Toward an expert system based menu interface evaluation tool

Stephen C. Gibbons

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

1992

25ème anniversaire

N° 1581

Programme 3
Intelligence