendix 5.5

Menu Structure for the Touchscreen Checklist.

Appendix 5.5

Approach and Final Descent Checklists.

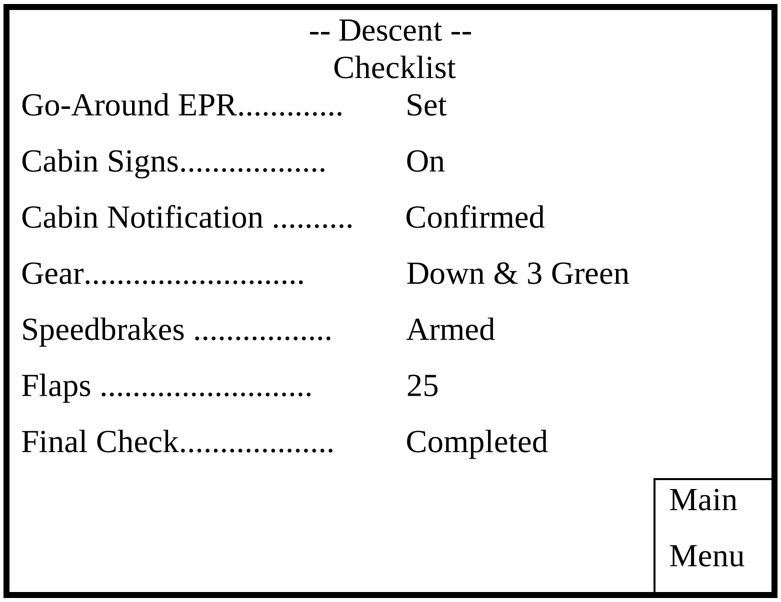


-- Approach --
Checklist

Approach......................	Entered & Confirmed
Altimeter......................	Set
Seatbelt Sign..................	On
Landing Lights...............	On
Anti-Skid......................	On
Autobrakes....................	Set as Req.
Approach Check.............	Completed

Main
Menu

Approach Checklist Screen.



-- Descent --
Checklist

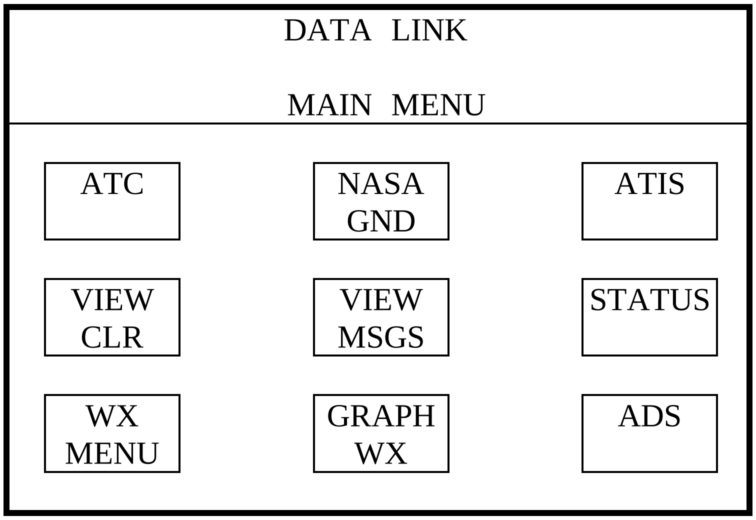
Go-Around EPR.............	Set
Cabin Signs..................	On
Cabin Notification	Confirmed
Gear...........................	Down & 3 Green
Speedbrakes	Armed
Flaps	25
Final Check...................	Completed

Main
Menu

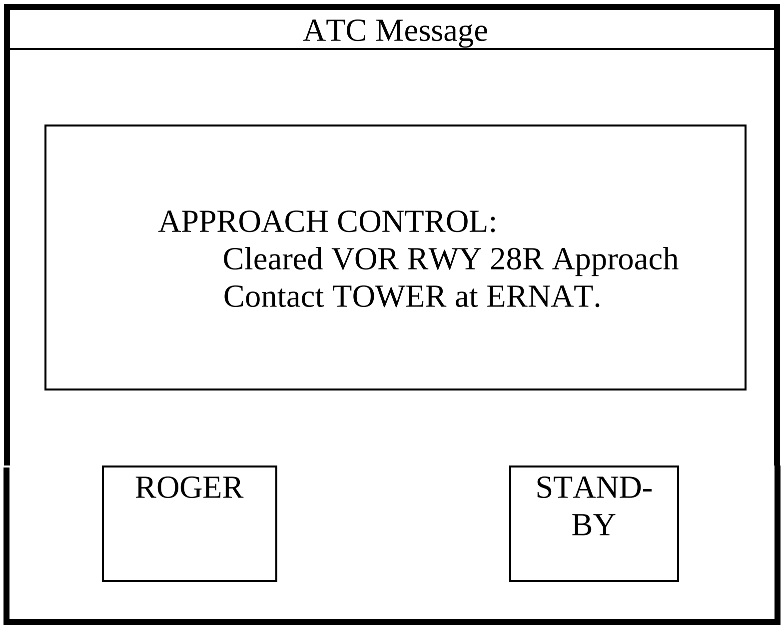
Final Descent Checklist Screen.

Appendix 5.6

Datalink Message Screen.



Datalink Initial Main Menu Screen.



APPROACH CONTROL:
Cleared VOR RWY 28R Approach
Contact TOWER at ERNAT.

Interrupting ATC Message Screen.

Appendix 5.7

Interruption Annunciation Messages

		Run-List1		Run-List2	
Script	Message	run	condition	run	condition
1.1	INCOMING MESSAGE. (machine voice)	12	11.07	12	11.07
		16	11.08	16	11.08
		20	11.06	20	21.06
		24	21.08	24	11.08
		28	21.06	28	11.06
		32	11.08	32	21.08
1.2	NASA 555, 'APPROACH	13	23.09	13	23.09
	Change crossing altitude at MAFAT	16	13.03	16	13.03
	to 6500, Over.	31	12.03	23	12.03
		32	13.03	24	13.03
1.3	NASA 555, 'APPROACH	11	23.04	11	23.04
	Change crossing speed at MAFAT	15	22.03	15	22.03
	to speed 160 knots, Over.	23	22.03	31	22.03
		24	23.03	32	23.03
2.1	NASA 555,'APPROACH	12	22.04	12	22.04
	In the event of missed approach,	24	22.02	19	22.06
	Climb and maintain 8000;	27	22.06	32	22.02
	Proceed direct MAFAT;				
	Hold S/E.				
2.2	NASA 555, 'TOWER'	31	23.10	23	23.10
	In the event of missed approach,				
	Climb and maintain 8000;				
	Proceed direct MAFAT;				
	Hold S/E.				
2.3	NASA 555,'APPROACH '	14	22.10	14	22.10
	In the event of missed approach,	17	23.05	18	23.02
	Climb and maintain 8000;	18	22.07	25	23.05
	Proceed direct MAFAB;	20	22.05	26	22.07
	Hold N/W.	22	22.10	28	22.05
		26	23.02	30	22.10

Appendix 5.7 (continued)

Interruption Annunciation Messages

Script	Message	Run-List1 run	Run-List1 condition	Run-List2 run	Run-List2 condition
3.1	NASA 555, 'TOWER' In the event of missed approach, Climb and maintain 8000; Proceed direct MAFAB; Hold N/W.	28 30	23.07 23.06	20 22	23.07 23.06
3.2	NASA 555, 'APPROACH', Runway 28 Right closed; Cleared VOR Runway28 Left approach.	11 13	12.09 12.08	11 13	12.09 12.08
3.3	NASA 555, 'APPROACH', Runway 1 Right closed; Cleared VOR Runway1 Left approach.	16 19 32	12.02 12.06 12.02	16 24 27	12.02 12.02 12.06
3.5	NASA 555, 'APPROACH', Runway 19 Left closed; Cleared VOR Runway19 Right approach.	25 30	13.05 12.10	17 22	13.05 12.10
3.6	NASA 555, 'APPROACH', Runway 10 Right closed; Cleared VOR Runway10 Left approach.	18 26 28	13.02 12.07 12.05	18 20 26	12.07 12.05 13.02
4.1	NASA 555, 'TOWER', Runway 28 Right closed; Runway 28 Left; Cleared to land.	12	13.08	12	13.08
4.3	NASA 555, 'TOWER', Runway 10 Left closed; Runway 10 Right; Cleared to land.	20	13.07	28	13.07
4.4	NASA 555, 'TOWER', Runway 1 Left closed; Runway 1 Right; Cleared to land.	15 23	13.10 13.10	15 31	13.10 13.10

Appendix 5.7 (continued)

Interruption Annunciation Messages

		Run-List1		Run-List2	
Script	Message	run	condition	run	condition
4.6	NASA 555, 'TOWER', Runway 19 Left closed; Runway 19 Right; Cleared to land.	14 22	13.06 13.06	14 30	13.06 13.06
5.2	NASA 555, 'APPROACH', Cleared VOR runway 28 Right approach. Contact tower at: ERNAT.	11 13	11.01 11.04	11 13	11.01 11.04
5.4	NASA 555, 'APPROACH', Cleared VOR runway 19 Left approach. Contact tower at: UNTRI.	14 17 22 25 30	11.03 11.09 11.03 21.09 21.03	14 17 22 25 30	11.03 21.09 21.03 11.09 11.03
5.6	NASA 555, 'APPROACH', Cleared VOR runway 1 Right approach. Contact tower at: INCRO.	19 27	11.02 21.02	19 27	21.02 11.02
5.11	NASA 555, 'APPROACH', Cleared VOR runway 10 Right approach. Contact tower at: YONKA.	18 26	11.05 21.05	18 26	21.05 11.05

Appendix 5.8

Flightpath Configurations.

	Flightpath Shapes (Direction of “doglegs” from Runway Heading)			
Initial Heading (Runways Used)	Shape 1 (Right, Right)	Shape 2 (Right, Left)	Shape 3 (Left, Right)	Shape 4 (Left, Left)
12^{0} (1RL)	Configuration 1	Configuration 2	Configuration 3	Configuration 4
102^{0} (10RL)	Configuration 5	Configuration 6	Configuration 7	Configuration 8
192^{0} (19RL)	Configuration 9	Configuration 10	Configuration 11	Configuration 12
282^{0} (28RL)	Configuration 13	Configuration 14	Configuration 15	Configuration 16

Appendix 5.9

Names for Flightpath Waypoints.

	Waypoint									
Config.	1	2	3	4	5	6	7	8 (*)	9 (*)	10
1	RALOF	BRUTO	TORPU	UCHAR	YOLIG	PORIT	INCRO	AP1	TD1	MAFAT
2	RALOF	BRUTO	TORPU	UCHAR	YOLIG	SANIS	INCRO	AP1	TD1	MAFAT
3	RALOF	TARAN	TORPU	UCHAR	YOLIG	PORIT	INCRO	AP1	TD1	MAFAT
4	RALOF	TARAN	TORPU	UCHAR	YOLIG	SANIS	INCRO	AP1	TD1	MAFAT
5	FIGIT	ASHAW	BRITO	VILAT	QUATI	NAZAN	YONKA	AP10	TD10	MAFAB
6	FIGIT	ASHAW	BRITO	VILAT	QUATI	ZANAS	YONKA	AP10	TD10	MAFAB
7	FIGIT	PARIN	BRITO	VILAT	QUATI	NAZAN	YONKA	AP10	TD10	MAFAB
8	FIGIT	PARIN	BRITO	VILAT	QUATI	ZANAS	YONKA	AP10	TD10	MAFAB
9	FLIAT	SILNE	VILAN	DORTA	FALIG	AYRIT	UNTRI	AP19	TD19	MAFAB
10	FLIAT	SILNE	VILAN	DORTA	FALIG	DILIN	UNTRI	AP19	TD19	MAFAB
11	FLIAT	SOLIG	VILAN	DORTA	FALIG	AYRIT	UNTRI	AP19	TD19	MAFAB
12	FLIAT	SOLIG	VILAN	DORTA	FALIG	DILIN	UNTRI	AP19	TD19	MAFAB
13	DALOF	KWOTI	NIRAV	NUNHE	PILAT	DALIX	ERNAT	AP28	TD28	MAFAT
14	DALOF	KWOTI	NIRAV	NUNHE	PILAT	PILAN	ERNAT	AP28	TD28	MAFAT
15	DALOF	ROLAT	NIRAV	NUNHE	PILAT	DALIX	ERNAT	AP28	TD28	MAFAT
16	DALOF	ROLAT	NIRAV	NUNHE	PILAT	PILAN	ERNAT	AP28	TD28	MAFAT

* The "AP##" and "TD##" also indicate which of the parallel runways is entered, *i.e*. "R" or "L"

Appendix 5.10

Activity-Level Descriptions of Procedures.

TOD Procedure Description	
•Tune Company Frequency	• Read Company frequency from kneepad form. • Turn large outer knob on COM2-LEFT to change integer value. • Turn small inner knob on COM2-LEFT to change decimal value.
•Tune ATIS Frequency	• Read ATIS frequency from kneepad form. • Turn large outer knob on COM2-RIGHT to change integer value. • Turn small inner knob on COM2-RIGHT to change decimal value.
•Listen to ATIS	• Move TFR toggle switch to RIGHT on COM2. • Move COM2 Listen-Toggle switch to UP • Listen to ATIS • Write: altimeter, tower frequency on kneepad form Remember: braking action • Move COM2 Listen-Toggle switch to DOWN
•Tune Tower Frequency	• Recall or Read Tower frequency from kneepad form. • Turn large outer knob on COM1-RIGHT to change integer value. • Turn small inner knob on COM1-RIGHT to change decimal value.
•Obtain Status Information	• Press INIT/REF button • Press Status line key (1R) • Place Check marks next to INOP items on kneepad form • Press LEGS button

18K’ Procedure Description

•Set Altimeters	• Press INIT/REF button on CDU • Press Perf/BarSet line key (3L) • Recall or Read altimeter from form (Obtained in ATIS). • Type "##.##" • Enter in field line key (3R) • Press LEGS button
•Contact Company	• Move TFR toggle switch to LEFT on COM2. • Turn Transmit-Knob to COM2. • Depress Mike-Switch • Speak: "NASA OPS, NASA 555; in range, for maintenance (read "INOP" items from kneepad) inop, request gate”. • Release Mike-Switch • Listen to Company: “NASA 555, Roger maintenance information, your gate is (gate)”. • Write down gate assignment on kneepad. • Depress Mike-Switch • Speak: “Roger gate (gate), NASA 555” • Release Mike-Switch • Turn Transmit-Knob to COM1.
•Obtain ETA Estimate & Calculate ETA-Local	• Press Index line key (6L) in CDU • Press Time line key (1L) • Calculate ETA-local: ETA-local=(ETA-zulu)-5hrs • Write down ETA-Local on kneepad • Press LEGS button
•Turn on Seatbelt Sign	• Press SEATBELTS button on Overhead Panel
•Announce to Cabin	• Turn Transmit-Knob to PA. • Depress Mike-Switch • Speak:"Ladies and Gentlemen, I have just turned on the 'seatbelt' sign. Please return to your seats and fasten your seatbelts. We will be arriving at approximately (ETA-local) to gate (gate)" • Release Mike-Switch • Turn Transmit-Knob to COM1.
•Turn on Landing Lights	• Press LANDING LIGHTS button on Overhead Panel
•Turn on Anti-Skid	• Press ANTI-SKID button on Overhead Panel
•Select Autobrakes	• Recall Braking action indicated in ATIS • Select (MIN/MED/MAX) Autobrakes button on Overhead Panel
•Do Approach Checklist	• Touch "Approach & Descent" Checklist Menu • Touch "Approach" Checklist • Recall or Look-at Altimeter Set to (##.##) • Recall or Look-at Approach Entered • Recall or Look-Overhead Seatbelt Sign ON • Recall or Look-Overhead Landing Lights ON • Recall or Look-Overhead Anti-Skid ON • Recall or Look-Notes & Recall or Look-Overhead Autobrakes Set. • Speak: "Approach Check Complete". • Touch "Main Menu" Checklist selection

FAF Procedure Description	
•<u>Select Go-Around EPR</u>	• Press N1 LIMIT button on CDU • Read or recall Go-Around EPR from notepad • Type "#.###" • Press Go-Around EPR line key (1L) • Press LEGS button
•<u>Contact Tower</u>	• Move TFR toggle switch to RIGHT on COM1. • Depress Mike-Switch • Speak:"AKRA tower, NASA 555, inbound from (FAF waypoint)". • Release Mike-Switch • Listen to Tower:"NASA 555, clear to land wind (dir) at (speed)". • Depress Mike-Switch • Speak: "Roger, cleared to land, wind (dir) at (speed)" • Release Mike-Switch
•<u>Obtain Target Speed (Vref)</u> & <u>Calculate Adjusted Speed</u>	• Press INIT/REF button on CDU • Press Approach line key (5L) • Read Vref30 (at line key (2R)) • Calculate Adjusted Target Speed= Vref30+.5(steady speed) • Write down Adjusted Target Speed • Press LEGS button
•<u>Turn on No Smoking Sign</u>	• Press "No Smoking" Sign (button) on Overhead Panel
•<u>Announce to Cabin</u>	• Turn Transmit-Knob to PA. • Depress Mike-Switch • Speak: "Ladies and Gentlemen, I have just turned on the 'No Smoking' sign. Please extinguish all cigarrettes at this time. Flight attendents- prepare for landing" • Release Mike-Switch • Turn Transmit-Knob to COM1.
•<u>Lower Gear</u>	• Lower Gear handle on Overhead Panel
•<u>Arm Speedbrakes</u>	• Pull Speedbrake Handle up & forward until unnotched.
•<u>Select Flaps 25</u>	• Pull Flaps Handle up & forward to drop in next notch.
•<u>Do Final Descent Checklist</u>	• Touch "Approach & Descent" Checklist Menu • Touch "Descent" Checklist • Recall Go-Around EPR Selected • Recall or Look-Overhead Signs ON ('seatbelts' and 'no smoking' signs) • Recall Cabin Notification • Recall or Look- Gear Down, no red lights & 3 green lights • Recall or Look-Throttle-Quadrant Speedbrake Armed. • Recall or Look-Throttle-Quadrant Flaps 25 • Speak: "Final Check complete" • Touch "Main Menu" Checklist selection

Appendix 5.11

Activity-Level Description of Interrupting Tasks.

Entering Initial Runway - Auditory Presentation.

- Hear "NASA 555, cleared VOR RWY 28R. Contact tower at (FAF)"
- Depress Mike-Switch
- Speak: "NASA 555: Roger runway 28R, contact at (FAF)"
- Release Mike-Switch
- Press DEP/ARR button
- Press ARR line key (2R)
- Press Desired-Runway line key (1-4 R,L)
- Press EXEC button
- Press LEGS button

Entering Initial Runway - Visual Presentation.

- Hear "INCOMING MESSAGE"
- Read: “NASA 555, cleared VOR RWY 28R. Contact tower at (FAF)”

(if not prepared to execute task immediately following acknowledgment:
Press "STAND-BY" on datalink screen)

- Press "ROGER" on Datalink Screen
- Press DEP/ARR button
- Press ARR line key (2R)
- Press Desired-Runway line key (1-4 R,L)
- Press EXEC button
- Press LEGS button

Change to Parallel Runway.

- Hear "NASA 555, runway 28R closed. Runway 28L clear to land."
- Depress Mike-Switch
- Speak: "NASA 555: Roger clear to land runway 28L"
- Release Mike-Switch
- Press DEP/ARR button
- Press ARR line key (2R)
- Press Desired-Runway line key (1-4 R,L)
- Press EXEC button
- Press LEGS button

Appendix 5.11 (continued)

Activity-Level Description of Interrupting Tasks.

Establish Holding Pattern at Missed Approach Fix.

- Hear: "NASA 555, in the event of a missed approach. Proceed direct MAFAT hold SE."
- Depress Mike-Switch
- Speak: "NASA 555: Roger hold at MAFAT"
- Release Mike-Switch
- Press HOLD button
- Press MAFAT waypoint linekey on (last L)
- Press Enter-Hold-Waypoint line key (6L)
- Press EXEC button
- Press LEGS button

Change Crossing Altitude at Missed Approach Fix.

- Hear:"NASA 555, Change crossing altitude at MAFAT to 6500."
- Depress Mike-Switch
- Speak: "NASA 555: Roger cross MAFAT at 6500"
- Release Mike-Switch
- Type "####"
- Press lineselect for MAFAT (#R)
- Press EXEC button

Change Crossing Speed at Missed Approach Fix.

- Hear"NASA 555, Change crossing speed at MAFAT to 160 knots"
- Depress Mike-Switch
- Speak: "NASA 555: Roger cross MAFAT at 160 knots"
- Release Mike-Switch
- Type "###/"
- Press lineselect for MAFAT (#R)
- Press EXEC button

Appendix 5.12

Subject Background Questionnaire.

1. General Information

Full Name: ________________________
First, Middle, Last

Address: ________________________ Street and Number, or P.O. Box

____
City, State, Zip Code, and Country (if not USA)

Home Phone: (___) _______ Work Phone: (____) ___
Area Code Number Area Code Number

Birth Date: _______
Month/Day/Year

Do you wear corrective lenses when you fly? Yes ○ No ○

2. General Experience Information

Current/Most Recent Airline:_________________

Current/Most Recent Position: _____________________ Captain, First Officer, Engineer, etc.

Are you currently flying military? Yes ○ No ○

Years Flying Commercial (approximate): _____

Years Flying Military (approximate): _____

Total Hours Flying (approximate): _____

Total Hours Flying as Pilot-in-Command (approximate): ______

Years of formal education: ____ (e.g. high school graduate = 12)

Appendix 5.12 (continued)

Subject Background Questionnaire.

3. Specific Aircraft Experience Information

Please list the types of aircraft on which you have experience, beginning with the most recently flown.

For each aircraft, please check the columns to indicate your approximate number of hours flying experience, and approximate number of hours simulator experience.

If you were an Instructor (I) or a Check Airman (CA) on any of these aircraft, please indicate by checking the last column.

Aircraft Type	**Hours in Type**			**Simulator Hours**			**I/CA ?**
	< 300	300-1000	> 1000	0	< 50	> 50	

Please check the appropriate column to indicate the approximate number of years of experience you have for each of the following categories:

Specific Aeronautical Experience	**Years Experience**		
	< 1	1-5	> 5
Long-range, Over-water (Class II) Operations (2 engines)			
Long-range, Over-water (Class II) Operations (> 2 engines)			
Total Multi-Engine (Captain or F/O, Military or Civil)			
Glass Cockpit (i.e. EFIS/CRT or FMS)			

Appendix 5.13

Task Ordering Exercise.

Instruction Set

As part of the simulation experiment, you will be asked to perform some Approach and Descent tasks in a specific order, as a "procedure". This specified order may be different from the order in which you would normally perform these tasks. For this reason, it is extremely important to understand how the order in which you would perform the task differs from the order required for the experiment.

(The following) table lists the tasks in alphabetical order and describes the task requirements. While some of these tasks are automated in certain aircraft or not required for domestic flights (i.e. Turning on the "No-Smoking" sign"), you should assume that you will need to perform all of the tasks listed. Please be sure you read and understand the specific task definitions for this experiment. This is important because there are some requirements unique to this scenario, e.g. the correct tower frequency is given in the ATIS, and communications require that you have previously obtained information to convey. You will be asked to arrange these tasks in the order in which you would perform them in a specific scenario. (The following figure) shows the profile and plan views of the scenario; a complex, step-down Approach and Descent with several turns and hard crossing restrictions at each waypoint. At the onset of the scenario, the entire flight path except for the runway and touchdown point, has been entered in the CDU. The scenario begins at the "Operate" Waypoint (20,000 feet, 290 KIAS). You must assume that you have not had the opportunity to perform any of these tasks prior to this point. For this scenario, you should also assume that you will be performing both pilot and co-pilot duties and that you will be manually (i.e. Attitude Control Wheel Steering) flying the aircraft. You should assume that all communication is through radio contact, i.e. that datalink is not available.

You are asked to: 1) re-arrrange the tasks defined in the table in the order in which you would perform them, 2) indicate the flight path segment (referencing the figure) in which you would perform each task, 3) indicate the rationale you used in ordering tasks as you did, for example sequential constraints or deadlines for performing a certain task. More detailed instructions are given on the response form, however there are several things to keep in mind when ordering these tasks:

- You are manually flying the aircraft as well as performing these tasks.
- Look at the flightpath to estimate how much time you would have in each interval.
- Consider the specific requirements for performing these tasks as stated in the table.

Alphabetical List of Tasks to Perform

Feel free to add your comments to this page.

Task Label	**Task Definition**
Calculate Adjusted Target Speed	Calculate by adding the gust and half of the steady winds to Vref30.
Enter Altimeter	Reset the altimeter from 29.92 to altimeter setting given in ATIS information.
Turn on Anti-skid	Turn on anti-skid
Do Approach Checklist	Verify that the following tasks have been performed: Altimeter Set, Approach Entered, Seatbelt Sign On, Landing-Lights On, Anti-skid On, Autobrakes Set.
Select Approach(runway)	Receive an approach clearance from ATC and enter this in the CDU. The current path is extended to include an approach point and touchdown point on the runway.
Pre-tune ATIS Frequency	Pre-tune the COM2 right head to a previously-specified frequency for obtaining ATIS information in the vicinity of the destination.
Obtain ATIS Information (tower freq. braking,altimeter)	Listen to the ATIS information. In addition to the usual weather and operations information, this ATIS information provides a new Tower frequency.
Select Autobrakes	Select the appropriate degree of braking.
Announce to Cabin: Gate, Seatbelt sign on, ETA-local.	Announce to the passengers that they must fasten their seatbelts, and inform them of the gate and ETA in Local time.
Announce to Cabin: No-Smoking, prepare to land.	Announce to the passengers that they must extinguish all smoking materials and to prepare for landing.
Call Company to give Status info & get Gate	Radio the company to inform them of any maintenance items (from the Status Information) and to obtain gate information.
Pre-tune Company Frequency	Pre-tune the COM2 left head to a previously-specified frequency for contacting the company in the vicinity of the destination.
Calculate ETA Local time	Convert the ETA-Zulu time to ETA-Local time.
Obtain ETA Zulu time	Obtain ETA estimate in Zulu time from the CDU.
Do Final Descent Checklist	Verify that the following tasks have been performed: Cabin Signs On, Go-Around EPR Set, Gear Down, Speedbrake Armed, Flaps 25.
Set Final Landing Flaps=30	Select final landing configuration flaps: Flaps 30.
Put Gear Down	Lower the gear
Enter Go-around EPR	Enter a previously-specified Go-around EPR in the CDU.
Set Initial Landing Flaps=25	Select initial landing configuration flaps: Flaps 25.
Turn on Landing Lights	Turn on landing lights
Turn on No-Smoking Sign	Turn on the No-Smoking sign.
Turn on Seatbelt Sign	Turn on the sign which instructs passengers to fasten seatbelts.
Set Speedbrakes	Arm the speedbrakes
Obtain Status Information	Obtain status information from the CDU in order to convey maintenance items to the company. You can assume that you have not received any alerts of faulty critical equipment during the flight.
Contact Tower near FAF & get Winds	Radio the tower to inform them you are at the outer marker and to obtain wind information for calculating Adjusted Target Speed.
Pre-tune Tower Frequency	Pre-tune the COM1 right head to the Tower frequency. The published tower frequency is inoperative and therefore you receive the correct tower frequency in the ATIS information.
Obtain Vref 30	Obtain Vref30 from the CDU as basis for Adjusted Target Speed.

Questionnaire Response Form

The first two columns below associate a **Task #** with each Task defined in the Table. The **Sequence** column lists the positions available in the sequence. Please enter the **Task #**s in the **Your Order** column in the order in which you would perform these tasks. For example, if you would perform the "Company Contact" task first, you would enter "11" in the **Your Order** column in the same row as the "1" position of the **Sequence** column. Please indicate in the **Interval** column, the number of the flightpath segment in which you would perform this task. For example, if you decide that you would perform this task in the second interval, you would enter a "2" in the first row of the **Interval** column. Please use the **Rationale** column to indicate if you considered any flight-path, task sequencing or other constraints when sequencing each task.

Task #	**Task Label** (definitions in the enclosed Table)
1	Calculate Adjusted Target Speed
2	Enter Altimeter
3	Turn on Anti-skid
4	Do Approach Checklist
5	Select Approach (runway)
6	Pre-tune ATIS Frequency
7	Obtain ATIS Information (tower freq. braking,altimeter)
8	Select Autobrakes
9	Announce to Cabin: Seatbelt sign on, ETA-local, gate, seatbelts
10	Announce to Cabin: No-Smoking, prepare to land.
11	Call Company to give Status info & get Gate info
12	Pre-tune Company Frequency
13	Calculate ETA Local time
14	Obtain ETA Zulu time
15	Do Final Descent Checklist
16	Set Final Landing Flaps=30
17	Put Gear Down
18	Enter Go-around EPR
19	Set Initial Landing Flaps=25
20	Turn on Landing Lights
21	Turn on No-Smoking Sign
22	Turn on Seatbelt Sign
23	Set Speedbrakes
24	Obtain Status Information
25	Contact Tower near FAF & get Winds
26	Pre-tune Tower Frequency
27	Obtain Vref 30

Sequence	**Your Order** (use task #)	**Interval** (see Figure)	**Rationale**
1 (first task)			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			
26			
27 (last task)			

Appendix 5.14

Training Phase 1 Flightpath Angle Instruction.

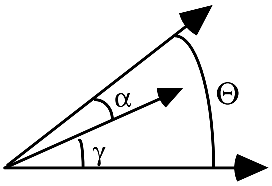
Using the FPA Diamond

What is Flight Path Angle?

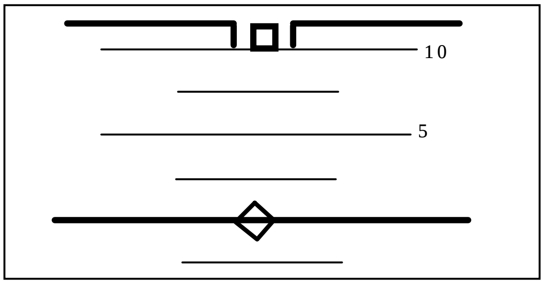
Flying to the Flight Path Angle (FPA) diamond is flying the center of gravity of the aircraft rather than, as when using pitch, the nose of the aircraft.

$$\text{Pitch} = \text{FPA} + \text{AOA}$$

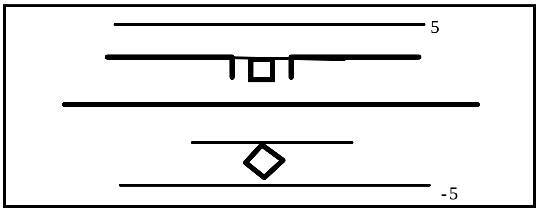
$$(\Theta = \gamma + \alpha)$$



Examples:
1) Level Stall: Pitch = 12, FPA = 0, AOA = 12.



2) Descent: Pitch = 2, FPA = -3, AOA = 5.



FPA and the Vertical Axis

- If you put the FPA diamond on the horizon line with the horizon line going directly through the widest part of the diamond, the aircraft will maintain level flight for any altitude, speed, or configuration.
- Raising the FPA diamond so that the diamond "sits" on the horizon line will result in approximately a 300-500 ft/minute speed decay. That is, you will lose approximately, 20 feet/second. Similarly, you will accelerate at that rate if you place the diamond under the horizon with its apex just touching the horizon line.

FPA and the Lateral Axis

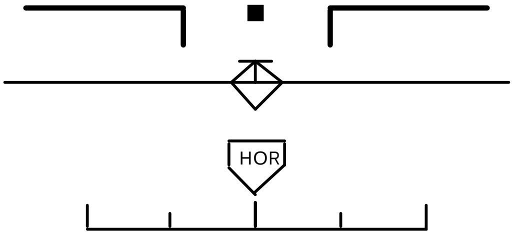
- No need to correct for winds by crabbing into the wind. It will automatically make these adjustments

Appendix 5.15

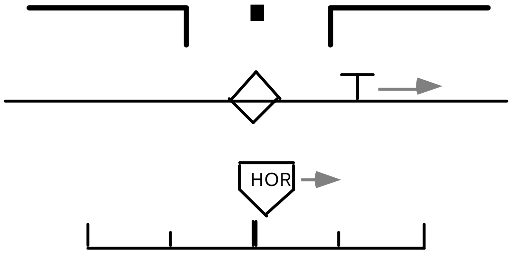
Training Phase 1 Lateral Control Instruction.

Using Lateral Guidance
(Refer to the following figure for a plan view of these steps.)

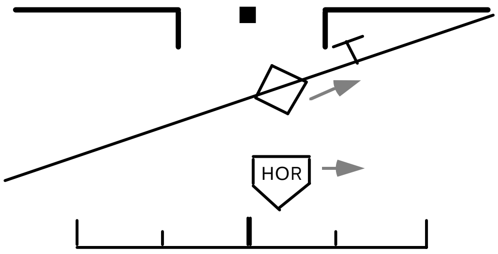
1. Prior to turn; zero bank angle; A/C on path. (Hor dev = 0)



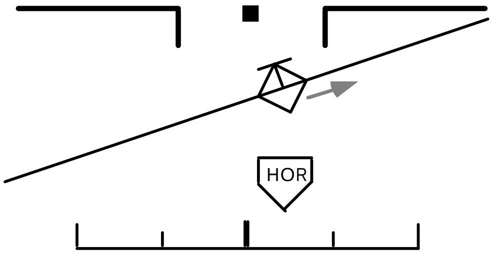
2. Pilot delays turn initiation. Thumb tack and HOR path indicator moving to the right.



3. Pilot rolls into 20° bank. HOR path indicator is still moving to the right since the FPA diamond is to the left of the thumb tack. FPA diamond is catching the thumb tack.



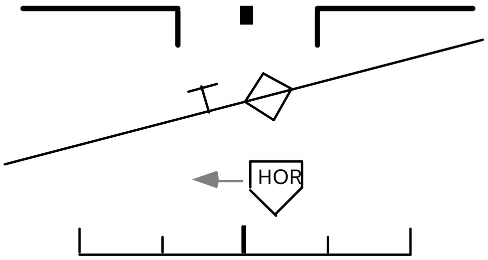
4. Since FPA diamond and thumb tack are at the same lateral position, the HOR path indicator will not move. Aircraft is on a parallel path, not the desired path.



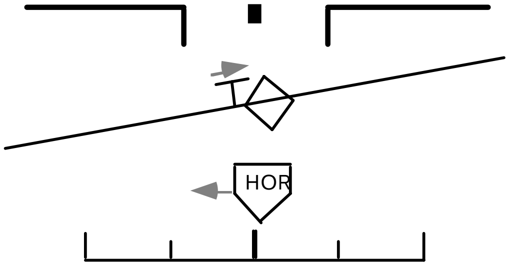
Appendix 5.15 (continued)

Training Phase 1 Lateral Control Instruction.

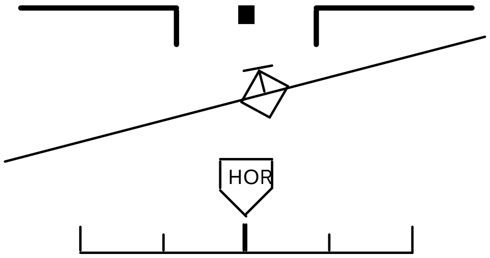
5. Pilot has reduced the bank angle to 15° and the distance between the FPA diamond and the thumb tack is constant. The HOR path indicator is now moving to the left.



6. As the HOR path indicator approaches the on-course mark, the pilot reduces the bank angle to 10° and the thumb tack moves closer to the FPA diamond.



7. When the aircraft is on course (HOR deviation = 0), the pilot increases the bank angle to 15° stabilizing the A/C in the turn.



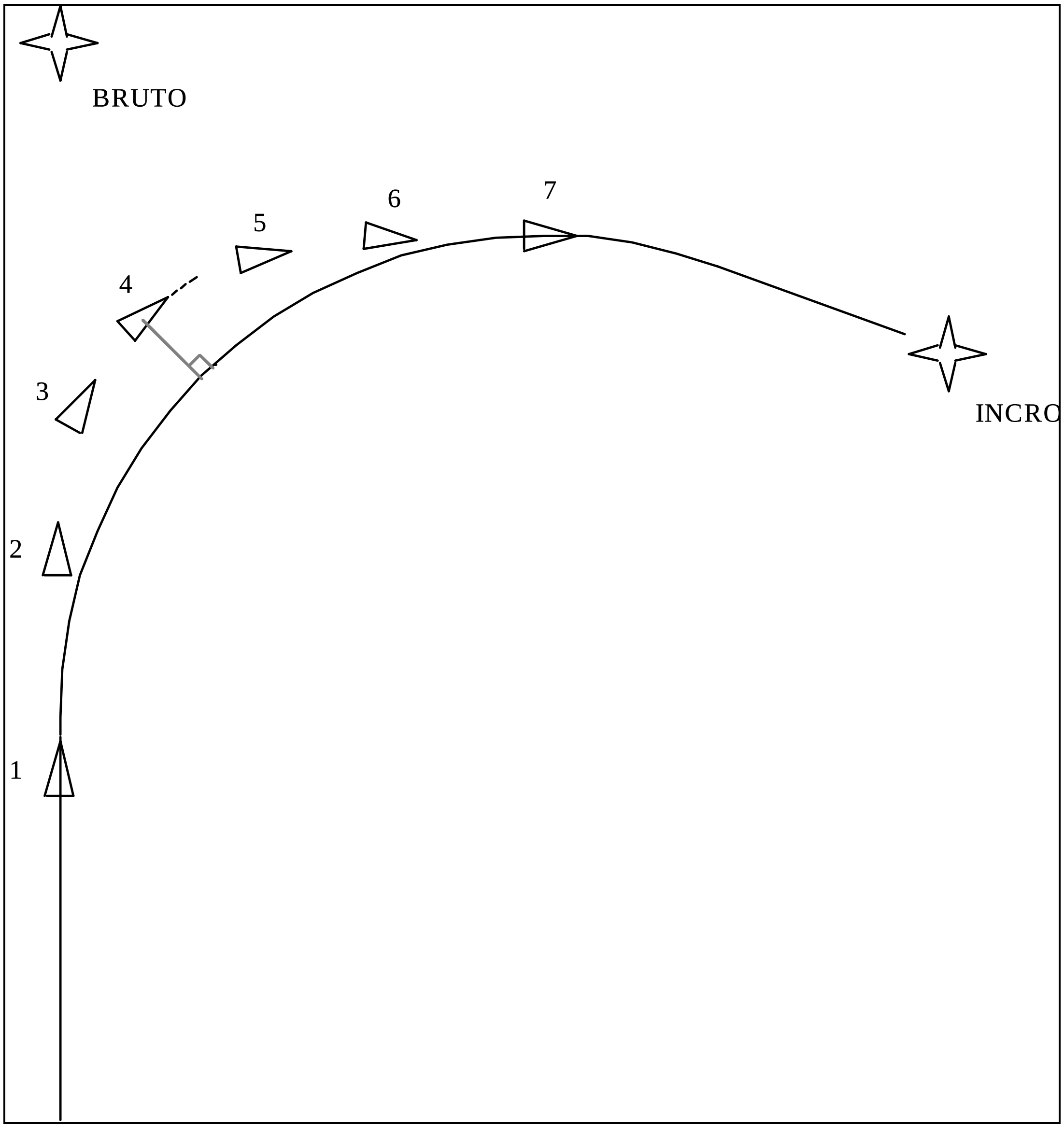
Notes:

- When the FPA diamond is aligned vertically with the thumb tack, the HOR path deviation indicator does not change. This does not mean thatthe path is correct, however, unless the lateral deviation is also zero.

- When the FPA diamond is to the left of the thumb tack, the HOR pathdeviation indicator moves to the right. When it is to the right, theHOR path deviation indicator moves to the left.

Appendix 5.15 (continued)

Training Phase 1 Lateral Control Instruction.



Plan View for Turning Exercise

Appendix 5.16

Phase 1 Instruction on PFD Guidance.

Developing A Scan

Approaching a Waypoint:

- When the THUMBTACK moves to indicate a turn, Bank 20^0 until you catch up to it, then 15^0 to maintain the turn.
- FPA DIAMOND should be approaching horizon.

Abeam of a Waypoint:

- When ALTITUDE BUG drops to new target altitude, and FPA REFERENCE LINE drops to the angle used in descending, lower the FPA DIAMOND to the FPA REFERENCE LINE, and reduce thrust to idle.

After Passing a Waypoint:

- Use relationship between the THUMBTACK, HORIZONTAL PATH INDICATOR, & FPA DIAMOND to know when to roll-out of turn.
- Glance at NAV display.

500' Above Target Altitude:

- Rehearse level-off procedure: Flaps required?, Constant Speed or Decelerating?

300' Above Target Altitude:

- When FPA REFERENCE LINE goes to the horizon, begin level-off by raising FPA DIAMOND to horizon.
- If the next to-waypoint requires a deceleration, the SPEED BUG will also change to the new target speed at 300' above the target altitude. If the SPEED BUG drops to a new target speed, wait until 5KIAS above the target speed to increase thrust. Else, manage throttles to maintain constant-speed throughout level-off.

Exceptions:

- SPEED BUG drops from 290KIAS to 240KIAS at 5nm to the second waypoint.
- When you take FLAPS-25, FPA REFERENCE LINE drops to -6.3^0 and the SPEED BUG drops from 150KIAS to 140KIAS. You do not need to make any flightpath inputs to adjust for this guidance, taking FLAPS-25 will gradually decay the speed to 140KIAS.

Lateral Corrections:

- Relationship between: the THUMBTACK, FPA DIAMOND, & HORIZONTAL PATH INDICATOR.

Minor Speed Corrections:

- Relationship between: FPA DIAMOND & FPA REFERENCE BAR

Appendix 5.17

The Sequential Coupling Task.

Sequential Coupling Instructions

For this assessment, consider that Task 1 must be performed before Task 2. Given this coupling, please indicate the strength of the sequential constraint on these two tasks. For example, if Task 1 must be performed immediately before Task 2, the strength of the sequential coupling would be high. If Task 2 need not follow Task 1 at all, if there is no advantage to this ordering, the strength of the sequential coupling would be low. Please rate the strength of the sequential constraint for each pair of tasks presented on the scale of 1-5; 1 represents a non-existent coupling, 5 represents an imperative coupling. The response form follows.

If you indicate there is some level of sequential coupling of Task 1 and Task 2 (higher than a rating of 3), please indicate the reason for this constraint. There may be several reasons why tasks may have a sequential coupling. Some of these binding principles are based on:

1.) LOGIC - the first task must be done before the second because it mechanically or functionally enables the second task.
2.) PROXIMITY - or "Flow", the second task is coupled to the first because they are physically near each other, or require utilization of the same resourse, for example, speech. You might say that you would perform the second of a pair of tasks which are proximally-coupled right away because it is more efficient to do it, for example, while your hand is already there.
3.) FUNCTION - the second task is coupled to the first because they are functionally related to each other, or require similar information to be foremost in your mind. You might say that you would perfor the second of a pair of tasks which are functionally-coupled right away because it is more efficient to do it, for example, while you are thinking about that goal.

There may be other binding principles by which you could describe why two tasks would have a strong sequential coupling. If you would like to express a coupling in other than these terms, please define the terms you use.

Driving Example
An analysis of sequential tasks' coupling strengths while driving on a dark and stormy evening.

Task 1	Task 2	Strength	Type of Coupling
turn on ignition	step on gas	1 2 3 4 **5**	logic - both required for engine to turn over.
turn on lights	turn on wipers	1 2 3 **4** 5	function - both to satisfy pre-driving conditions for visibility
turn on wipers	select radio station	**1** 2 3 4 5	no relation (could be rated higher if intent is to obtain weather information)

Sequential Coupling Assessment Response Form

Task 1	**Task 2**	**Strength of Coupling**	**Type of Coupling**
Pre-tune Company Frequency	Pre-tune ATIS Frequency	1 2 3 4 5	
Pre-tune ATIS Frequency	Obtain ATIS (contains: altimeter, braking conditions, tower frequency)	1 2 3 4 5	
Obtain ATIS (contains: altimeter, braking conditions, tower frequency)	Pre-tune Tower Frequency	1 2 3 4 5	
Pre-tune Tower Frequency	Obtain Status Information in CDU	1 2 3 4 5	
Crossing 18,000' Altitude	Set Altimeter in CDU	1 2 3 4 5	
Set Altimeter in CDU	Contact Company	1 2 3 4 5	
Contact Company	Obtain ETA Estimate in CDU	1 2 3 4 5	
Obtain ETA Estimate in CDU	Calculate ETA- Local Time	1 2 3 4 5	
Calculate ETA- Local Time	Turn on Seatbelt Sign	1 2 3 4 5	
Turn on Seatbelt Sign	Announce to Cabin (contains: gate, ETA-Local, Seatbelt Sign on)	1 2 3 4 5	
Announce to Cabin (contains: gate, ETA-Local, Seatbelt Sign on)	Turn on Landing Lights	1 2 3 4 5	
Turn on Landing Lights	Turn on Anti-skid	1 2 3 4 5	
Turn on Anti-skid	Select Appropriate Autobrakes	1 2 3 4 5	
Select Appropriate Autobrakes	Approach Checklist	1 2 3 4 5	
Cross Final Approach Fix	Enter Go-Around EPR in CDU	1 2 3 4 5	
Enter Go-Around EPR in CDU	Contact Tower: get winds	1 2 3 4 5	
Contact Tower: get winds	Obtain Vref30 Target Speed	1 2 3 4 5	
Obtain Vref30 Target Speed	Calculate Speed Adjusted for Wind	1 2 3 4 5	
Calculate Speed Adjusted for Wind	Turn on No-Smoking Sign	1 2 3 4 5	
Turn on No-Smoking Sign	Announce to Cabin (contains: prepare for landing, No Smoking)	1 2 3 4 5	
Announce to Cabin (contains: prepare for landing, No Smoking)	Lower Gear	1 2 3 4 5	
Lower Gear	Arm Speedbrakes	1 2 3 4 5	
Arm Speedbrakes	Select Flaps 25 (landing config.)	1 2 3 4 5	
Select Flaps 25 (landing config.)	Final Descent Checklist	1 2 3 4 5	

Forms of Coupling: LOGICAL, PROXIMAL, FUNCTIONAL, Other (please define)

Appendix 5.18

Phase 2 Flightpath Management Review.

Flightpath Management Review

1.) What's the most important thing about the first leg? *Keep your hands off the sidestick controller. Once it is in trim, it will maintain a constant bank and attitude. This is also what will allow you to perform all the procedural activities in the designated legs. The key to that is to roll-out aggressively and get stabilized.*

2.) When would you expect to receive a new target altitude? *At waypoints. Except for the first waypoint, when you are abeam a waypoint you will always get a new target altitude.*

3.) How can you tell that you are abeam a waypoint? *PFD: the waypoint name in the upper right corner changes. ND: crossing a waypoint star, the dme count=0 in the upper right corner. CDU: the legs page dme=0 and the passed waypoint dissapears.*

4.) What symbology indicates the new target altitude? *The green bug on and the green text below the altitude scale.*

5.) This symbology gives the target altitude, at wayoints you also start to...? *Descend.*

6.) What indicates the descent rate? What energy level do you use to descend? *The FPA reference bar, Idle*

7.) When would you expect to receive a new target speed? *If you get a new target speed, it will occur at 300' above your target altitude. The only speed change outside this rule is the one in the second leg which occurs 5nm before the next waypoint.*

8.) What else do you expect to see when you are 300' above a target altitude? *The FPA reference bar pops to the horizon, indicating a leveling-off.*

9.) What bank angle is instantaneously assumed by the Thumbtack? *Fifteen.*

10.) What bank angle should you initially assume to catch the Thumbtack?*Approximately 20.*

11.) What is the Flaps 1 speed?, Flaps 5 speed?, When you do take Flaps 15? *210 KIAS, 190 KIAS, At altitude = 8,300'. When 8000' level-off*

12.) Remember that about 500' before a level-off, you should ask yourself what type of level-off it is. Ask yourself two things, what are they? *Is it decelerating or constant-speed level-off, and are flaps required.*

13.) If you don't remember what type of level-off it is, where can you find out? *On the PFD; If the speed bug changes at 300' above the target altitude, it is not constant-speed. On the ND and CDU, compare the current speed bug value with the restriction for the to-waypoint.*

Appendix 5.19

Flightpath Management Instruction for Run 1 of Phase 2 Training.

Run 1 In-context Instruction

1. First leg. (at 19,000’)	• Small pitch changes (1/2 diamond; note v/s) • Use ND to anticipate turn • "T " to initiate 20° bank. • 15° bank when stabilized.
2. Turn to second leg & speed reduction. (at 19,000’)	• ND to anticipate roll out. • Roll out when "T" stops. • Take hand off side stick. • Throttle movement & speed tape. • Lead target airspeed by 5 kias (18° throttle angle)
3. Turn & descent from 19,000’ to 18,000’.	• Reinforce turn technique. • Altitude bug & FPA reference line change at wypts. • All descents at idle. • During descent, maintain speed with pitch. • Review L/O procedure 500’ above L/O alt. • FPA reference bar change 300' above L/O. (18,300')
4. Turn & descent from 18,000’ to 12,000’.	• Reinforce turn technique. • Reinforce pitch & thrust change technique at wypts. • Hands off side stick. • Review L/O technique 500’ above L/O (12,500') • FPA reference & speed bug change at 300' above L/O.(12,300')
5. Turn & descent from 12,000’ to 10,000’.	• Reinforce turn technique. • Reinforce pitch & thrust change at waypoint. • Reinforce airspeed control during descent. • Review L/O technique at 10,500’t. (constant speed) • Reinforce L/O technique.
6. Turn & descent from 10,000’ to 8,000’.	• Reinforce turn technique. • Reinforce pitch & thrust change at waypoint. • Review L/O technique at 8,500’. (decreasing airspeed) • When FPA changes, select flaps to 15°
7. Turn & descent from 8,000’ to 4,000’.	• Reinforce turn technique • Reinforce hands off side stick. • No pitch change when flap selected to 25°, only after speed decays to 140 kias. • Review L/O technique at 4,500’. (constant speed) • Reinforce constant speed level off technique.
8. Turn to runway and descent from 4,000’ to 3500’.	• Place FPA diamond on runway. • Select 30° flaps.(one more notch) • Achieve corrected reference speed.

Appendix 5.20

Interruption Annunciation and Performance Characteristics.

Intervening Task	IT Annunciation avg. time (sec)	IT performance (# keystrokes)
Initial Runway- Visual	1.5	5
Initial Runway- Auditory	5.8	5
Change Runway	5.8	5
Establish Hold	5.3	5
Change Altitude	5.5	6
Change Speed	5.5	6

Appendix 5.21

Definitions of Experimental Conditions.

Condition #s	IT	IP	Training	Testing
11.01, 21.01	IRA	Before TOD Leg	▨	
11.02, 21.02	IRA	Before TOD Procedure		*
11.03, 21.03	IRA	Between Tune company / Tune ATIS		*
11.04, 21.04	IRA	Between Tune ATIS / Obtain ATIS	▨	
11.05, 21.05	IRA	Within Obtain ATIS		▨
11.06, 21.06	IRV	Within Obtain ATIS		▨
11.07, 21.07	IRV	Between Tune Tower / Obtain Status	▨	
11.08, 21.08	IRV	Within Obtain Status		▨
11.09, 21.09	IRA	Within Obtain Status		▨
11.11, 21.11, 11.12, 21.12		No TOD Interruption	▨	▨
12.02	CR	Before 18K' Procedure		▨
12.03	CA	Within Altimeter Setting		▨
12.05	CR	Between Altimeter Set / Call company		▨
12.06	CR	Between Seatbelt sign / PA		▨
12.07	CR	Between landing lights / anti-skid		▨
12.08	EH	Between autobrakes / Approach checklist	▨	
12.10	CR	After 18K' Procedure		▨
22.02	EH	Before 18K' Procedure		▨
22.03	CS	Within Altimeter Setting		▨
22.05	EH	Between Altimeter Set / Call company		▨
22.06	EH	Between Seatbelt sign / PA		▨
22.07	EH	Between landing lights / anti-skid		▨
22.10	EH	After 18K' Procedure		▨
12.11, 22.11, 12.12, 22.12		No 18K' Interruption	▨	▨
13.02	CR	Before FAF Procedure		▨
13.03	CA	Within GA-EPR Setting		▨
13.05	CR	Between GA-EPR Setting / Call Tower		▨
13.06	CR	Between No Smoking / PA		▨
13.07	CR	Between Speedbrakes / Flaps 25		▨
13.08	CR	Between Flaps 25 / Final Descent checklist	▨	
13.10	CR	After FAF Procedure		▨
23.02	EH	Before FAF Procedure		▨
23.03	CS	Within GA-EPR Setting		▨
23.05	EH	Between GA-EPR Setting / Call Tower		▨
23.06	EH	Between No Smoking / PA		▨
23.07	EH	Between Speedbrakes / Flaps 25		▨
23.09	EH	Within Final Descent checklist	▨	
23.10	EH	After FAF Procedure		▨
13.11, 23.11, 13.12, 23.12		No FAF Interruption	▨	▨

* conditions in testing runs but not used in individual analysis of task factors

Appendix 5.22

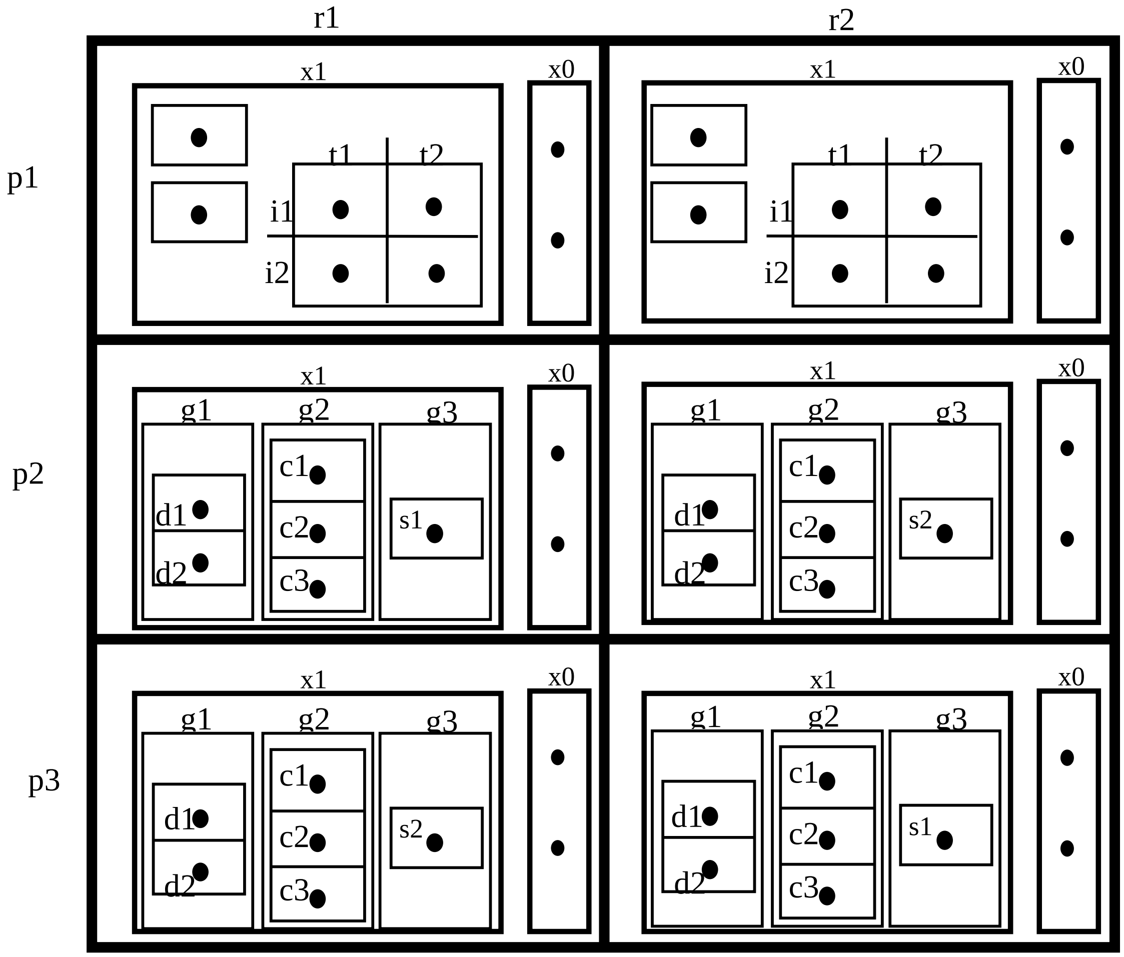
Composition of Runs.

		Run-List 1				Run-List 2			
run	**path**	run-type	TOD	18K’	FAF	run-type	TOD	18K’	FAF
1	**13**	Training FPM	11.12	12.12	13.12	Training FPM	11.12	12.12	13.12
2	**14**	Training FPM	11.12	12.12	13.12	Training FPM	11.12	12.12	13.12
3	**15**	Training FPM	11.12	12.12	13.12	Training FPM	11.12	12.12	13.12
4	**16**	Assess FPM	11.12	12.12	13.12	Assess FPM	11.12	12.12	13.12
5	**16**	Assess FPM	11.12	12.12	13.12	Assess FPM	11.12	12.12	13.12
6	**16**	Assess FPM	11.12	12.12	13.12	Assess FPM	11.12	12.12	13.12
7	**16**	Offline Procedure	11.12	12.12	13.12	Offline Procedure	11.12	12.12	13.12
8	**13**	Offline Procedure	11.12	12.12	13.12	Offline Procedure	11.12	12.12	13.12
9	**13**	Procedure Training	11.12	12.12	13.12	Procedure Training	11.12	12.12	13.12
10	**13**	Procedure Training	11.11	12.12	13.12	Procedure Training	11.11	12.12	13.12
11	**14**	Whole Training	11.01	12.09	23.04	Whole Training	11.01	12.09	23.04
12	**14**	Whole Training	11.07	22.04	13.08	Whole Training	11.07	22.04	13.08
13	**15**	Whole Training	11.04	12.08	23.09	Whole Training	11.04	12.08	23.09
14	**12**	Refresher-1	11.03	22.10	13.06	Refresher-1	11.03	22.10	13.06
15	**2**	Refresher-2	21.11	22.03	13.10	Refresher-2	21.11	22.03	13.10
16	**3**	Refresher-3	11.08	12.02	13.03	Refresher-3	11.08	12.02	13.03
17	**9**	Block A-Run1	11.09	12.11	23.05	Block B-Run1	21.09	22.11	13.05
18	**6**	Block A-Run2	11.05	22.07	13.02	Block B-Run2	21.05	12.07	23.02
19	**4**	Block A-Run3	11.02	12.06	23.11	Block B-Run3	21.02	22.06	13.11
20	**7**	Block A-Run4	11.06	22.05	13.07	Block B-Run4	21.06	12.05	23.07
21	**1**	Block A-Run5	11.12	12.12	23.12	Block B-Run5	21.12	22.12	13.12
22	**12**	Block A-Run6	11.03	22.10	13.06	Block B-Run6	21.03	12.10	23.06
23	**2**	Block A-Run7	21.11	22.03	13.10	Block B-Run7	11.11	12.03	23.10
24	**3**	Block A-Run8	21.08	22.02	23.03	Block B-Run8	11.08	12.02	13.03
25	**9**	Block B-Run1	21.09	22.11	13.05	Block A-Run1	11.09	12.11	23.05
26	**6**	Block B-Run2	21.05	12.07	23.02	Block A-Run2	11.05	22.07	13.02
27	**4**	Block B-Run3	21.02	22.06	13.11	Block A-Run3	11.02	12.06	23.11
28	**7**	Block B-Run4	21.06	12.05	23.07	Block A-Run4	11.06	22.05	13.07
29	**1**	Block B-Run5	21.12	22.12	13.12	Block A-Run5	11.12	12.12	23.12
30	**12**	Block B-Run6	21.03	12.10	23.06	Block A-Run6	11.03	22.10	13.06
31	**2**	Block B-Run7	11.11	12.03	23.10	Block A-Run7	21.11	22.03	13.10
32	**3**	Block B-Run8	11.08	12.02	13.03	Block A-Run8	21.08	22.02	23.03

Appendix 5.23

Experimental Data Partitioning & Statistical Models

Partitioning of Data for Each Subject



Variable Definitions:
r = replication (b: 1= first, 2= second)
p = procedure leg (e: 1= TOD, 1= 18K’, 3= FAF)
x = interrrupted procedure (f: 1=yes, 2=no)
t = interrupted task modality (h: 1= auditory, 2= visual)
i = interrupting task modality (j: 1= auditory, 2= visual)
g = goal-level (l: 1= outside procedure, 2= between tasks, 3= within task)
d = outside procedure (m: 1= before procedure, 2= after procedure)
c = coupling-strength/type (n: 1= low/uncoupled, 2= medium/physical, 3= high/functional)
s = similarity (q: 1= similar, 2= dissimilar)
k = subjects (v: 1-14)

Appendix 5.23

Experimental Data Partitioning & Statistical Models (continued).

Statistical Models for Hypothesis Tests.

Model for Analyzing Effects of Interruptions.

Source	Variable Type	EMS
Subject (S)	random	$\sigma_\varepsilon^2 + px\sigma_k^2$
Procedure Leg (PL)	fixed	$\sigma_\varepsilon^2 + kx\sigma_p^2 + x\sigma_{pk}^2$
Interruption (I)	fixed	$\sigma_\varepsilon^2 + kp\sigma_x^2 + p\sigma_{xk}^2$
I*PL	fixed	$\sigma_\varepsilon^2 + k\sigma_{px}^2 + \sigma_{xpk}^2$
S*PL	random	$\sigma_\varepsilon^2 + x\sigma_{pk}^2$
S*I	random	$\sigma_\varepsilon^2 + p\sigma_{xk}^2$
S*PL*I	random	$\sigma_\varepsilon^2 + \sigma_{xpk}^2$
residual		σ_ε^2

Model for Analyzing Effects of Modality.

Source	Variable Type	EMS
Subject (S)	random	$\sigma_\varepsilon^2 + rti\sigma_k^2$
replication (R)	random	$\sigma_\varepsilon^2 + kti\sigma_r^2$
Task Modality (T)	fixed	$\sigma_\varepsilon^2 + kri\sigma_t^2 + ri\sigma_{kt}^2 + r\sigma_{kti}^2$
Interrupt Modality (I)	fixed	$\sigma_\varepsilon^2 + krt\sigma_i^2 + kr\sigma_{it}^2 + r\sigma_{kti}^2$
T*I	fixed	$\sigma_\varepsilon^2 + kr\sigma_{it}^2 + r\sigma_{kti}^2$
S*T	random	$\sigma_\varepsilon^2 + ri\sigma_{kt}^2$
S*I	random	$\sigma_\varepsilon^2 + rt\sigma_{ki}^2$
S*T*I	random	$\sigma_\varepsilon^2 + r\sigma_{kti}^2$
residual		σ_ε^2

Appendix 5.23

Experimental Data Partitioning & Statistical Models (continued).

Model for Analyzing Effects of Goal-Level.

Source	Variable Type	EMS
Subject (S)	random	$\sigma_\varepsilon^2 + pg\sigma_k^2$
Procedure Leg (PL)	fixed	$\sigma_\varepsilon^2 + kg\sigma_p^2 + g\sigma_{kp}^2$
Goal-Level (GL)	fixed	$\sigma_\varepsilon^2 + kp\sigma_g^2 + p\sigma_{kg}^2$
S*PL	random	$\sigma_\varepsilon^2 + g\sigma_{kp}^2$
S*GL	random	$\sigma_\varepsilon^2 + p\sigma_{kg}^2$
PL*GL	fixed	$\sigma_\varepsilon^2 + k\sigma_{pg}^2 + \sigma_{kpg}^2$
S*PL*GL	random	$\sigma_\varepsilon^2 + \sigma_{kpg}^2$
residual		σ_ε^2

Model for Analyzing Effects of Coupling-Strength.

Source	Variable Type	EMS
Subject (S)	random	$\sigma_\varepsilon^2 + pc\sigma_k^2$
Procedure Leg (PL)	fixed	$\sigma_\varepsilon^2 + kc\sigma_p^2 + c\sigma_{kp}^2$
Coupling-Strength (CS)	fixed	$\sigma_\varepsilon^2 + kp\sigma_c^2 + p\sigma_{kc}^2$
S*PL	random	$\sigma_\varepsilon^2 + c\sigma_{kp}^2$
S*C	random	$\sigma_\varepsilon^2 + p\sigma_{kc}^2$
PL*C	fixed	$\sigma_\varepsilon^2 + k\sigma_{pc}^2 + \sigma_{kpc}^2$
S*PL*C	random	$\sigma_\varepsilon^2 + \sigma_{kpc}^2$
residual		σ_ε^2

Model for Analyzing Effects of Similarity.

Source	Variable Type	EMS
Subject (S)	random	$\sigma_\varepsilon^2 + ps\sigma_k^2$
Procedure Leg (PL)	fixed	$\sigma_\varepsilon^2 + ks\sigma_p^2 + s\sigma_{kp}^2$
Similarity (Si)	fixed	$\sigma_\varepsilon^2 + kp\sigma_s^2 + p\sigma_{ks}^2$
S*PL	random	$\sigma_\varepsilon^2 + s\sigma_{kp}^2$
S*Si	random	$\sigma_\varepsilon^2 + p\sigma_{ks}^2$
residual		σ_ε^2

Appendix 5.23

Experimental Data Partitioning & Statistical Models (continued).

Model for Analyzing Environmental Stress.

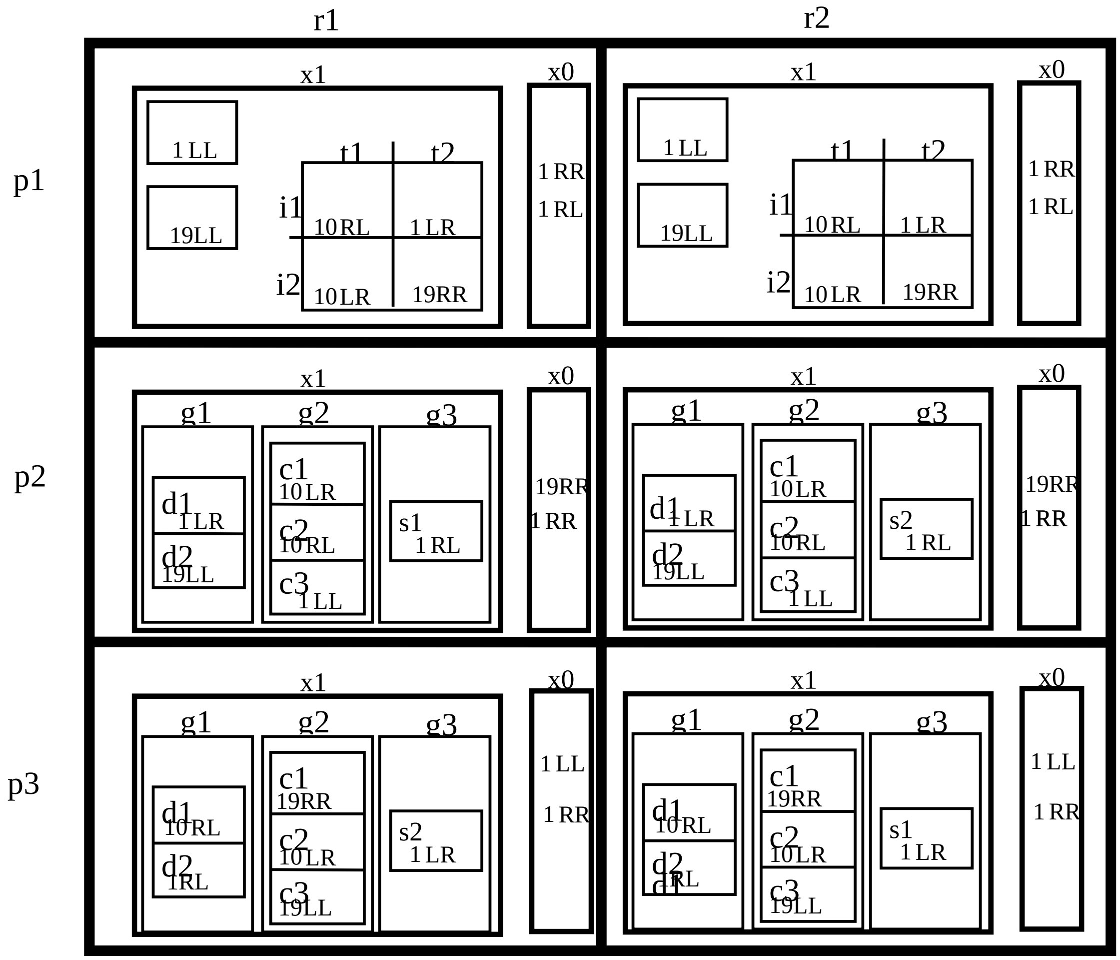
Source	Variable Type	EMS
Subject (S)	random	$\sigma_{\varepsilon}^{2} + rd_{1}\sigma_{k}^{2}$
Replication (R)	random	$\sigma_{\varepsilon}^{2} + kd_{1}\sigma_{r}^{2}$
Procedure Leg (PL)	fixed	$\sigma_{\varepsilon}^{2} + kr\sigma_{d1}^{2} + r\sigma_{k\,d1}^{2}$
S*PL	random	$\sigma_{\varepsilon}^{2} + r\sigma_{k\,d1}^{2}$
residual		σ_{ε}^{2}

Model for Analyzing Subject and Interruption Condition Differences.

Source	Variable Type	EMS
Subject (S)	random	$\sigma_{\varepsilon}^{2} + x_{1}\sigma_{k}^{2}$
Interrupt Conditions(X)	random	$\sigma_{\varepsilon}^{2} + kp\sigma_{x1}^{2}$
residual		σ_{ε}^{2}

Appendix 5.24

Allocation of Path-Types to Experimental Conditions



Appendix 6.1

Subjective Assessments and Designed FPM Difficulty Levels.

Analysis of Variance for Bedford Ratings of FPM Difficulty.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	1256.647	96.665	102.444	0.0001	Residual
Run (R)	2	2.070	1.035	0.186	0.8313	S * R
Design-level (DL)	3	803.361	267.787	90.985	0.0001	S * DL
S * R	24	133.475	5.561	5.894	0.0001	Residual
S * DL	39	114.784	2.943	3.119	0.0001	Residual
R * DL	6	1.037	0.173	0.253	0.9566	S * R * DL
S * R * DL	72	49.208	0.683	0.724	0.9551	Residual
Residual	534	503.876	0.944			

* Type II Sums of Squares

Bedford Rating Means by Design-Level.

Design-level	Count	Mean	Std.Dev.
0	134	2.515	1.128
1	160	4.128	1.702
2	200	5.096	1.870
3	200	5.416	1.927

Scheffé Tests on Design-Levels.

Design-level	Vs. Design-level	*S*	*p* -value
0	1	1.612	0.0001
	2	2.581	0.0001
	3	2.901	0.0001
1	2	0.968	0.0001
	3	1.289	0.0001
2	3	0.321	0.3353

Appendix 6.2

Individual FPM Difficulty Subjective Assessments

Subject 3's Means by Design-level and ANOVA Results.

Design-level	Count	Mean	Std.Dev.
0	9	2.555	0.389
1	12	2.996	0.410
2	15	3.906	0.683
3	15	4.333	1.148
F (3,39) = 13.801, *p* = 0.0001			

Subject 4's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	9	2.899	0.733
1	12	5.633	1.284
2	15	6.971	1.041
3	15	7.111	0.766
F (3,39) = 54.705, *p* = 0.0001			

Subject 5's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	9	3.735	0.475
1	12	5.611	0.996
2	15	6.767	1.551
3	15	6.889	1.167
F (3,39) = 5.709, *p* = 0.0024			

Subject 7's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	9	1.428	0.532
1	12	2.915	1.721
2	15	3.342	1.358
3	15	3.811	1.502
F (3,39) = 10.189, *p* = 0.0001			

Appendix 6.2 (continued)

Individual FPM Difficulty Subjective Assessments.

Subject 8's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	9	2.068	0.739
1	12	3.500	1.222
2	15	4.318	1.058
3	15	4.767	1.272
F (3,39) = 16.879, p = 0.0001			

Subject 9's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	9	2.847	1.030
1	12	5.062	1.103
2	15	5.944	1.079
3	15	6.867	1.141
F (3,39) = 25.423, p = 0.0001			

Subject 10's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	6	1.257	0.368
1	8	2.508	0.407
2	10	3.525	0.754
3	10	3.775	0.752
F (3,26) = 20.932, p = 0.0001			

Subject 11's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	12	3.830	1.203
1	12	6.183	1.068
2	15	7.878	0.256
3	15	7.702	0.583
F (3,42) = 61.312, p = 0.0001			

Appendix 6.2 (continued)

Individual FPM Difficulty Subjective Assessments.

Subject 12's Means by Design-level and ANOVA Result.

Design-level	Count	Mean	Std.Dev.
0	10	2.212	0.795
1	12	4.226	1.571
2	15	5.183	1.183
3	15	5.889	1.321
F (3,40) = 16.901, *p* = 0.0001			

Subject 13's Means by Design-level and ANOVA Result.

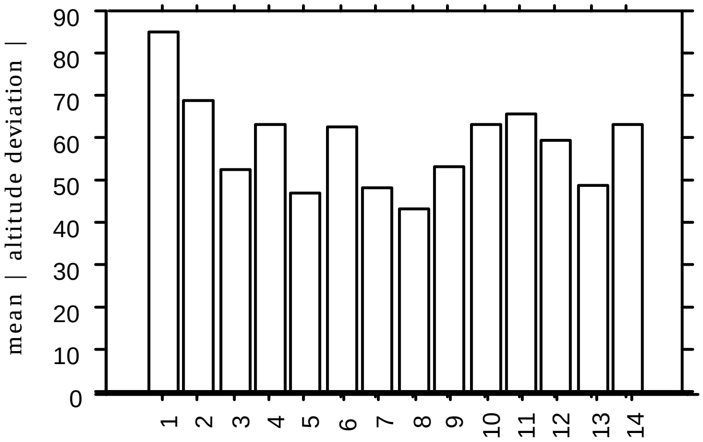
Design-level	Count	Mean	Std.Dev.
0	11	2.818	0.908
1	12	5.194	1.301
2	15	6.600	1.124
3	15	7.111	0.993
F (3,41) = 38.632, *p* = 0.0001			

Subject 14's Means by Design-level and ANOVA Result.

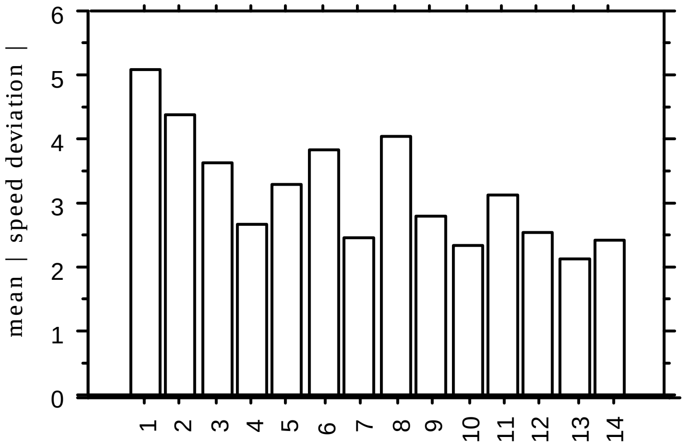
Design-level	Count	Mean	Std.Dev.
0	12	1.403	0.215
1	12	2.125	0.579
2	15	2.967	1.004
3	15	3.194	1.108
F (3,42) = 13.393, *p* = 0.0001			

Appendix 6.3.

Figures of FPM Deviations over Training Runs.



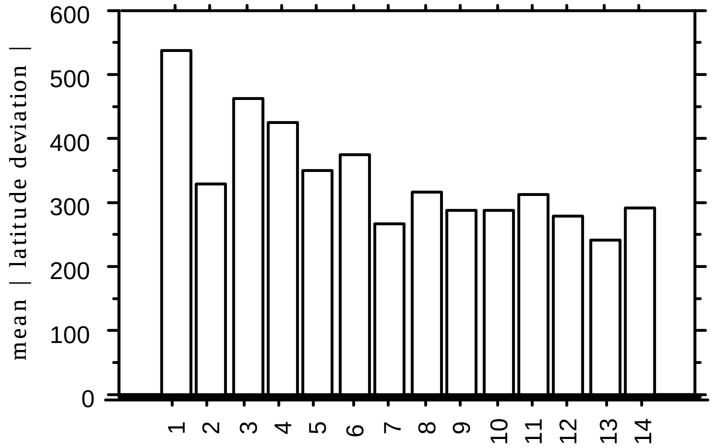
Mean Absolute Altitude Deviations over Training Runs.



Mean Absolute Speed Deviations over Training Runs.

Appendix 6.3 (continued)

Figures of FPM Deviations over Training Runs.



Mean Absolute Lateral Deviations over Training Runs.

Appendix 6.4.

FPM Criterion Assessment during Training.

Summary of FPM Criterion t-Tests*

Measure**	Subject	Mean	df	t-value	*p*-value
ADC	3	0.061	20	1.000	0.3293
ADC	6	0.218	20	1.000	0.3293
ADC	12	25.310	20	1.674	0.1097
ADC	14	3.215	20	1.070	0.2976
SDC	3	2.128	20	1.147	0.2649
SDC	6	0.200	20	1.000	0.3293
SDC	7	0.080	20	1.000	0.3293
SDC	8	1.210	20	1.118	0.2768
SDC	10	0.030	20	1.000	0.3293
SDC	14	0.689	20	1.000	0.3293
SDC	16	0.013	20	1.000	0.3293

* all other criterion measure means equalled zero.

** Altitude Deviation Criterion (ADC)
Speed Deviation Criterion (SDC)

Appendix 6.5

FPM Criterion Assessment prior to Testing.

Summary of FPM Criterion *t*-Tests*

Measure**	Subject	Mean	df	t-value	*p*-value
ADC	12	0.256	11	1.000	0.3388
ADC	14	6.006	11	1.000	0.3388
SDC	3	5.334	11	1.698	0.1176
SDC	5	1.238	11	1.000	0.3388
SDC	6	1.049	11	1.000	0.3388
SDC	8	3.006	11	1.000	0.3388
SDC	9	3.291	11	1.000	0.3388
SDC	13	1.103	11	1.000	0.3388
SDC	15	2.063	11	1.000	0.3388

* all other criterion measure means equalled zero.

** Altitude Deviation Criterion (ADC)
Speed Deviation Criterion (SDC)

Appendix 6.6

Speed Deviations on Runs prior to Testing.

Analysis of Variance for Absolute Speed Deviations

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	938.022	72.156	1.273	0.2363	Residual
Run (R)	1	143.359	143.359	3.907	0.0697	S * R
S * R	13	477.022	36.694	0.647	0.8102	Residual
Residual	138	7821.792	56.680			

* Type II Sums of Squares

Absolute Speed Deviation Means by Subject

Subject	Count	Mean	Std.Dev.
3	12	9.913	14.080
4	12	1.783	1.835
5	11	3.688	7.126
6	11	1.452	1.451
7	12	1.068	0.906
8	12	5.819	12.912
9	12	5.864	13.867
10	12	2.055	1.800
11	12	1.909	2.654
12	12	1.646	1.805
13	12	2.805	6.535
14	12	1.842	1.764
15	12	3.911	9.756
16	12	1.812	1.847

Appendix 6.7

Lateral Deviations on Runs prior to Testing.

Analysis of Variance for Absolute Lateral Deviations

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	910243.200	70018.708	1.237	0.2598	Residual
Run (R)	1	17204.021	17204.021	0.466	0.5068	S * R
S * R	13	479947.409	36919.031	0.652	0.8059	Residual
Residual	138	7812187.018	56610.051			

* Type II Sums of Squares

Absolute Lateral Deviation Means by Subject

Subject	Count	Mean	Std.Dev.
3	12	447.451	220.535
4	12	188.530	123.536
5	11	344.787	298.393
6	11	218.498	226.160
7	12	341.833	279.674
8	12	249.620	135.315
9	12	333.091	255.567
10	12	216.912	250.021
11	12	368.247	227.756
12	12	234.003	248.732
13	12	208.148	150.106
14	12	336.381	285.548
15	12	336.001	227.881
16	12	249.586	264.505

Appendix 6.8

Altitude Deviations on Runs prior to Testing.

Analysis of Variance for Absolute Altitude Deviations

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	47673.133	3667.164	2.028	0.0227	Residual
Run (R)	1	3610.388	3610.388	2.309	0.1526	S * R
S * R	13	20328.684	1563.745	0.865	0.5916	Residual
Residual	138	249591.712	1808.636			

* Type II Sums of Squares

Absolute Altitude Deviation Means by Subject

Subject	Count	Mean	Std.Dev.
3	12	63.378	44.455
4	12	51.378	45.776
5	11	64.443	44.876
6	11	34.171	23.668
7	12	37.248	38.203
8	12	24.716	23.860
9	12	63.506	48.133
10	12	28.718	24.639
11	12	36.983	28.983
12	12	47.323	66.624
13	12	16.173	9.295
14	12	74.577	83.639
15	12	32.229	25.976
16	12	31.821	21.602

Appendix 6.9

Summary of FPM Deviation Regressions over Testing Runs.

Regression Summary of FPM Deviations over Testing Runs

parameter	subject	intercept	slope	slope *p*-value	R^2	# > criterion
Altitude	3	53.341	-0.254	0.8023	0.0010	2
	4	37.085	0.255	0.6839	0.0020	0
	5	48.128	-0.183	0.8081	0.0010	0
	6	63.101	-1.514	0.0238 **	0.0460	1
	7	43.488	-0.045	0.9416	0.0005	0
	8	1.788	1.071	0.0438 **	0.0370	0
	9	40.484	-0.354	0.5616	0.0030	0
	10	47.808	-0.772	0.3069	0.0090	1
	11	27.252	0.121	0.8142	0.0010	0
	12	58.284	-1.329	0.0179 **	0.0500	0
	13	30.846	-0.341	0.3740	0.0070	0
	14	70.776	-0.753	0.4377	0.0050	3
	15	44.786	-0.349	0.5299	0.0040	0
	16	7.319	1.117	0.1623	0.0180	1
Lateral	3	766.766	-10.318	0.4908	0.0040	5
	4	156.236	-0.452	0.8569	0.0003	1
	5	409.216	-5.796	0.2636	0.0110	1
	6	682.193	-16.48	0.1635	0.0180	2
	7	94.788	8.121	0.1170	0.0220	0
	8	184.808	0.673	0.8399	0.0004	5
	9	293.857	1.380	0.8088	0.0010	1
	10	54.704	3.652	0.2167	0.0140	2
	11	379.916	-5.535	0.2139	0.0140	1
	12	155.436	-0.602	0.8396	0.0004	0
	13	136.154	3.136	0.4145	0.0060	0
	14	189.084	4.159	0.4108	0.0060	3
	15	284.315	-1.580	0.7323	0.0010	1
	16	325.282	-4.112	0.3641	0.0070	5
Speed	3	6.671	-0.172	0.1679	0.0170	5
	4	-0.607	0.085	0.3759	0.0070	0
	5	3.201	-0.061	0.2093	0.0140	0
	6	2.883	-0.016	0.7516	0.0010	2
	7	1.287	0.008	0.8064	0.0010	0
	8	-1.155	0.147	0.0737 *	0.0290	0
	9	2.368	-0.023	0.6070	0.0020	0
	10	0.803	0.041	0.5163	0.0040	0
	11	6.513	-0.183	0.0668 *	0.0300	0
	12	2.149	-0.034	0.1565	0.0180	0
	13	1.860	-0.038	0.0218 **	0.0470	0
	14	-4.605	0.301	0.0256 **	0.0440	0
	15	3.245	-0.052	0.5761	0.0030	0
	16	1.111	0.079	0.5826	0.0030	0

* $p < 0.10$, ** $p < 0.05$

Appendix 6.10

Subject Orderings of Procedural Tasks

Comparison of the Order from Each Subject with Designed Order.

Subject	Score	*tau*	*Z*-value	*p*-value
3	23	0.066	0.479	0.6316
4	19	0.054	0.396	0.6920
5	-5	-0.014	-0.104	0.9170
6	-7	-0.020	-0.146	0.8840
7	-21	-0.060	-0.438	0.6615
8	17	0.048	0.354	0.7230
9	-71	-0.202	-1.480	0.1388
10	19	0.054	0.396	0.6920
11	27	0.077	0.563	0.5735
12	17	0.048	0.354	0.7230
13	-29	-0.083	-0.605	0.5455
14	-73	-0.208	-1.522	0.1281
15	119	0.339	2.481	0.0131
16	37	0.105	0.771	0.4405

Comparison of Task Orders Among Subjects.

Subject														Designed Task Order
3	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	
24	6	24	14	24	6	27	3	14	14	6	24	6	24	5
27	12	14	13	12	7	6	21	13	13	12	14	12	12	12
6	7	13	27	6	14	7	6	6	12	7	13	7	11	6
7	13	6	6	14	13	14	7	7	6	27	12	26	6	7
14	14	7	12	11	22	13	13	12	24	1	11	24	7	26
13	11	12	7	7	9	12	12	24	11	5	22	14	14	24
12	24	11	26	13	18	11	11	11	7	18	9	13	13	2
11	21	27	18	22	27	9	22	2	2	14	6	11	1	11
20	22	2	24	9	2	5	4	22	27	13	7	22	2	14
2	2	3	11	5	20	1	2	9	22	24	2	9	20	13
4	9	20	22	27	5	3	9	20	9	11	15	2	22	22
18	15	22	9	1	21	8	27	3	20	3	20	20	9	9
5	4	9	2	2	10	18	1	5	3	8	3	5	3	20
22	27	23	20	20	3	20	26	10	8	9	25	3	10	3
9	3	5	3	18	8	22	18	21	4	2	5	8	21	8
26	1	4	8	8	4	24	5	26	26	20	27	4	19	4
21	8	21	4	3	26	2	17	25	5	22	4	27	26	18
10	20	10	21	4	12	4	8	1	21	4	1	18	27	25
25	18	18	10	26	17	10	23	27	10	26	8	21	4	27
1	10	26	5	21	23	21	15	17	18	21	21	10	5	1
17	5	25	25	10	19	19	24	18	17	10	10	25	8	21
3	26	1	23	25	15	26	10	23	23	17	23	1	17	10
8	25	19	1	23	25	25	19	8	19	19	26	17	16	17
23	19	17	17	19	1	17	16	19	15	16	19	19	23	23
19	17	16	19	15	16	16	25	4	25	23	18	23	25	19
16	23	8	15	16	24	23	20	16	1	25	17	16	15	15
15	16	15	16	17	11	15	14	15	16	15	16	15	18	16

Kendell's *W* Approximation to X^2 (df=13) = 50.499, $p < 0.0005$

Appendix 6.11

Perceived Coupling-Strengths Ratings.

Analysis of Variance for Coupling-Strength Ratings.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	49.063	3.774	6.290	0.0001	Residual
Procedure Leg (PL)	1	3.143	3.143	5.460	0.0361	S * PL
Designed-Type (DT)	2	197.696	98.848	98.581	0.0001	S * DT
PL * DT	2	0.218	0.109	0.223	0.8014	S * PL * DT
S * PL	13	7.482	0.576	0.959	0.4991	Residual
S * DT	26	26.070	1.003	1.671	0.0467	Residual
S * PL * DT	26	12.682	0.488	0.813	0.7169	Residual
Residual	70	42.000	0.600			

* Type II Sums of Squares

Coupling-Strength Rating Means by Designed Coupling Type

	Count	Mean	Std.Dev.
uncoupled	50	1.480	0.789
physically-coupled	52	3.462	1.196
functionally-coupled	52	4.231	0.854

Scheffé Tests on Coupling-Strength Rating Means by Designed Coupling Type.

		S	*p* -value
uncoupled	physically-coupled	0.515	0.0001
uncoupled	functionally-coupled	0.515	0.0001
physically-coupled	functionally-coupled	0.510	0.0024

Coupling-Strength Rating Means by Procedural Leg

	Count	Mean	Std.Dev.
18K’	77	2.935	1.463
FAF	77	3.221	1.536

Appendix 6.12

Perceived Coupling-Type Assignments.

Type assignment ratings for "18K' - Uncoupled" Designed Coupling Type

	count	sum of ranks	mean rank
uncoupled	*25*	*120.5*	*4.820*
functionally-coupled	25	65.5	2.620
physically-coupled	25	63.0	2.520
logically-coupled	25	63.0	2.520
other	25	63.0	2.520

Friedman Rank test: X(4)=41.48, p < 0.0001, #ties= 25
X(4)-adjusted for ties= 86.417, *p*-adjusted for ties < 0.0001
(two cases omitted due to missing values)

Type assignment ratings for "18K' - Functionally-coupled" Designed Coupling Type

	count	sum of ranks	mean rank
uncoupled	26	69.0	2.654
functionally-coupled	*26*	*99.0*	*3.808*
physically-coupled	26	66.5	2.558
logically-coupled	26	89.0	3.423
other	26	66.5	2.558

Friedman Rank test: X(4)=13.962, p =0.0074, #ties= 32
X(4)-adjusted for ties= 25.034, *p*-adjusted for ties < 0.0001
(one case omitted due to missing values)

Type assignment ratings for "18K' - Physically-coupled" Designed Coupling Type

	count	sum of ranks	mean rank
uncoupled	26	81.5	3.135
functionally-coupled	26	74.0	2.846
physically-coupled	*26*	*104.0*	*4.000*
logically-coupled	26	64.0	2.462
other	26	66.5	2.558

Friedman Rank test: X(4)=15.885, p = 0.0032, #ties= 28
X(4)-adjusted for ties= 30.593, *p*-adjusted for ties < 0.0001
(one case omitted due to missing values)

Appendix 6.12 (continued)

Perceived Coupling-Type Assignments.

Type assignment ratings for "FAF - Uncoupled" Designed Coupling Type

	count	sum of ranks	mean rank
uncoupled	*25*	*112.5*	*4.5*
functionally-coupled	25	72.5	2.9
physically-coupled	25	62.5	2.5
logically-coupled	25	65.0	2.6
other	25	62.5	2.5

Friedman Rank test: X(4)=29.2, p < 0.0001, #ties= 26
X(4)-adjusted for ties= 59.592, *p*-adjusted for ties < 0.0001
(two cases omitted due to missing values)

Type assignment ratings for "FAF - Functionally-coupled" Designed Coupling Type

	count	sum of ranks	mean rank
uncoupled	26	66.0	2.538
functionally-coupled	*26*	*96.0*	*3.692*
physically-coupled	26	63.5	2.442
logically-coupled	26	98.5	3.788
other	26	66.0	2.538

Friedman Rank test: X(4)=19.115, p = 0.0007, #ties= 33
X(4)-adjusted for ties= 33.695, *p*-adjusted for ties < 0.0001
(one case omitted due to missing values)

Type assignment ratings for "18K' - Physically-coupled" Designed Coupling Type

	count	sum of ranks	mean rank
uncoupled	26	77.5	2.981
functionally-coupled	26	77.5	2.981
physically-coupled	*26*	*90.0*	*3.462*
logically-coupled	26	77.5	2.981
other	26	67.5	2.596

Friedman Rank test: X(4)=3.923, p= 0.4165, #ties= 31
X(4)-adjusted for ties= 7.158, *p*-adjusted for ties= 0.1278
(one case omitted due to missing values)

Appendix 6.13

Percent Data Loss if Error Data Removed.

		Measure					
Analysis Factor	Level	acknT	initT	resT	resFPM	ensT	ensFPM
Interrupt Conditions	11.05 , 21.05	85.71	85.71	85.19	84.62	84.00	84.00
	11.06 , 21.06	42.86	38.46	42.11	41.18	37.50	37.50
	11.08 , 21.08	42.86	42.86	100.00	100.00	42.86	42.86
	11.09 , 21.09	57.14	55.56	86.67	88.89	54.17	54.17
	12.02 , 22.02	21.43	21.43	21.43	17.39	21.43	21.43
	12.03	50.00	50.00	50.00	46.15	46.15	46.15
	22.03	21.43	21.43	23.08	10.00	21.43	21.43
	12.05 , 22.05	35.71	35.71	37.04	37.04	37.04	37.04
	12.06 , 22.06	53.57	53.57	53.57	53.57	51.85	51.85
	12.07 , 22.07	14.81	14.81	16.00	16.00	14.81	14.81
	12.10 , 22.10	36.00	36.00	100	100	36.00	36.00
	13.02 , 23.02	38.46	38.46	100.00	37.50	33.33	33.33
	13.03	53.85	53.85	50.00	50.00	50.00	50.00
	23.03	57.14	57.14	57.14	57.14	53.85	53.85
	13.05 , 23.05	55.56	55.56	57.69	56.00	53.85	53.85
	13.06 , 23.06	50.00	50.00	51.85	51.85	44.00	44.00
	13.07 , 23.07	39.29	39.29	37.50	37.50	34.62	34.62
	13.10 , 23.10	65.38	64.00	100	100	64.00	64.00
Subjects	3	53.33	53.33	52.00	47.83	52.00	52.00
	4	16.13	16.13	9.09	9.09	13.33	13.33
	5	34.48	34.48	38.10	27.78	33.33	33.33
	6	38.71	38.71	47.83	45.45	36.67	36.67
	7	28.13	28.13	34.78	40.00	25.81	25.81
	8	59.38	58.06	60.71	60.71	55.17	55.17
	9	65.63	64.52	59.09	59.09	63.33	63.33
	10	65.63	65.63	79.17	80.95	65.63	65.63
	11	38.71	38.71	40.91	31.25	32.14	32.14
	12	41.94	40.00	38.10	38.10	33.33	33.33
	13	62.50	62.50	66.67	66.67	61.29	61.29
	14	59.38	58.06	57.69	54.55	58.06	58.06
	15	34.38	34.38	26.09	26.09	34.38	34.38
	16	38.71	38.71	41.67	41.67	36.67	36.67
Interrupted?	interrupted	*	*	*	*	*	41.11
	uninterrupted	*	*	*	*	*	21.74

"acknT" = acknowledgment time, "initT" = initiation time, "resT" = resumption time, "resFPM" = resumption FPM activity, "ensT" = ensemble performance time, "ensFPM" = ensemble FPM activity, * = analysis not performed

Appendix 6.13

Percent Data Loss if Error Data Removed (continued).

		Measure					
Analysis Factor	Level	acknT	initT	resT	resFPM	ensT	ensFPM
Task Modality	auditory	64.29	62.96	67.39	67.44	61.22	61.22
	visual	50.00	49.09	90.48	92.86	48.08	48.08
Interruption Modality	auditory	71.43	70.91	85.71	85.71	69.39	69.39
	visual	42.86	40.74	56.00	54.55	40.38	39.22
Modality Interactions	auditory/auditory	85.71	85.71	85.19	84.62	84.00	84.00
	auditory/visual	42.86	38.46	42.11	41.18	37.50	37.50
	visual/auditory	57.14	55.56	86.67	88.89	54.17	54.17
	visual/visual	42.86	42.86	100.00	100.00	42.86	42.86
Goal-Level	outside-proc	40.00	39.42	29.63	27.66	38.24	38.24
	between-task	41.57	41.57	42.68	42.31	39.24	39.24
	within-task	45.45	45.45	45.28	42.86	42.31	42.31
Coupling-Strength	low	45.45	45.45	47.17	46.15	45.28	45.28
	med	27.27	27.27	26.53	26.53	24.53	24.53
	high	51.79	51.79	52.73	52.73	48.08	48.08
Similarity	similar	39.29	39.29	42.86	44.44	34.62	34.62
	dissimilar	51.85	51.85	48.00	40.91	50.00	50.00
Stress Level	low	21.43	21.43	21.43	17.39	21.43	21.43
	high	38.46	38.46	38.46	37.50	33.33	33.33

"acknT" = acknowledgment time, "initT" = initiation time, "resT" = resumption time, "resFPM" = resumption FPM activity, "ensT" = ensemble performance time, "ensFPM" = ensemble FPM activity, * = analysis not performed

Appendix 6.14

Performing Interrupting Tasks.

Summary of Interruption Performance Measures

measure	count	mean	std. dev.	median	mode	10% trimmed mean
Acknowledgement Time	438	8.201	5.573	7.065	7.250	7.175
Initiation Time	434	7.709	8.694	5.470	*	6.350
Interruption Errors	438	0.171	0.464	0	0	0.057

* mode is undefined

Appendix 6.15

Effect of Interruption Conditions and Subjects on Acknowledgment Times.

Analysis of Variance for Acknowledgment Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value
Experimental Conditions	17	1998.651	117.568	4.881	.0001
Subjects	13	1777.062	132.697	5.675	.0001
Residual	407	9804.225	24.089		

* Type II Sums of Squares

Acknowledgment Time Means by Interruption Conditions

	Count	Mean	Std.Dev.
11.05 , 21.05	28	11.904	11.682
11.06 , 21.06	28	14.686	14.940
11.08 , 21.08	28	9.868	7.136
11.09 , 21.09	28	7.880	2.459
12.02 , 22.02	28	6.776	0.775
12.03	14	7.881	1.963
22.03	14	7.614	2.246
12.05 , 22.05	28	7.259	0.867
12.06 , 22.06	28	7.081	1.042
12.07 , 22.07	27	7.583	0.938
12.10 , 22.10	25	6.997	0.666
13.02 , 23.02	26	7.301	0.818
13.03	13	7.525	1.307
23.03	14	7.287	0.868
13.05 , 23.05	27	7.058	0.725
13.06 , 23.06	28	7.129	0.897
13.07 , 23.07	28	7.399	0.886
13.10 , 23.10	26	6.761	0.724

Appendix 6.15 (continued)

Effect of Interruption Conditions and Subjects on Acknowledgment Times

Acknowledgment Time Means by Subjects

	Count	Mean	Std.Dev.
3	30	9.631	3.497
4	31	7.608	1.150
5	29	7.644	1.808
6	31	7.090	1.349
7	32	9.472	7.583
8	32	7.658	5.769
9	32	6.917	0.754
10	32	6.504	0.588
11	31	6.470	0.942
12	31	7.208	1.113
13	32	7.801	3.513
14	32	14.271	14.187
15	32	9.664	7.383
16	31	6.731	1.214

Appendix 6.16

Effect of Interruption Conditions and Subjects on Interruption Initiation Times.

Analysis of Variance for Interruption Initiation Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value
Experimental Conditions	17	10095.955	593.880	11.646	.0001
Subjects	13	2109.950	162.304	3.183	.0001
Residual	403	20551.023	50.995		

* Type II Sums of Squares

Interruption Initiation Time Means by Interruption Conditions

	Count	Mean	Std.Dev.
11.05 , 21.05	28	12.009	10.725
11.06 , 21.06	26	23.668	20.322
11.08 , 21.08	28	10.952	9.704
11.09 , 21.09	27	10.197	9.113
12.02 , 22.02	28	4.850	2.403
12.03	14	7.484	2.841
22.03	14	10.079	4.747
12.05 , 22.05	28	3.766	3.570
12.06 , 22.06	28	3.561	3.441
12.07 , 22.07	27	5.422	2.904
12.10 , 22.10	25	5.095	3.867
13.02 , 23.02	26	6.112	3.893
13.03	13	6.763	2.671
23.03	14	9.552	5.011
13.05 , 23.05	27	3.780	2.772
13.06 , 23.06	28	4.567	3.799
13.07 , 23.07	28	7.016	4.761
13.10 , 23.10	25	5.780	5.556

Appendix 6.16 (continued)

Effect of Interruption Conditions and Subjects on Interruption Initiation Times.

Interruption Initiation Time Means by Subjects

	Count	Mean	Std.Dev.
3	30	6.351	9.176
4	31	7.869	4.861
5	29	10.326	10.117
6	31	8.539	10.547
7	32	4.570	2.940
8	31	7.155	6.803
9	31	6.144	6.801
10	32	7.958	11.629
11	31	8.672	9.766
12	30	11.955	11.163
13	32	4.172	2.781
14	31	5.796	4.396
15	32	7.992	7.488
16	31	10.888	12.942

Appendix 6.17

Effect of Interruption Conditions and Subjects on Interruption Performance Errors

Analysis of Variance for Interruption Performance Errors

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value
Experimental Conditions	17	4.911	0.289	1.386	0.1388
Subjects	13	4.468	0.344	1.650	0.0694
Residual	407	84.799	0.208		

* Type II Sums of Squares

Interruption Performance Error Means by Interruption Conditions

	Count	Mean	Std.Dev.
11.05 , 21.05	28	0.143	0.448
11.06 , 21.06	28	0.250	0.799
11.08 , 21.08	28	0.107	0.315
11.09 , 21.09	28	0.429	0.742
12.02 , 22.02	28	0.036	0.189
12.03	14	0.143	0.363
22.03	14	0.071	0.267
12.05 , 22.05	28	0.179	0.390
12.06 , 22.06	28	0.286	0.535
12.07 , 22.07	27	0.074	0.267
12.10 , 22.10	25	0.160	0.374
13.02 , 23.02	26	0.115	0.431
13.03	13	0.154	0.376
23.03	14	0.143	0.363
13.05 , 23.05	27	0.185	0.396
13.06 , 23.06	28	0.143	0.356
13.07 , 23.07	28	0.036	0.189
13.10 , 23.10	26	0.346	0.689

Appendix 6.17 (continued)

Effect of Interruption Conditions and Subjects on Interruption Performance Errors

Interruption Performance Error Means by Subjects

	Count	Mean	Std.Dev.
3	30	0.300	0.085
4	31	0.065	0.045
5	29	0.103	0.058
6	31	0.129	0.061
7	32	0.062	0.043
8	32	0.219	0.108
9	32	0.438	0.134
10	32	0.188	0.070
11	31	0.129	0.077
12	31	0.129	0.101
13	32	0.094	0.052
14	32	0.250	0.110
15	32	0.188	0.083
16	31	0.097	0.054

Appendix 6.18.

Effect of Interruption on Ensemble FPM Activity.

Analysis of Variance for Ensemble FPM Activity

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	3.056	0.235	31.038	.0001	Residual
Procedure Leg (PL)	2	3.132	1.566	35.822	.0001	S * PL
Interruption (I)	1	0.028	0.028	4.986	.0438	S * I
S * PL	26	1.137	0.044	5.772	.0001	Residual
S * I	13	0.072	0.006	0.735	.7290	Residual
PL * I	2	0.004	0.002	0.303	.7412	S * PL * I
S * PL * I	26	0.171	0.007	0.866	.6581	Residual
Residual	544	4.120	0.008			

* Type II Sums of Squares

Ensemble FPM Activity Means by Interruption Condition

	Count	Mean	Std.Dev.
Interrupted	467	0.160	0.142
Uninterrupted	161	0.146	0.122

Ensemble FPM Activity Means by Procedure Leg

	Count	Mean	Std.Dev.
TOD	215	0.107	0.086
18K	204	0.260	0.161
FAF	209	0.106	0.092

Scheffé Tests on Procedure Leg Means.

		S	*p* -value
TOD	18K’	0.053	0.9984
18K’	FAF	0.053	0.0001
FAF	TOD	0.053	0.0001

Appendix 6.19.

Effect of Interruption on Procedure Performance Errors.

Analysis of Variance for Procedure Performance Errors

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	47.476	3.652	6.853	0.0001	Residual
Procedure Leg (PL)	2	18.812	9.406	4.052	0.0294	S * PL
Interruption (I)	1	4.018	4.018	25.809	0.0002	S * I
S * PL	26	60.354	2.321	4.356	0.0001	Residual
S * I	13	2.024	0.156	0.292	0.9930	Residual
PL * I	2	1.509	0.754	1.402	0.2641	S * PL * I
S * PL * I	26	13.991	0.538	1.010	0.4518	Residual
Residual	588	313.333	0.533			

* Type II Sums of Squares

Procedure Performance Error Means by Interruption Condition

	Count	Mean	Std.Dev.
Interrupted	504	0.518	0.860
Uninterrupted	168	0.339	0.716

Procedure Performance Errors Means by Procedure Leg

	Count	Mean	Std.Dev.
TOD	224	0.357	0.566
18K	224	0.353	0.749
FAF	224	0.710	1.051

Scheffé Tests on Procedure Leg Means.

		S	*p* -value
TOD	18K’	0.004	0.9995
18K’	FAF	0.357	0.0632
FAF	TOD	0.353	0.0672

Examples of Operationally-significant Omissions.

	% of runs with omission	
task omitted	no interrupt	after interrupt
tune tower	2.9	17.5
obtain vref	---	5.0
descent check	1.8	8.4

Appendix 6.20

Effect of Interruption on Procedure Performance Times.

Comparison of Ensemble and Composite Performance Times

	Mean	Std. Dev.
Ensemble Times	111.014	19.059
Composite Times	112.644	16.973
t (242) = -1.672, *p* = 0.0958.		

Comparison of Ensemble and Composite Performance Times on Error-free Performance

	Mean	Std. Dev.
Ensemble Times	113.831	18.346
Composite Times	115.865	13.206
t (132) = -1.665, *p* = 0.0984.		

Appendix 6.21

Effects of Modality on Interruption Acknowledgment Times.

Analysis of Variance for Acknowledgment Times

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	6089.284	468.406	21.682	0.0001	Residual
Replication (R)	1	76.362	76.362	3.535	0.0654	Residual
Task Modality (TM)	1	547.461	547.461	4.303	0.0585	S * TM
Interrupt Modality (IM)	1	159.270	159.270	1.142	0.3046	S * IM
TM * IM	1	4.416	4.416	0.134	0.7204	S * TM * IM
S * TM	13	1653.840	127.218	5.889	0.0001	Residual
S * IM	13	1812.737	139.441	6.455	0.0001	Residual
S * TM * IM	13	429.164	33.013	1.528	0.1368	Residual
Residual	55	1188.205	21.604			

* Type II Sums of Squares

Acknowledgment Time Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	56	13.295	13.362
Visual	56	8.874	5.383

Acknowledgment Time Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	56	9.892	8.607
Visual	56	12.277	11.853

Acknowledment Time Means by Task Modality * Interruption Modality Interaction

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.
Auditory	Auditory	28	11.904	11.682
Auditory	Visual	28	14.686	14.940
Visual	Auditory	28	7.880	2.459
Visual	Visual	28	9.868	7.136

Same-Modality v. Cross-Modality contrast was not estimable.

Appendix 6.22

Effects of Modality on Interruption Initiation Times.

Analysis of Variance for Interruption Initiation Times

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	5194.344	399.565	5.099	0.0001	Residual
Replication (R)	1	236.002	236.002	3.011	0.0886	Residual
Task Modality (TM)	1	1600.309	1600.309	10.298	0.0068	S * TM
Interrupt Modality (IM)	1	1189.872	1189.872	3.159	0.0989	S * IM
TM * IM	1	1005.433	1005.433	6.976	0.0204	S * TM * IM
S * TM	13	2020.267	155.405	1.983	0.0413	Residual
S * IM	13	4897.318	376.717	4.807	0.0001	Residual
S * TM * IM	13	1873.774	144.136	1.839	0.0612	Residual
Residual	52	4075.108	78.367			

* Type II Sums of Squares

Interruption Initiation Time Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	54	17.623	16.970
Visual	55	10.581	9.340

Interruption Initiation Time Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	55	11.120	9.916
Visual	54	17.074	16.851

Interruption Initiation Time Means by Task Modality * Interrupt Modality Interaction.

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.	Scheffé tests *p*-values AV	VA	VV
Auditory	Auditory	28	12.009	10.725	0.0555	0.9767	0.9950
Auditory	Visual	26	23.668	20.322	---	0.0216	0.0311
Visual	Auditory	27	10.197	9.113		---	0.9982
Visual	Visual	28	10.951	9.707			---

Same-Modality v. Cross-Modality contrast

F (1,13) = 7.402, *p* = 0.0175

Appendix 6.23

Effects of Modality on Interruption Performance Errors.

Analysis of Variance for Interruption Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	2.964	0.228	0.578	0.8614	Residual
Replication (R)	1	1.286	1.286	3.257	0.0766	Residual
Task Modality (TM)	1	0.143	0.143	0.317	0.5830	S * TM
Interrupt Modality (IM)	1	0.321	0.321	0.807	0.3854	S * IM
TM * IM	1	1.286	1.286	5.200	0.0401	S * TM * IM
S * TM	13	5.857	0.451	1.141	0.3466	Residual
S * IM	13	5.179	0.398	1.009	0.4560	Residual
S * TM * IM	13	3.214	0.247	0.626	0.8216	Residual
Residual	55	21.714	0.395			

* Type II Sums of Squares

Interruption Performance Error Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	56	0.196	0.644
Visual	56	0.268	0.587

Interruption Performance Error Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	56	0.286	0.624
Visual	56	0.179	0.606

Interruption Performance Error Means by Task Modality * Interruption Modality

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.	Scheffé tests *p*-values		
					AV	VA	VV
Auditory	Auditory	28	0.143	0.448	0.9312	0.3845	0.9971
Auditory	Visual	28	0.250	0.799	---	0.7456	0.8533
Visual	Auditory	28	0.429	0.742		---	0.2818
Visual	Visual	28	0.107	0.315			---

Same-Modality v. Cross-Modality contrast

F (1,13) = 5.200, *p* = 0.0401

Appendix 6.24

Effects of Modality on Procedure Resumption Time.

Analysis of Variance of Procedure Resumption Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	721.509	55.501	0.933	0.5354	Residual
Replication (R)	1	36.516	36.516	0.614	0.4406	Residual
Task Modality (TM)	1	182.166	182.166	2.644	0.1384	S * TM
Interrupt Modality (IM)	1	7.921	7.921	0.362	0.5588	S * IM
TM * IM	1	1.535	1.535	0.189	0.6932	S * TM * IM
S * TM	9	620.180	68.909	1.159	0.3618	Residual
S * IM	12	262.864	21.905	0.368	0.9631	Residual
S * TM * IM	3	24.382	8.127	0.137	0.9372	Residual
Residual	25	1486.615	59.465			

* Type II Sums of Squares

Procedure Resumption Time Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	46	5.966	8.082
Visual	21	2.339	2.823

Procedure Resumption Time Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	42	3.978	5.963
Visual	25	6.258	8.538

Procedure Resumption Time Means by Task Modality * Interruption Modality Interaction

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.
Auditory	Auditory	27	5.246	7.052
Auditory	Visual	19	6.989	9.465
Visual	Auditory	15	1.697	1.759
Visual	Visual	6	3.943	4.345

Same-Modality v. Cross-Modality contrast

Appendix 6.25

Effects of Modality on Resumptive FPM Activity.

Analysis of Variance for Resumptive FPM Activity

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	0.455	0.035	0.904	0.5643	Residual
Replication (R)	1	0.012	0.012	0.310	0.5841	Residual
Task Modality (TM)	1	0.005	0.005	0.415	0.5398	S * TM
Interrupt Modality (IM)	1	0.013	0.013	1.498	0.2466	S * IM
TM * IM	1	0.031	0.031	1.473	0.3488	S * TM * IM
S * TM	7	0.086	0.012	0.317	0.9371	Residual
S * IM	11	0.095	0.009	0.222	0.9929	Residual
S * TM * IM	2	0.042	0.021	0.541	0.5906	Residual
Residual	19	0.736	0.039			

* Type II Sums of Squares

Resumptive FPM Activity Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	43	0.570	0.119
Visual	14	0.087	0.267

Resumptive FPM Activity Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	35	0.074	0.191
Visual	22	0.049	0.111

Resumptive FPM Activity Means by Task Modality * Interruption Modality Interaction

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.
Auditory	Auditory	26	0.053	0.113
Auditory	Visual	17	0.064	0.124
Visual	Auditory	9	0.135	0.330
Visual	Visual	5	0.000	0.000

Same-Modality v. Cross-Modality contrast was not estimable.

Appendix 6.26

Effects of Modality on Ensemble Performance Time.

Analysis of Variance for Ensemble Performance Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	3377.754	259.827	2.374	0.0161	Residual
Replication (R)	1	694.607	694.607	6.345	0.0154	Residual
Task Modality (TM)	1	135.721	135.721	1.005	0.3345	S * TM
Interrupt Modality (IM)	1	1231.986	1231.986	10.674	0.0061	S * IM
TM * IM	1	99.096	99.096	1.347	0.2684	S * TM * IM
S * TM	13	1756.329	135.102	1.234	0.2881	Residual
S * IM	13	1500.452	115.419	1.054	0.4205	Residual
S * TM * IM	12	882.888	73.574	0.672	0.7683	Residual
Residual	45	4926.098	109.469			

* Type II Sums of Squares

Ensemble Performance Time Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	49	86.569	13.462
Visual	52	84.980	10.872

Ensemble Performance Time Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	49	89.526	12.717
Visual	52	82.194	10.546

Ensemble Performance Time Means by Task Modality * Interruption Modality

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.
Auditory	Auditory	25	91.043	15.476
Auditory	Visual	24	81.908	9.158
Visual	Auditory	24	87.945	9.081
Visual	Visual	28	82.439	11.768

Same-Modality v. Cross-Modality contrast was not estimable.

Appendix 6.27

Effects of Modality on Procedure Performance Errors.

Analysis of Variance Procedure Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	4.107	0.316	1.600	0.1133	Residual
Replication (R)	1	0.143	0.143	0.724	0.3986	Residual
Task Modality (TM)	1	2.893	2.893	4.500	0.0537	S * TM
Interrupt Modality (IM)	1	5.143	5.143	16.278	0.0014	S * IM
TM * IM	1	1.750	1.750	9.100	0.0099	S * TM * IM
S * TM	13	8.357	0.643	3.257	0.0011	Residual
S * IM	13	4.107	0.316	1.600	0.1133	Residual
S * TM * IM	13	2.500	0.192	0.974	0.4876	Residual
Residual	55	10.857	0.197			

* Type II Sums of Squares

Procedure Performance Error Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	56	0.625	0.648
Visual	56	0.304	0.502

Procedure Performance Error Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	56	0.679	0.664
Visual	56	0.250	0.437

Procedure Performance Error Means by Task Modality * Interruption Modality Interaction

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.	Scheffé tests *p*-values		
					AV	VA	VV
Auditory	Auditory	28	0.964	0.637	0.0027	0.0146	0.0008
Auditory	Visual	28	0.286	0.460	---	0.9357	0.9795
Visual	Auditory	28	0.393	0.567		---	0.7624
Visual	Visual	28	0.214	0.418			---

Same-Modality v. Cross-Modality contrast

F (1, 13) = 9.100, *p* = 0.0099

Appendix 6.28

Effects of Modality on Ensemble FPM Activity.

Analysis of Variance Ensemble FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subjects (S)	13	0.323	0.025	3.087	0.0025	Residual
Replication (R)	1	0.052	0.052	6.403	0.0150	Residual
Task Modality (TM)	1	6.155E-5	6.155E-5	0.015	0.9032	S * TM
Interrupt Modality (IM)	1	9.998E-5	9.998E-5	0.030	0.8660	S * IM
TM * IM	1	0.010	0.010	0.839	0.3777	S * TM * IM
S * TM	13	0.052	0.004	0.497	0.9149	Residual
S * IM	13	0.044	0.003	0.419	0.9549	Residual
S * TM * IM	12	0.137	0.011	1.421	0.1919	Residual
Residual	45	0.363	0.008			

* Type II Sums of Squares

Ensemble FPM Activity Means by Task Modality

	Count	Mean	Std.Dev.
Auditory	49	0.096	0.080
Visual	52	0.101	0.115

Ensemble FPM Activity Means by Interruption Modality

	Count	Mean	Std.Dev.
Auditory	49	0.101	0.088
Visual	51	0.096	0.110

Ensemble FPM Activity Means by Task Modality * Interruption Modality

Task Modality	Interrupt Modality	Count	Mean	Std.Dev.
Auditory	Auditory	25	0.109	0.093
Auditory	Visual	24	0.081	0.063
Visual	Auditory	24	0.093	0.083
Visual	Visual	28	0.108	0.138

Same-Modality v. Cross-Modality contrast was not estimable.

Appendix 6.29

Effects of Goal-Level on Interruption Acknowledgment Time.

Analysis of Variance for Interruption Acknowledgment Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	89.928	6.918	13.795	0.0001	Residual
Procedure Leg (PL)	1	0.228	0.228	0.317	0.5830	S * PL
Goal Level (GL)	2	12.761	6.380	1.910	0.1684	S * GL
S * PL	13	9.361	0.720	1.436	0.1434	Residual
S * GL	26	86.875	3.341	6.663	0.0001	Residual
PL * GL	2	2.668	1.334	0.999	0.3820	S * PL * GL
S * PL * GL	26	34.730	1.336	2.664	0.0001	Residual
Residual	242	121.355	0.501			

* Type II Sums of Squares

Interruption Acknowledgment Time Means by Goal-Level

	Count	Mean	Std.Dev.
Outside Procedure	105	6.955	0.770
Between Tasks	166	7.251	0.904
Within Task	55	7.578	1.658

Scheffé Tests on Goal-Level Means.

		S	*p* -value
Outside Procedure	Between Tasks	0.296	0.4423
Between Tasks	Within Task	0.327	0.5243
Within Task	Outside Procedure	0.623	0.1432

Appendix 6.30

Effects of Goal-Level on Interruption Initiation Time.

Analysis of Variance for Interruption Initiation Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	483.594	37.200	2.679	0.0016	Residual
Procedure Leg (PL)	1	39.559	39.559	2.308	0.1526	S * PL
Goal Level (GL)	2	616.149	308.075	16.192	0.0001	S * GL
S * PL	13	222.810	17.139	1.235	0.2552	Residual
S * GL	26	494.677	19.026	1.370	0.1149	Residual
PL * GL	2	24.416	12.208	1.054	0.3629	S * PL * GL
S * PL * GL	26	301.079	11.580	0.834	0.7005	Residual
Residual	241	3345.871	13.883			

* Type II Sums of Squares

Interruption Initiation Time Means by Goal-Level.

	Count	Mean	Std.Dev.
Outside Procedure	104	5.448	4.017
Between Tasks	166	4.687	3.760
Within Task	55	8.501	4.114

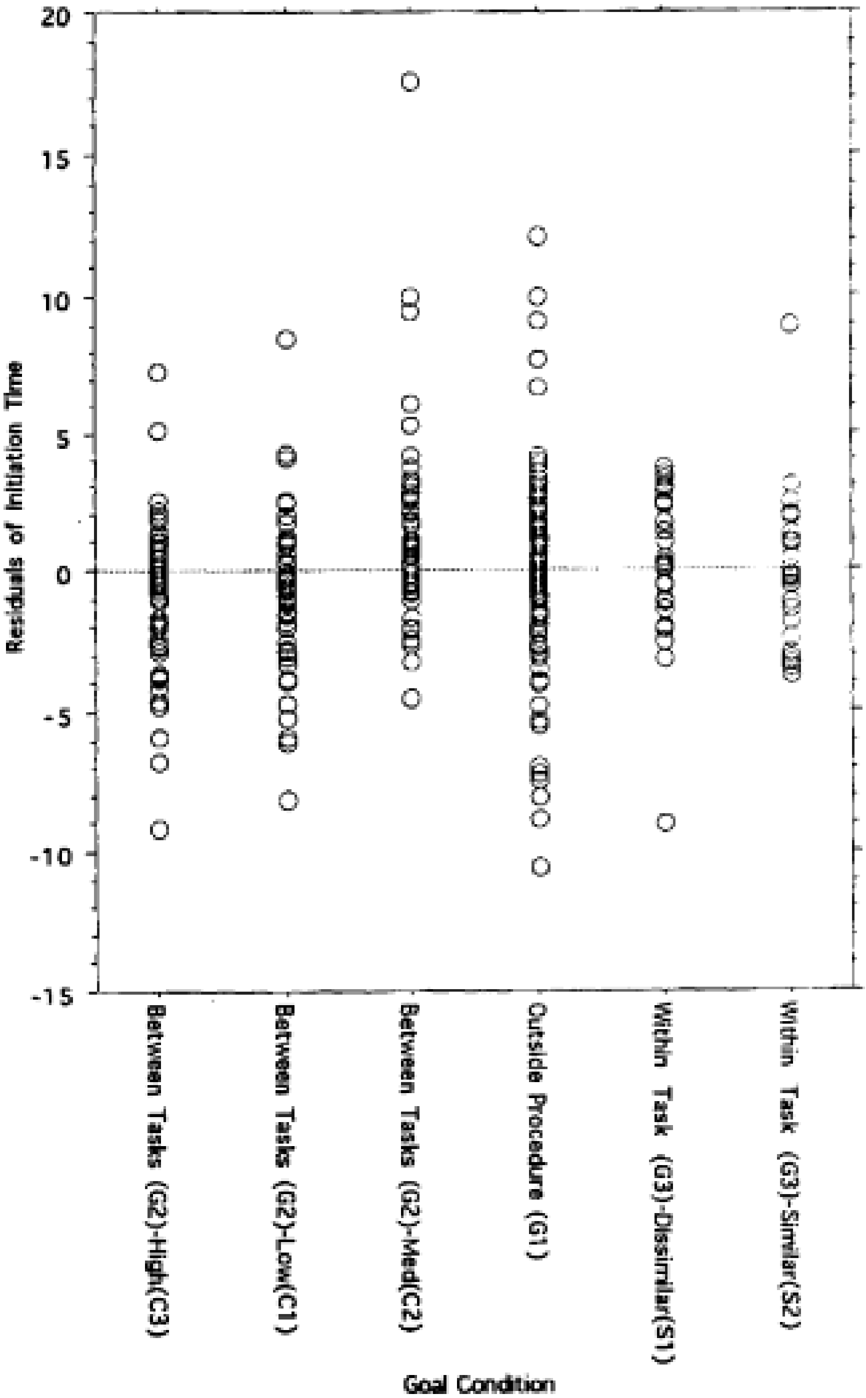
Scheffé Tests on Goal-Level Means.

		S	*p* -value
Outside Procedure	Between Tasks	0.762	0.3906
Between Tasks	Within Task	3.814	0.0001
Within Task	Outside Procedure	3.053	0.0012

Appendix 6.30

Effects of Goal-Level on Interruption Initiation Time (continued).

Plot of Initiation Time Residuals by Conditions in Goal-Level Analyses.



Appendix 6.31

Effects of Goal-Level on Interruption Performance Errors.

Analysis of Variance for Interruption Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	3.809	0.293	2.016	0.0202	Residual
Procedure Leg (PL)	1	0.035	0.035	0.174	0.6835	S * PL
Goal Level (GL)	2	0.051	0.026	0.133	0.8760	S * GL
S * PL	13	2.647	0.204	1.401	0.1591	Residual
S * GL	26	4.999	0.192	1.323	0.1421	Residual
PL * GL	2	0.634	0.317	1.942	0.1637	S * PL * GL
S * PL * GL	26	4.245	0.163	1.124	0.3145	Residual
Residual	242	35.167	0.145			

* Type II Sums of Squares

Interruption Performance Errors Means by Goal-Level.

	Count	Mean	Std.Dev.
Outside Procedure	105	0.162	0.463
Between Tasks	166	0.151	0.375
Within Task	55	0.127	0.336

Scheffé Tests on Goal-Level Means.

		S	*p* -value
Outside Procedure	Between Tasks	0.011	0.9789
Between Tasks	Within Task	0.023	0.9433
Within Task	Outside Procedure	0.035	0.8940

Appendix 6.32

Effects of Goal-Level on Procedure Resumption Time.

Analysis of Variance for Procedure Resumption Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	908.466	69.882	2.261	0.0089	Residual
Procedure Leg (PL)	1	244.211	244.211	4.002	0.0668	S * PL
Goal Level (GL)	2	22.856	11.428	0.365	0.6977	S * GL
S * PL	13	793.227	61.017	1.975	0.0251	Residual
S * GL	26	814.041	31.309	1.013	0.4529	Residual
PL * GL	2	5.776	2.888	0.146	0.8647	S * PL * GL
S * PL * GL	26	513.693	19.757	0.639	0.9109	Residual
Residual	180	5562.207	30.901			

* Type II Sums of Squares

Procedure Resumption Time Means by Goal-Level.

	Count	Mean	Std.Dev.
Outside Procedure	54	6.025	5.369
Between Tasks	157	6.864	6.279
Within Task	53	6.606	4.777

Scheffé Tests on Goal-Level Means.

		S	*p* -value
Outside Procedure	Between Tasks	0.839	0.6415
Between Tasks	Within Task	0.258	0.9589
Within Task	Outside Procedure	0.581	0.8664

Appendix 6.33

Effects of Goal-Level on Resumptive FPM Activity.

Analysis of Variance for Resumption FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	3.150	0.242	4.372	0.0001	Residual
Procedure Leg (PL)	1	0.832	0.832	11.871	0.0043	S * PL
Goal Level (GL)	2	0.290	0.145	2.326	0.1177	S * GL
S * PL	13	0.911	0.070	1.265	0.2387	Residual
S * GL	26	1.624	0.062	1.127	0.3162	Residual
PL * GL	2	0.325	0.163	2.843	0.0772	S * PL * GL
S * PL * GL	25	1.430	0.057	1.032	0.4287	Residual
Residual	169	9.365	0.055			

* Type II Sums of Squares

Resumption FPM Activity Means by Goal-Level

	Count	Mean	Std.Dev.
Outside Procedure	47	0.248	0.295
Between Tasks	156	0.171	0.239
Within Task	49	0.238	0.316

Scheffé Tests on Goal-Level Means.

		S	*p* -value
Outside Procedure	Between Tasks	0.077	0.2001
Between Tasks	Within Task	0.066	0.2849
Within Task	Outside Procedure	0.010	0.9791

Resumption FPM Activity Means by Procedure Leg * Goal-Level Interaction

Procedure Leg	Goal-Level	Count	Mean	Std.Dev.
18K’	Outside Procedure	23	0.140	0.157
	Between Tasks	80	0.137	0.209
	Within Task	23	0.151	0.213
FAF	Outside Procedure	24	0.352	0.357
	Between Tasks	76	0.208	0.264
	Within Task	26	0.315	0.373

Appendix 6.34

Effects of Goal-Level on Ensemble Performance Time.

Analysis of Variance for Ensemble Performance Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	25991.790	1999.368	10.593	0.0001	Residual
Procedure Leg (PL)	1	2830.444	2830.444	7.537	0.0167	S * PL
Goal Level (GL)	2	86.610	43.305	0.302	0.7417	S * GL
S * PL	13	4882.084	375.545	1.990	0.0225	Residual
S * GL	26	3725.318	143.281	0.759	0.7958	Residual
PL * GL	2	56.675	28.338	0.116	0.8907	S * PL * GL
S * PL * GL	25	6090.111	243.604	1.291	0.1677	Residual
Residual	229	43220.792	188.737			

* Type II Sums of Squares

Ensemble Performance Time Means by Goal-Level.

	Count	Mean	Std.Dev.
Outside Procedure	102	115.366	17.287
Between Tasks	158	116.350	17.099
Within Task	52	114.551	14.617

Scheffé Tests on Goal-Level Means.

		S	*p* -value
Outside Procedure	Between Tasks	0.984	0.8123
Between Tasks	Within Task	1.799	0.6475
Within Task	Outside Procedure	0.815	0.9235

Appendix 6.35

Effects of Goal-Level on Ensemble FPM Activity.

Analysis of Variance for Ensemble FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	2.939	0.226	29.433	0.0001	Residual
Procedure Leg (PL)	1	1.803	1.803	89.807	0.0001	S * PL
Goal Level (GL)	2	0.037	0.019	1.724	0.1981	S * GL
S * PL	13	0.261	0.020	2.615	0.0021	Residual
S * GL	26	0.279	0.011	1.398	0.1017	Residual
PL * GL	2	0.038	0.019	2.369	0.1143	S * PL * GL
S * PL * GL	25	0.199	0.008	1.038	0.4182	Residual
Residual	229	1.759	0.008			

* Type II Sums of Squares

Ensemble FPM Activity Means by Goal-Level

	Count	Mean	Std.Dev.
Outside Procedure	102	0.202	0.180
Between Tasks	158	0.176	0.136
Within Task	52	0.183	0.152

Appendix 6.36

Effects of Goal-Level on Procedure Performance Errors.

Analysis of Variance for Procedure Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	22.226	1.710	2.711	0.0013	Residual
Procedure Leg (PL)	1	3.857	3.857	3.223	0.0959	S * PL
Goal Level (GL)	2	1.622	0.811	0.981	0.3885	S * GL
S * PL	13	15.560	1.197	1.898	0.0307	Residual
S * GL	26	21.503	0.827	1.311	0.1488	Residual
PL * GL	2	1.741	0.871	1.538	0.2337	S * PL * GL
S * PL * GL	26	14.717	0.566	0.898	0.6122	Residual
Residual	252	158.917	0.631			

* Type II Sums of Squares

Procedure Performance Error Means by Goal-Level.

	Count	Mean	Std.Dev.
Outside Procedure	112	0.312	0.817
Between Tasks	168	0.405	0.863
Within Task	56	0.518	0.853

Appendix 6.37

Effect of Coupling-Strength on Interruption Acknowledgment Time.

Analysis of Variance for Interruption Acknowledgment Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	65.668	5.051	9.197	0.0001	Residual
Procedure Leg (PL)	1	0.368	0.368	1.871	0.1946	S * PL
Coupling-Strength (CS)	2	4.415	2.208	6.324	0.0058	S * CS
S * PL	13	2.560	0.197	0.359	0.9787	Residual
S * CS	26	9.076	0.349	0.636	0.9039	Residual
PL * CS	2	0.454	0.227	0.862	0.4340	S * PL * CS
S * PL * CS	26	6.850	0.263	0.480	0.9817	Residual
Residual	82	45.038	0.549			

* Type II Sums of Squares

Interruption Acknowledgment Time Means by Coupling-Strength.

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	55	7.160	0.799
Medium	(Physically-Coupled)	55	7.489	0.908
High	(Functionally-Coupled)	56	7.105	0.964

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	0.329	0.0249
Medium (Physically-Coupled)	High (Functionally-Coupled)	0.384	0.0079
High (Functionally-Coupled)	Low (Uncoupled)	0.055	0.8879

Appendix 6.38

Effect of Coupling-Strength on Interruption Initiation Time.

Analysis of Variance for Interruption Initiation Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	398.360	30.643	30.16	0.0012	Residual
Procedure Leg (PL)	1	34.852	34.852	2.057	0.1751	S * PL
Coupling-Strength (CS)	2	189.438	94.719	8.225	0.0017	S * CS
S * PL	13	220.227	16.941	1.667	0.0839	Residual
S * CS	26	299.429	11.517	1.134	0.3260	Residual
PL * CS	2	15.041	7.521	0.585	0.5643	S * PL * CS
S * PL * CS	26	334.315	12.858	1.266	0.2102	Residual
Residual	82	833.128	10.160			

* Type II Sums of Squares

Interruption Initiation Time Means by Coupling-Strength

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	55	3.773	3.174
Medium	(Physically-Coupled)	55	6.233	4.005
High	(Functionally-Coupled)	56	4.064	3.627

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	2.460	0.0032
Medium (Physically-Coupled)	High (Functionally-Coupled)	2.169	0.0090
High (Functionally-Coupled)	Low (Uncoupled)	0.291	0.9035

Appendix 6.39

Effect of Coupling-Strength on Procedure Resumption Times.

Analysis of Variance for Procedure Resumption Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	652.884	50.222	1.582	0.1103	Residual
Procedure Leg (PL)	1	134.340	134.340	2.874	0.1138	S * PL
Coupling-Strength (CS)	2	649.083	324.542	10.537	0.0004	S * CS
S * PL	13	607.677	46.744	1.472	0.1489	Residual
S * CS	26	800.814	30.801	0.970	0.5167	Residual
PL * CS	2	111.501	55.750	2.058	0.1480	S * PL * CS
S * PL * CS	26	704.420	27.093	0.853	0.6667	Residual
Residual	73	2317.692	31.749			

* Type II Sums of Squares

Procedure Resumption Time Means by Coupling-Strength

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	53	7.291	5.241
Medium	(Physically-Coupled)	49	9.360	8.559
High	(Functionally-Coupled)	55	4.230	3.069

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	20.69	0.1905
Medium (Physically-Coupled)	High (Functionally-Coupled)	5.130	0.0003
High (Functionally-Coupled)	Low (Uncoupled)	3.061	0.0282

Appendix 6.40

Effect of Coupling-Strength on Resumptive FPM Activity.

Analysis of Variance for Resumptive FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	1.409	0.108	2.071	0.0265	Residual
Procedure Leg (PL)	1	0.167	0.167	6.316	0.0259	S * PL
Coupling-Strength (CS)	2	0.313	0.156	2.822	0.0778	S * CS
S * PL	13	0.344	0.026	0.505	0.9144	Residual
S * CS	26	1.439	0.055	1.058	0.4113	Residual
PL * CS	2	0.149	0.074	1.702	0.2021	S * PL * CS
S * PL * CS	26	1.137	0.044	0.836	0.6889	Residual
Residual	72	3.768	0.052			

* Type II Sums of Squares

Resumptive FPM Activity Means by Coupling-Strength

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	52	0.236	0.271
Medium	(Physically-Coupled)	49	0.156	0.207
High	(Functionally-Coupled)	55	0.124	0.224

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	0.080	0.2502
Medium (Physically-Coupled)	High (Functionally-Coupled)	0.032	0.7877
High (Functionally-Coupled)	Low (Uncoupled)	0.112	0.0652

Appendix 6.41

Effect of Coupling-Strength on Interruption Performance Errors.

Analysis of Variance for Interruption Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	3.846	0.296	2.554	0.0053	Residual
Procedure Leg (PL)	1	0.137	0.137	0.740	0.4051	S * PL
Coupling-Strength (CS)	2	0.806	0.403	3.602	0.0416	S * CS
S * PL	13	2.408	0.185	1.599	0.1023	Residual
S * CS	26	2.910	0.112	0.966	0.5211	Residual
PL * CS	2	0.172	0.086	0.648	0.5312	S * PL * CS
S * PL * CS	26	3.453	0.133	1.146	0.3132	Residual
Residual	82	9.500	0.116			

* Type II Sums of Squares

Interruption Performance Error Means by Coupling-Strength.

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	55	0.182	0.389
Medium	(Physically-Coupled)	55	0.055	0.229
High	(Functionally-Coupled)	56	0.214	0.456

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	0.127	0.1569
Medium (Physically-Coupled)	High (Functionally-Coupled)	0.160	0.0589
High (Functionally-Coupled)	Low (Uncoupled)	0.032	0.8781

Appendix 6.42

Effect of Coupling-Strength on Procedure Performance Errors.

Analysis of Variance for Procedure Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	16.321	1.255	2.163	0.0238	Residual
Procedure Leg (PL)	1	2.579	2.579	4.987	0.0437	S * PL
Coupling-Strength (CS)	2	9.368	4.684	6.966	0.0038	S * CS
S * PL	13	6.721	0.517	0.891	0.5670	Residual
S * CS	26	17.482	0.672	1.159	0.3154	Residual
PL * CS	2	0.725	0.362	0.879	0.4273	S * PL * CS
S * PL * CS	26	10.725	0.412	0.711	0.8282	Residual
Residual	56	23.500	0.580			

* Type II Sums of Squares

Procedure Performance Error Means by Coupling-Strength.

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	56	0.679	1.130
Medium	(Physically-Coupled)	56	0.125	0.384
High	(Functionally-Coupled)	28	0.214	0.568

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	0.554	0.0056
Medium (Physically-Coupled)	High (Functionally-Coupled)	0.089	0.8957
High (Functionally-Coupled)	Low (Uncoupled)	0.464	0.0677

Appendix 6.43

Effect of Coupling-Strength on Ensemble Performance Times.

Analysis of Variance for Ensemble Performance Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	16521.282	1270.868	6.757	0.0001	Residual
Procedure Leg (PL)	1	972.468	972.468	3.793	0.0734	S * PL
Coupling-Strength (CS)	2	68.493	34.246	0.151	0.8608	S * CS
S * PL	13	3332.971	256.382	1.363	0.1980	Residual
S * CS	26	5904.572	227.099	1.207	0.2605	Residual
PL * CS	2	503.218	251.609	1.482	0.2458	S * PL * CS
S * PL * CS	26	4414.734	169.797	0.903	0.6030	Residual
Residual	74	13918.165	188.083			

* Type II Sums of Squares

Ensemble Performance Time Means by Coupling-Strength.

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	53	114.985	16.624
Medium	(Physically-Coupled)	53	117.243	17.453
High	(Functionally-Coupled)	52	116.831	17.456

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	2.257	0.7453
Medium (Physically-Coupled)	High (Functionally-Coupled)	0.412	0.9902
High (Functionally-Coupled)	Low (Uncoupled)	1.845	0.8226

Appendix 6.44

Effect of Coupling-Strength on Ensemble FPM Activity.

Analysis of Variance for Ensemble FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	1.271	0.098	17.427	0.0001	Residual
Procedure Leg (PL)	1	0.738	0.738	72.768	0.0001	S * PL
Coupling-Strength (CS)	2	0.003	0.001	0.244	0.7851	S * CS
S * PL	13	0.132	0.010	1.807	0.0576	Residual
S * CS	26	0.148	0.006	1.011	0.4656	Residual
PL * CS	2	0.011	0.005	1.133	0.3376	S * PL * CS
S * PL * CS	26	0.123	0.005	0.845	0.6775	Residual
Residual	74	0.415	0.006			

* Type II Sums of Squares

Ensemble FPM Activity Means by Coupling-Strength.

Coupling-Strength	(Coupling Type)	Count	Mean	Std.Dev.
Low	(Uncoupled)	53	0.172	0.144
Medium	(Physically-Coupled)	53	0.173	0.135
High	(Functionally-Coupled)	52	0.182	0.131

Scheffé Tests on Coupling-Strength Means.

		S	*p* -value
Low (Uncoupled)	Medium (Physically-Coupled)	0.002	0.9944
Medium (Physically-Coupled)	High (Functionally-Coupled)	0.009	0.8202
High (Functionally-Coupled)	Low (Uncoupled)	0.011	0.7639

Appendix 6.45

Effects of Similarity on Interruption Acknowledgment Times.

Analysis of Variance for Interruption Acknowledgment Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	87.612	6.739	10.409	0.0001	Residual
Procedure Leg (PL)	1	1.591	1.591	0.555	0.4694	S * PL
Similarity (SI)	1	0.003	0.003	0.003	0.9576	S * SI
S * PL	13	37.259	2.866	4.427	0.0073	Residual
S * SI	13	13.435	1.033	1.596	0.2130	Residual
PL * SI	1	1.262	1.262	1.949	0.1880	Residual
Residual	12	7.769	0.647			

* Type II Sums of Squares

Interruption Acknowledgment Time Means by Similarity.

	Count	Mean	Std.Dev.
Similar	28	7.584	1.519
Dissimilar	27	7.571	1.820

Appendix 6.46

Effects of Similarity on Interruption Initiation Times.

Analysis of Variance for Interruption Initiation Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	412.756	31.750	1.874	0.1429	Residual
Procedure Leg (PL)	1	3.650	3.650	0.406	0.5350	S * PL
Similarity (SI)	1	0.001	0.001	2.17E-4	0.9885	S * SI
S * PL	13	116.835	8.987	0.531	0.8645	Residual
S * SI	13	84.949	6.535	0.386	0.9492	Residual
PL * SI	1	79.732	79.732	4.707	0.0508	Residual
Residual	12	203.258	16.938			

* Type II Sums of Squares

Interruption Initiation Time Means by Similarity

	Count	Mean	Std.Dev.
Similar	28	8.518	4.134
Dissimilar	27	8.483	4.173

Interruption Initiation Time Means by Procedure Leg * Similarity Interaction

Procedure Leg		Count	Mean	Std.Dev.
18K’	Dissimilar	14	10.079	4.747
	Similar	14	7.484	2.841
FAF	Dissimilar	13	6.763	2.671
	Similar	14	9.552	5.011

Appendix 6.47

Effects of Similarity on Interruption Performance Errors.

Analysis of Variance Interruption Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	2.358	0.181	1.769	0.1661	Residual
Procedure Leg (PL)	1	0.017	0.017	0.184	0.6750	S * PL
Similarity (SI)	1	0.017	0.017	0.184	0.6753	S * SI
S * PL	13	1.231	0.095	0.923	0.5581	Residual
S * SI	13	1.234	0.095	0.925	0.5566	Residual
PL * SI	1	0.019	0.019	0.187	0.6727	Residual
Residual	12	1.231	0.103			

* Type II Sums of Squares

Interruption Performance Error Means by Similarity.

	Count	Mean	Std.Dev.
Similar	28	0.143	0.356
Dissimilar	27	0.111	0.320

Appendix 6.48

Effects of Similarity on Procedure Resumption Times.

Analysis of Variance for Procedure Resumption Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	498.634	38.356	2.060	0.1287	Residual
Procedure Leg (PL)	1	86.790	86.790	3.798	0.0732	S * PL
Similarity (SI)	1	4.955	4.955	0.806	0.3855	S * SI
S * PL	13	297.086	22.853	1.227	0.3788	Residual
S * SI	13	79.883	6.145	0.330	0.9677	Residual
PL * SI	1	22.494	22.494	1.208	0.2975	Residual
Residual	10	186.229	18.623			

* Type II Sums of Squares

Procedure Resumption Time Means by Similarity.

	Count	Mean	Std.Dev.
Similar	28	6.872	5.099
Dissimilar	25	6.309	4.474

Appendix 6.49

Effects of Similarity on Resumptive FPM Activity.

Analysis of Variance for Resumptive FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	2.022	0.156	3.149	0.0672	Residual
Procedure Leg (PL)	1	0.427	0.427	3.154	0.0854	S * PL
Similarity (SI)	1	0.031	0.031	0.602	0.4517	S * SI
S * PL	12	1.459	0.122	2.461	0.1194	Residual
S * SI	13	0.663	0.051	1.032	0.5081	Residual
PL * SI	1	0.029	0.029	0.592	0.4670	Residual
Residual	7	0.346	0.049			

* Type II Sums of Squares

Resumptive FPM Activity Means by Similarity.

	Count	Mean	Std.Dev.
Similar	27	0.261	0.345
Dissimilar	22	0.210	0.283

Appendix 6.50

Effects of Similarity on Ensemble Performance Times.

Analysis of Variance for Ensemble Performance Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	4660.392	358.492	2.653	0.0644	Residual
Procedure Leg (PL)	1	589.728	589.728	3.524	0.0850	S * PL
Similarity (SI)	1	0.305	0.305	0.002	0.9611	S * SI
S * PL	12	2007.871	167.323	1.238	0.3727	Residual
S * SI	13	1603.680	123.360	0.913	0.5701	Residual
PL * SI	1	142.200	142.200	1.052	0.3292	Residual
Residual	10	1351.465	135.146			

* Type II Sums of Squares

Ensemble Performance Time Means by Similarity.

	Count	Mean	Std.Dev.
Similar	26	114.617	13.924
Dissimilar	26	114.484	15.555

Appendix 6.51

Effects of Similarity on Ensemble FPM Activity.

Analysis of Variance for Ensemble FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	0.625	0.050	10.059	0.0004	Residual
Procedure Leg (PL)	1	0.235	0.235	21.572	0.0006	S * PL
Similarity (SI)	1	2.833E-4	2.833E-4	0.043	0.8390	S * SI
S * PL	12	0.131	0.011	2.185	0.1124	Residual
S * SI	13	0.086	0.007	1.323	0.3333	Residual
PL * SI	1	0.007	0.007	1.305	0.2799	Residual
Residual	10	0.050	0.005			

* Type II Sums of Squares

Ensemble FPM Activity Means by Similarity.

	Count	Mean	Std.Dev.
Similar	26	0.189	0.168
Dissimilar	26	0.177	0.138

Appendix 6.52

Effects of Similarity on Procedure Performance Errors.

Analysis of Variance for Procedure Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	16.732	1.287	3.646	0.0133	Residual
Procedure Leg (PL)	1	4.018	4.018	4.867	0.0460	S * PL
Similarity (SI)	1	0.161	0.161	0.582	0.4591	S * SI
S * PL	13	10.732	0.826	2.339	0.0693	Residual
S * SI	13	3.589	0.276	0.782	0.6679	Residual
PL * SI	1	0.161	0.161	0.455	0.5117	Residual
Residual	13	4.589	0.353			

* Type II Sums of Squares

Procedure Performance Error Means by Similarity.

	Count	Mean	Std.Dev.
Similar	28	0.571	0.836
Dissimilar	28	0.464	0.881

Appendix 6.53

Effect of Environmental Stress on Interruption Acknowledgment Times.

Analysis of Variance for Interruption Acknowledgment Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	16.326	1.256	2.581	0.0202	Residual
Replication (R)	1	0.740	0.740	1.521	0.2289	Residual
Procedural Leg (PL)	1	4.678	4.678	14.962	0.0019	S * PL
S * PL	13	4.064	0.313	0.642	0.7961	Residual
Residual	25	12.165	0.487			

* Type II Sums of Squares

Interruption Acknowledgment Time Means by Environmental Stress.

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	28	6.776	0.775
High	FAF	26	7.301	0.818

Appendix 6.54

Effects of Environmental Stress on Interruption Initiation Times.

Analysis of Variance for Interruption Initiation Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	150.493	11.576	0.898	0.5659	Residual
Replication (R)	1	3.783	3.783	0.294	0.5927	Residual
Procedural Leg (PL)	1	19.183	19.183	4.226	0.0605	S * PL
S * PL	13	59.003	4.539	0.352	0.9734	Residual
Residual	25	322.138	12.886			

* Type II Sums of Squares

Interruption Initiation Time Means by Environmental Stress.

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	28	4.850	2.403
High	FAF	26	6.112	3.893

Appendix 6.55

Effects of Environmental Stress on Resumptive FPM Activity.

Analysis of Variance for Resumptive FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	1.429	0.110	1.512	0.2048	Residual
Replication (R)	1	0.029	0.029	0.397	0.5368	Residual
Procedural Leg (PL)	1	0.580	0.580	10.788	0.0059	S * PL
S * PL	13	0.699	0.054	0.740	0.7059	Residual
Residual	18	1.308	0.073			

* Type II Sums of Squares

Resumptive FPM Activity Means by Environmental Stress

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	23	0.140	0.157
High	FAF	24	0.352	0.357

Appendix 6.56

Effects of Environmental Stress on Interruption Performance Errors.

Analysis of Variance for Interruption Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	1.184	0.091	0.779	0.6743	Residual
Replication (R)	1	0.083	0.083	0.709	0.4078	Residual
Procedural Leg (PL)	1	0.083	0.083	0.759	0.3993	S * PL
S * PL	13	1.419	0.109	0.934	0.5352	Residual
Residual	25	2.923	0.117			

* Type II Sums of Squares

Interruption Performance Error Means by Environmental Stress.

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	28	0.036	0.189
High	FAF	26	0.115	0.431

Appendix 6.57

Effects of Environmental Stress on Procedure Resumption Times.

Analysis of Variance for Procedure Resumption Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	536.081	41.237	1.595	0.1531	Residual
Replication (R)	1	41.683	41.683	1.612	0.2159	Residual
Procedural Leg (PL)	1	48.463	48.463	2.290	0.1541	S * PL
S * PL	13	275.063	21.159	0.818	0.6380	Residual
Residual	25	646.402	25.856			

* Type II Sums of Squares

Procedure Resumption Time Means by Environmental Stress.

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	28	5.230	6.073
High	FAF	26	6.882	4.451

Appendix 6.58

Effects of Environmental Stress on Ensemble FPM Activity.

Analysis of Variance for Ensemble FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	0.460	0.035	7.737	0.0001	Residual
Replication (R)	1	0.001	0.001	0.176	0.6788	Residual
Procedural Leg (PL)	1	0.239	0.239	41.156	0.0001	S * PL
S * PL	12	0.070	0.006	1.269	0.2978	Residual
Residual	24	0.110	0.005			

* Type II Sums of Squares

Ensemble FPM Activity Means by Environmental Stress

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	28	0.120	0.096
High	FAF	24	0.268	0.131

Appendix 6.59

Effects of Environmental Stress on Procedure Performance Errors.

Analysis of Variance for Procedure Performance Errors.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	15.089	1.161	1.230	0.3125	Residual
Replication (R)	1	3.018	3.018	3.198	0.0850	Residual
Procedural Leg (PL)	1	2.161	2.161	1.553	0.2347	S * PL
S * PL	13	18.089	1.391	1.474	0.1909	Residual
Residual	27	25.482	0.944			

* Type II Sums of Squares

Procedure Performance Error Means by Environmental Stress

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	28	0.357	0.780
High	FAF	28	0.750	1.295

Appendix 6.60

Effects of Environmental Stress on Ensemble Performance Times.

Analysis of Variance for Ensemble Performance Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Subject (S)	13	4513.478	347.191	1.234	0.3163	Residual
Replication (R)	1	36.591	36.591	0.130	0.7216	Residual
Procedural Leg (PL)	1	1146.613	1146.613	3.437	0.0885	S * PL
S * PL	12	4033.311	333.609	1.185	0.3467	Residual
Residual	24	6753.922	281.413			

* Type II Sums of Squares

Ensemble Performance Time Means by Environmental Stress.

Environmental Stress	Procedural Leg	Count	Mean	Std.Dev.
Low	18K’	28	120.307	2.349
High	FAF	24	111.068	22.008

Appendix 6.61

Effect of Interruption Conditions and Subjects on Resumptive FPM Activity.

Analysis of Variance for Resumptive FPM Activity.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value	Error Term
Experimental Conditions	15	3.098	0.207	4.041	0.0001	S * PL
Subject (S)	13	3.033	0.233	4.564	0.0001	Residual
Residual	280	14.312	0.051			

* Type II Sums of Squares

Resumptive FPM Activity Means by Experimental Conditions

Experimental Conditions	Count	Mean	Std.Dev.
11.05 , 21.05	26	0.053	0.113
11.06 , 21.06	17	0.064	0.124
11.08 , 21.08	5	0.000	0.000
11.09 , 21.09	9	0.135	0.330
12.02 , 22.02	23	0.140	0.157
12.03	13	0.125	0.219
22.03	10	0.185	0.212
12.05 , 22.05	27	0.190	0.216
12.06 , 22.06	28	0.132	0.232
12.07 , 22.07	25	0.084	0.163
13.02 , 23.02	24	0.352	0.357
13.03	12	0.231	0.340
23.03	14	0.386	0.398
13.05 , 23.05	25	0.286	0.318
13.06 , 23.06	27	0.116	0.219
13.07 , 23.07	24	0.231	0.224

Appendix 6.61 (continued)

Effect of Interruption Conditions and Subjects on Resumptive FPM Activity.

Resumptive FPM Activity Means by Subjects

Subject	Count	Mean	Std.Dev.
3	23	0.059	0.093
4	22	0.249	0.288
5	18	0.027	0.066
6	22	0.167	0.166
7	20	0.185	0.328
8	28	0.086	0.209
9	22	0.157	0.210
10	21	0.097	0.164
11	16	0.078	0.148
12	21	0.142	0.218
13	27	0.311	0.322
14	22	0.376	0.369
15	23	0.173	0.212
16	24	0.260	0.317

Appendix 6.62

Effect of Interruption Conditions and Subjects on Ensemble Performance Times.

Analysis of Variance for Ensemble Performance Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value
Experimental Conditions	17	7374.014	4337.766	23.305	0.0001
Subject	13	24424.652	1878.819	10.094	0.0001
Residual	382	71101.369	186.129		

* Type II Sums of Squares

Ensemble Performance Time Means by Experimental Conditions

Experimental Conditions	Count	Mean	Std.Dev.
11.05 , 21.05	25	91.043	15.476
11.06 , 21.06	24	81.908	9.158
11.08 , 21.08	28	82.439	11.768
11.09 , 21.09	24	87.945	9.081
12.02 , 22.02	28	120.307	12.428
12.03	13	115.677	12.914
22.03	14	121.243	15.708
12.05 , 22.05	27	118.628	13.027
12.06 , 22.06	27	116.498	15.331
12.07 , 22.07	27	121.264	17.211
12.10 , 22.10	25	117.569	13.130
13.02 , 23.02	24	111.068	22.008
13.03	12	106.599	11.505
23.03	13	113.558	15.319
13.05 , 23.05	26	111.203	19.212
13.06 , 23.06	25	117.190	19.815
13.07 , 23.07	26	113.066	17.027
13.10 , 23.10	25	111.754	19.536

Appendix 6.62 (continued)

Effect of Interruption Conditions and Subjects on Ensemble Performance Times.

Ensemble Performance Time Means by Subjects

Subject	Count	Mean	Std.Dev.
3	25	112.360	19.728
4	30	109.442	18.447
5	27	117.200	21.921
6	30	109.499	16.945
7	31	101.225	17.630
8	29	111.122	23.545
9	30	95.656	20.198
10	32	106.631	18.459
11	28	110.633	21.422
12	27	120.413	19.258
13	31	100.092	16.939
14	31	108.493	17.868
15	32	113.837	18.791
16	30	95.877	14.866

Appendix 6.63.

Effect of Interruption Conditions and Subjects on Procedure Resumption Times.

Analysis of Variance for Procedure Resumption Times.

Source	df	Sums of Squares*	Mean Square	*F*-value	*p*-value
Experimental Conditions	15	1584.469	105.631	3.163	0.0001
Subject	13	677.558	52.120	1.561	0.0954
Residual	302	10085.301	33.395		

* Type II Sums of Squares

Procedure Resumption Time Means by Experimental Conditions.

Experimental Conditions	Count	Mean	Std.Dev.
11.05 , 21.05	27	5.246	7.052
11.06 , 21.06	19	6.989	9.465
11.08 , 21.08	6	3.943	4.345
11.09 , 21.09	15	1.697	1.759
12.02 , 22.02	28	5.230	6.073
12.03	14	4.959	2.546
22.03	13	5.690	4.200
12.05 , 22.05	27	6.737	2.887
12.06 , 22.06	28	4.191	2.574
12.07 , 22.07	25	7.029	4.445
13.02 , 23.02	26	6.882	4.451
13.03	12	6.979	4.847
23.03	14	8.785	6.297
13.05 , 23.05	26	7.866	6.914
13.06 , 23.06	27	4.270	3.560
13.07 , 23.07	24	11.788	10.964

Appendix 6.63 (continued)

Effect of Interruption Conditions and Subjects on Procedure Resumption Times.

Procedure Resumption Time Means by Subjects.

Subjects	Count	Mean	Std.Dev.
3	25	7.575	5.871
4	22	5.982	6.386
5	21	4.496	3.484
6	23	6.873	4.093
7	23	6.517	11.062
8	28	6.850	7.479
9	22	6.020	2.889
10	24	5.163	5.783
11	22	3.660	3.740
12	21	5.247	3.193
13	27	9.684	8.485
14	26	5.300	4.784
15	23	7.078	3.982
16	24	6.353	6.383

References

Abbott, T.S. and Rogers, W.H. (1993). Functional categories for human-centered flight deck design. In *Proceedings of the 12th Digital Avionics Systems Conference*, 66-74.

Adams, M. J., Tenney, Y. J., and Pew, R. W. (1995). Situation awareness and the cognitive management of complex systems. *Human Factors*, *37* (1), 85-104.

Adams, M., Tenney, Y., and Pew, R. (1991). *State-of-the-Art-Report: Strategic Workload and the Cognitive Management of Advanced Multi-Task Systems*. Crew Systems Ergonomics Information Analysis Center, Wright Patterson Air Force Base, Ohio: BBN Report number 7650.

Agre, P.E. and Chapman, D. (1990). What are plans for? *Robotics and Autonomous Systems*, *6,* 17-34.

Allport, D.A. (1993). Attention and control: Have we been asking the wrong questions? A critical review of 25 years. In D.E. Meyer and S. Kornblum (Eds.), *Attention and Performance XIV: A Silver Jubilee* (pp. 183-218). Hillsdale, N.J.: Erlbaum Associates.

Anderson, J.R. (1983). A spreading activation theory of memory. *Journal of Verbal Learning and Verbal Behavior*, *22,* 261-295.

Anderson, J.R. (1983). *The Architecture of Cognition*. Cambridge, M.A.: Harvard University Press.

Bainbridge, L. (1984). Diagnostic skills in process operation. In *Proceedings of the 1984 International Converence on Occupational Ergonomics, 2*, 1-10.

Balota, D.A. and Rayner, K. (1991). Word recognition processes in foveal and parafoveal vision: The range of influence of lexical variables. In D. Besner & G.W. Humphreys (Eds.), *Basic Processes in Reading* (pp. 198-232). Hillsdale, N.J.: Lawrence Erlbaum.

Barnes, V.E. and Monan, W.P. (1990). Cockpit distractions: Precursors to emergencies. In *Proceedings of the Human Factors Society 34th Annual Meeting*, 1142-1144.

Bechtel, L.P. (1965). *The Effects of a Work Plan on the Resumption of an Interrupted Task.* Unpublished manuscript., Bates College, Lewiston.

Ben-Arieh, D. and Moodie, C.L. (1987). Knowledge-based routing and sequencing for discrete part production. *Journal of Manufacturing Systems,* 6, 287-297.

Bohnen, H.G.M. and Leermakers, M.A.M. (1991). Sampling behavior in a four-instrument monitoring task. *IEEE Transactions on Systems, Man, and Cybernetics*, *21*(4), 893-897.

Braune, R. and Wickens, C.D. (1986). Time-sharing revisited: Test of a componential model for the assessment of individual differences. *Ergonomics*, *29*(11), 1399-1414.

Brown, J. (1959). Some tests of the decay theory of immediate memory. *Quarterly Journal of Experimental Psychology, 10,* 12-21.

Cairns, H.S. and Cairns, C.E. (1976). *Psycholinguistics: A Cognitive View of Language*. New York: Holt, Rinehart, and Winston.

Cellier, J. and Eyrolle, H. (1992). Interference between switched tasks. *Ergonomics*, *35*(1), 25-36.

Chase, W.G. and Ericsson, A. (1981). Skilled memory. In S.A. Anderson (Ed.) *Cognitive Skills and their Acquisition*. Hillsdale, N.J.: Erlbaum Associates.

Chou, C.D. and Funk, K. (1993). Cockpit task management errors in a simulated flight operation. In *Proceedings of the 7th International Symposium on Aviation Psychology*, 965-969.

Chou, C.D. and Funk, K. (1990). Management of multiple tasks: Cockpit task management errors. In *IEEE International Conference on Systems, Man, and Cybernetics*, 1154-1156.

Cohen, S. (1980). After effects of stress on human performance and social behavior: A review of research and theory. *Psychological Bulletin, 88*, 82-108.

Cooper, R. and Franks, B. (1993). Interruptibility as a constraint on hybrid systems. *Minds and Machines*, 3, 73-96.

Cowan, N. (1993). Activation, attention, and short-term memory, *Memory and Cognition*, 21(2), 162-167.

Degani, A. and Wiener, E.L. (1990). *Human Factors of Flightdeck Checklists: The Normal Checklist* . NASA-CR-177549. Washington D.C.: National Aeronautics and Space Administration.

Detweiler, M. C., Hess, S. M., and Phelps, M. P. (1994). *Interruptions and Working Memory*. Unpublished technical report. University Park, PA: The Pennsylvania State University.

Diehl, J.M. (1975). *Human Factors Experiments for Data Link, Interim Report No. 6: An Evaluation of Data Link Input/Output Devices Using Airline Flight Simulators*, FAA-RD-75-160. Washington D.C.: Federal Aviation Administration.

Easterbrook, J.A. (1959). The effect of emotion on cue utilization and the organization of behavior. *Psychological Review, 66*, 183-201.

Elio, R. (1986) Representation of similar well-learned procedures. *Cognitive Science, 10,* 41-73.

Eriksen, C.W. and Schultz, D. (1978). Temporal factors in visual information processing. In J. Requin (Ed.) *Attention and Performance VII.* Hillsdale, N.J.: Erlbaum Associates.

Eysenck, M.I. (1982). *Attention and Arousal: Cognition and Performance*. New York: Springer-Verlag.

Field, G. (1987). Experimentus Interruptus. *SIGCHI Bulletin*, *19* (2), 42-46.

Folk, C.L., Remington, R.W., and Johnston, J.C. (1992). Involuntary covert orienting is contingent on attentional control settings. *Journal of Experimental Psychology: Human Perception and Performance*, *18,* 1030-1044.

Fox, E. (1994). Interference and negative priming from ignored distractors: The role of selection difficulty. *Perception & Psychophysics, 56* (5), 565-574.

French, S. (1982). *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-shop*. Chichester, England: Harwood.

Funk II, K. H. (1996). A functional model of flightdeck agenda management. In *Proceedings of the Human Factors and Ergonomics Society 40th Annual Meeting*. Santa Monica, CA: Human Factors and Ergonomics Society, 254-259.

Funk II, K. (1991). Cockpit task management: Preliminary definitions, normative theory, error taxonomy, and design recommendations. *The International Journal of Aviation Psychology, 1* (4), 271-285.

Funk II, K.H. and Linde, J. H. (1992) Agent-based Pilot-Vehicle Interfaces: Concept and Prototype. *Transactions on IEEE Systems, Man, and Cybernetics, 22*, 6, 1309-1322.

Georgeff, M.P. (1987). Planning. *Annual Review of Computer Science, 2,* 359-400.

Gillie, T. and Broadbent, D. (1989). What makes interruptions disruptive? A study of length, similarity, and complexity. *Psychological Research, 50,* 243-250.

Govindaraj, T., Poturalski, R.J., Vikmanis, M.M., and Ward, S.L. (1981). Human attention allocation strategies in situations with competing criteria - Development of an experimental paradigm and outlines for a model. *IFAC Control Science and Technology (Eighth Triennial World Congress),* pp. 3501-3506. Kyoto, Japan.

Griffon-Fouco, M. and Ghertman, F. (1984). Receueil de donnees sur les facteurs humains a Electricite de France. In *Operational Safety of Nuclear Power Plants* (pp. 157-172). Vienna: International Atomic Energy Agency.

Haider, S.W., Moodie, C.L., and Buck, J.R. (1981). An investigation of the advantages of using a man-computer interactive scheduling methodology for job shops. *International Journal of Production Research, 19,* 381-392.

Hamilton, V. and Warburton, D.M. (Eds.). (1979). *Human Stress and Cognition: An Information Processing Approach*. Chichester, England: Wiley Press.

Hammer, J.M. (1984) An intelligent flight-management aid for procedure execution. *IEEE Transactions on Systems, Man, and Cybernetics*, *14*(6), 885-888.

Hancock, P. (1991). On operator strategic behavior. In *Proceedings of the 6th International Symposium on Aviation Psychology*, 999-1007.

Hart, S.G. and Wickens, C.D. (1990). Workload assessment and prediction. In H. R. Booher (Ed.) *MANPRINT: An Emerging Technology. Advanced Concepts for Integrating People, Machines, and Organizations* (pp. 257-300). New York: Van Nostrand Reinhold.

Hayes-Roth, B. and Hayes-Roth, F. (1979). A cognitive model of planning. *Cognitive Science, 3,* 275-310.

Hess, S.M. and Detweiler, M.C. (1994). Training to reduce the disruptive effects of interruptions. In *Proceedings of the Human Factors and Ergonomics Society 38th Annual Meeting,* 1173-1177.

Hinton, D.A. and Lohr, G.W. (1988). *Simulation Investigation of Digital Data-Link ATC Communications in a Single-Pilot Operation.* NASA-TP-2837. Washington, D.C.: The National Aeronautics and Space Administration.

Hrebec, D.G., Infield, S., Rhodes, S., and Fiedler, F.E. (1994). A simulator study of the effect of information load and datalink on crew error. In *Proceedings of the Human Factors and Ergonomics Society 38th Annual Meeting*, 66-70.

Jenkins, C.D., Zyzanski, S.J., and Rosenman, M.D. (1979). *Jenkins Activity Survey*. New York: Psychological Corporation.

Johannsen, G. and Rouse, W.B. (1983). Studies of planning behavior of aircraft pilots in normal, abnormal, and emergency situations. *IEEE Transactions on Systems, Man, and Cybernetics, 13* (3), 267-278.

Jolly, E.J. and Reardon, R. (1985). Cognitive differentiation, automaticity, and interruptions of automatized behaviors. *Personality and Social Psychology Bulletin, 11* (3), 301-314.

Just, M.A. and Carpenter, P.A. (1992). A capacity theory of comprehension: Individual differences in working memory. *Psychological Review, 99*, 122-149.

Kahneman, D. (1973). *Attention and Effort.* Englewood Cliffs: Prentice Hall.

Kerns, K. (1990). *Data Link Communication Between Controllers and Pilots: A Review and Synthesis of the Simulation Literature*. MP-90W00027. McLean, V.A.: The MITRE Corporation.

Kirmeyer, S.L. (1988). Coping with competing demands: Interruption and the Type-A pattern. *Journal of Applied Psychology, 73* (4), 621-629.

Knox, C.E. and Scanlon, C.H. (1990). *NASA Transport Systems Research Vehicle B-737 Data Link System and Display Formats.* NASA-TM-102717. Washington, D.C.: The National Aeronautics and Space Administration.

Kramer, A.F., Sirevaag, E.J., Wickens, C.D., Reisweber, M., Strayer, D.L., and Grenell, J.F. (1991). Assessment of pilot performance and mental workload in rotary wing aircraft. In *Proceedings of the Workshop Psychophysiological Measures in Transport Operations (IB 316-91-01).* Cologne: Commission of the European Communities , Medical and Public Health Research Program. DLR - Institute for Aerospace Medicine.

Kreifeldt, J. G. and McCarthey, M. E. (1981). Interruption as a test of the user-computer interface. In *Proceedings of the 17th Annual Conference on Manual Control*. JPL Publication 81-95. Jet Propulsion Laboratory, California Institute of Technology.

Lewin, K. (1951). Intention, will and need. In D. Rapaport (Ed.) *Organization and Pathology of Thought*, pp. 95-153. New York: Columbia Universtiy Press.

Linde, C. and Goguen, J. (1987). *Checklist Interruption and Resumption: A Linguistic Study*. NASA-CR-177460. Washington, D.C.: The National Aeronautics and Space Administration.

Liu, Y. and Wickens, C.D. (1988). Patterns of task interference when human functions as a controller or a monitor. *IEEE International Conference on Systems, Man, and Cybernetics*, 864-867.

Loftus, G.R. and Mackworth, N.H. (1978). Cognitive determinants of fixation location during picture viewing. *Journal of Experimental Psychology: Human Perception and Performance*, *4*, 565-572.

Lorch, E.P. (1987). Perceived structure and the maintenance of attention. Bulletin of the *Psychonomic Society, 25*(1), 47-50.

Lysaght, R.J., Hill, S.G., Dick, A.O., Plamondon, B.D., Linton, P.M., Wierwille, W.W., Zaklad, A.L., Bittner, A.C., Jr., and Wherry, R.J. (1989). *Operator workload: Comprehensive Review and Evaluation of Operator Workload Methodologies*. ARI Technical Report 851. Fort Bliss, T.X.: United States Army Research Institute.

Madhaven, D. and Funk, K. (1993). Cockpit task management errors in critical in-flight incidents. *Seventh International Symposium on Aviation Psychology,* 970-974.

Massaro, D.W. and Cowan, N. (1993). Information processing models: Microscopes of the mind. *Annual Review of Psychology, 44*, 383-425.

McClelland, J.L. (1979). On the time-relations of mental processes: An examination of processes in cascade. *Psychological Review*, *86*, 287-330.

McDonald, S. and Stevenson, R.J. (1996). Disorientation in hypertext: The effects of three text structures on navigation performance. *Applied Ergonomics, 27*(1), 61-68.

Melton, A.W. (1963). Implications of short-term memory for a general theory of memory. *Journal of Verbal Learning and Verbal Behavior, 2*, 1-21.

Meyer, D.E. and Schvaneveldt, R.W. (1971). Facilitation in recognizing pairs of words: Evidence of a dependence between retrieval operations. Journal of *Experimental Psychology, 90*, 227-234.

Miller, J. (1989). The control of attention by abrupt visual onsets and offsets. *Perception & Psychophysics, 45,* 567-571.

Miller, G.A. (1956). The magical number even plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63*, 81-97.

Miller, G.A., Galanter, E., and Pribram, K.H. (1960). *Plans and the Structure of Behavior.* New York: Holt, Rinehart, and Winston.

Monan, W.P. (1979). *Distraction - A Human Factor in Air Carrier Hazard Events.* NASA-TM-78608. Washington, D.C.: National Aeronautics and Space Administration.

Moray, N. (1986). Monitoring behavior and supervisory control. In K.R. Boff, L. Kaufman, and J.P. Thomas (Eds.) *Handbook of Perception and Human Performance. Vol. II. Cognitive Processing and Performance*. New York: Wiley Press.

Moray, N. (1980). *Human Information Processing and Supervisory Control.* Technical Report. Cambridge, M.A.: Massachusetts Institute of Technology, Man-machine System Laboratory.

Moray, N., Dessouky, M.I., Kijowski, B.A., and Adapathya, R. (1991). Strategic behavior, workload, and performance in task scheduling. *Human Factors, 33*(6), 607-629.

Moray, N. and Hart, S. (1990). The human factors of strategic behavior: A panel discussion. In *Proceedings of the Human Factors Society 34th Annual Meeting,* 596-597.

Muller, H.J. and Rabbitt, P.M.A. (1989). Reflexive and voluntary orienting of visual attention: Time course of activation and resistance to interruption. *Journal of Experimental Psychology: Human Perception and Performance, 15*, 315-330.

Murphy, M., Randle, R., Tanner, T., Frankel, R., Goguen, J., and Linde, C. (1984). A full Mission simulator study of aircrew performance: The measurement of crew coordination factors and their relation to flight task performance. In E.J. Hartzell and S. Hart (Eds.), *Papers from the 20th Annual Conference on Manual Control,* NASA Ames Research Center.

National Transportation Safety Board. (1988). *Aircraft Accident Report - Northwest Airlines, McDonnell Douglas DC-9-82, N312RC, Detroit Metropolitan Wayne Co., Airport, Romulus MI, August 16, 1987*. Washington, DC: National Transportation Safety Board.

National Transportation Safety Board. (1973). *Aircraft Accident Report: Eastern Airlines Inc. Lockheed L-1011, N310EA, Miami International Airport, Miami, Florida, December 29, 1972.* Washington, D.C.: National Transportation Safety Board

Nissen, M.J.M. (1974). *Facilitation and Selection: Two Modes of Sensory Interaction*. Unpublished master's thesis, University of Oregon, Eugene, OR.

Ovsiankina, M. (1928). Die Wiederaugnahme unterbrochener Handlungen. *Psychologische Forschung, 11,* 302-379.

Panwalkar, S.S. and Iskander, W. (1977). A survey of scheduling rules. *Operations Research, 25,* 45-61.

Paquiot, J., Eyrolle, H., and Cellier, J. (1986). Consequences de l'interference entre taches de saisie sur la duree des traitements et l'occurrence des erreurs. In R. Patesson (Ed.), *L'homme et L'ecran* (pp. 199-209). Editions de l'Universite de Bruxelles.

Pattipati, K.R., Kleinman, D.L., and Ephrath, A.R. (1983). A dynamic decision model of human task selection performance. *IEEE Transactions on Systems, Man, and Cybernetics, 13*(3), 145-166.

Peterson, L.R. and Peterson, M.J. (1959). Short-term retention of individual verbal items. *Journal of Experimental Psychology, 58*, 193-198.

Plocher, T., Lockett, J.F., Kovach, P., and Powers, J. (1991). CREWCUT: A new tool for predicting human performance in conceptual systems. In *Proceedings of the Human Factors Society 35th Annual Meeting*.

Posner, M.I. (1980). Orienting of attention. *Quarterly Journal of Experimental Psychology, 78*, 391-408.

Posner, M.I., Nissen, J.M., and Klein, R. (1976). Visual dominance: An informtion processing account of its origins and significance. *Psychological Review, 83*, 157-171.

Posner, M.I., Snyder, C.R.R, and Davidson, B.J. (1980). Attention and the detection of signals. *Journal of Experimental Psychology: General 109*(2), 160-174.

Raby, M. and Wickens, C.D. (1990). Planning and scheduling in flight workload management. In *Proceedings of the Human Factors Society 34th Annual Meeting*, pp. 71-75.

Raby, M., Wickens, C.D., and Marsh, R. (1990). *Investigation of Factors Comprising a Model of Pilot Decision Making: Part 1. Cognitive Biases in Workload Management Strategy*. ARL-90-7/SCEEE-90-1, Aviation Research Laboratory, Institute of Aviation, University of Illinois at Urbana-Champaign.

Raby, M. and Wickens, C.D. (1991). Strategic behavior in flight workload management. In *Proceedings of the 6th International Symposium on Aviation Psychology*, pp. 1130-1135.

Rasmussen, J. (1986). *Information Processing and Human-Machine Interation: An Approach to Cognitive Engineering*. New York: North-Holland.

Remington, R.W., Johnston, J.C., and Yantis, S. (1992). Involutary attentional capture by abrupt onsets. *Perception & Psychophysics, 51*, 279-290.

Rollins, R.A. and Hendricks, R. (1980). Processing for words presented simultaneously to eye and ear. *Journal of Experimental Psychology: Human Perception and Performance*, 6, 99-109.

Rouse, W.B., Geddes, N.D., and Hammer, J.M. (1990) Computer-aided fighter pilots. *IEEE Spectrum, 27*, 3, 38-41.

Sadowski, R.P. and Medeiros, D.J. (1982). Planning and control for manufacturing systems and projects. In G. Salvendy (ed.) *Handbook of Industrial Engineering* (pp. 11.2-1 - 11.2-24). New York: Wiley Press.

Sanderson, P. (1989). The human planning and scheduling role in advanced manufacturing systems: An emerging human factors domain. *Human Factors,* 31, 635-666.

Sanford, A.J. and Garrod, S.C. (1981). *Understanding Written Language*. New York: Wiley Press.

Sarcerdoti, E.D. (1975). *A Structure for Plans and Behavior*. Technical Note 109, Menlo Park, C.A.: Stanford Research Institute.

Scholnick, E.K. and Friedman, S.L. (1993). Planning in context: Developmental and situational considerations. *International Journal of Behavioral Development, 16* (2), 145-167.

Schneider, W. and Shiffrin, R.M. (1977). Controlled and automatic human information processing: I. Detection, search, and attention. *Psychological Review, 84*, 1-66.

Schumacher, W. and Geiser, G. (1983). Human strategies in concurrent binary tasks under overload conditions. *Automatica, 19*(6), 723-727.

Segal, L. and Wickens, C.D. (1990). Taskillan II: Pilot strategies for workload management. In *Proceedings of the Human Factors Society 34th Annual Meeting*, 66-70.

Segal, L.D. and Wickens, C.D. (1991). *Strategic Management of Pilot Workload*. ARL-91-1/NASA-91-1, Aviation Research Laboratory, Institute of Aviation, University of Illinois at Urbana-Champaign.

Shankar, R. (1991). Z-Scheduler: Integrating theories on scheduling behavior into a computational model. In *IEEE International Conference on Systems, Man, and Cybernetics,* pp. 1367-1371.

Sheridan, T. (1981). Understanding human error and aiding human diagnostic behavior in nuclear power plants. In J. Rasmussen and W.B. Rouse (Eds.), *Human Detection and Diagnoses of System Failures*. New York: Plenum Press.

Sheridan, T.B. and Johannsen, G. (1976). *Monitoring Behavior and Supervisory Control*. New York: Plenum Press.

Shiffrin, R.M. and Schneider, W. (1977). Controlled and automatic human information processing 2: Perceptual learning, automatic attending and a general theory. *Psychological Review, 84,* 127-190.

Stanton, N.A., Booth, R.T., and Stammers, R.B. (1992). Alarms in human supervisory control: A human factors perspective. *International Journal of Computer Integrated Manufacturing, 5*(2), 81-93.

Tabe, T. and Salvendy, G. (1988). Toward a hybrid intelligent system for scheduling and rescheduling of FMS. *International Journal of Computer Integrated Manufacturing, 1*, 154-164.

Tabe, T., Yamanuro, S. and Salvendy, G. (1988). An approach to knowledge elicitation in scheduling FMS: Toward a hybrid intelligent system. In W. Karwowski, H.R. Parsaei, and M.R. Wilhelm (Eds.), *Ergonomics of Hybrid Automated Systems I* (pp. 259-266). Amsterdam: Elsevier.

Treisman, A. and Davies, A. (1973). Divided attention to eye and ear. In S. Kornblum (Ed.) *Attention and Performance IV*. New York: Academic Press.

Tulga, M.K. and Sheridan, T. (1980). Dynamic decisions and work load in multitask supervisory control. *IEEE Transactions on Systems, Man, and Cybernetics, 10*(5), 217-232.

Turner, J.W. and Huntley Jr., M.S. (1991). *The Use and Design of Flightcrew Checklists and Manuals*. FAOE2-A0180. Washington, D.C.: U.S. Department of Transportation.

Underwood, B.J. (1957). Interference and forgetting. *Psychological Review, 64*, 49-60.

Van Bergen, A. (1968). *Task Interruption*. Amsterdam: North-Holland.

van der Heijden, A.H.C. and Stebbins, S. (1990). The information-processing approach. *Psychological Research, 52*, 197-206.

Waller, M.C. and Lohr, G.W. (1989). *A Piloted Simulation Study of Data Link ATC Message Exchange*. NASA-TP-2859. Washington, DC: The National Aeronautics and Space Administration.

Wickens, C.D. (1984). *Engineering Psychology and Human Performance*. New York: Harper Collins.

Wickens, C.D., Larish, I., and Contorer, A. (1989). Predictive performance models and multiple task performance. In *Proceedings of the Human Factors 33rd Annual Meeting*, 96-100.

Wickens, C.D. and Liu, Y. (1988). Codes and modalities in multiple resources: A success and a qualification. *Human Factors, 30*, 599-616.

Williams, C. (1995). Potential conflicts of time sharing the flight management system control display unit with data link communications. *Eighth International Symposium on Aviation Psychology*.

Woods, D.D. (1995). The alarm problem and directed attention in dynamic fault management, *Ergonomics, 38* (11), 2371-2393.

Yantis, S. (1993). Stimulus-driven attentional capture and attentional control settings. *Journal of Experimental Psychology: Human Perception and Performance, 19*(3) 676-681.

Zeigarnick, B. (1927). Das Behalten erledigter and unerledigter Handlungen. *Psychologische Forschung, 9*, 1-85.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
October 1999

3. REPORT TYPE AND DATES COVERED
Technical Memorandum

4. TITLE AND SUBTITLE
Investigating Interruptions: Implications for Flightdeck Performance

5. FUNDING NUMBERS
WU 522-14-11-03

6. AUTHOR(S)
Kara A. Latorella

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Langley Research Center
Hampton, VA 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER
L-17910

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA/TM-1999-209707

11. SUPPLEMENTARY NOTES
Republication of Dissertation: State University of New York at Buffalo, 1996.
Dissertation Committee: Dr. Colin Drury, Dr. Valerie Shalin, Dr. Joseph Sharit.
Funded by NASA Graduate Student Researcher Program fellowship, NGT-50992.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified-Unlimited
Subject Category 54 Distribution: Standard
Availability: NASA CASI (301) 621-0390

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A fundamental aspect of multiple task management is attending to new stimuli and integrating associated task requirements into an ongoing task set; this is “interruption management” (IM). Anecdotal evidence and field studies indicate the frequency and consequences of interruptions, however experimental investigations of mechanisms influencing IM are scarce. Interruptions on commercial flightdecks are numerous, of various forms, and have been cited as contributing factors in many aviation incident and accident reports. This research grounds an experimental investigation of flightdeck interruptions in a proposed IM stage model. This model organizes basic research, identifies influencing mechanisms, and suggests appropriate dependent measures for IM. Fourteen airline pilots participated in a flightdeck simulation experiment to investigate the general effects of performing an interrupting task and interrupted procedure, and the effects of specific task factors: (1) modality; (2) embeddedness, or goal-level, of an interruption; (3) strength of association, or coupling-strength, between interrupted tasks; (4) semantic similarity; and (5) environmental stress. General effects of interruptions were extremely robust. All individual task factors significantly affected interruption management, except “similarity.” Results extend the Interruption Management model, and are interpreted for their implications for interrupted flightdeck performance and intervention strategies for mitigating their effects on the flightdeck.

14. SUBJECT TERMS
Pilot Performance, Interruption, Distraction, Multiple Task Management, Human Factors, Flight Deck Design, Procedures

15. NUMBER OF PAGES
309

16. PRICE CODE
A14

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z-39-18
298-102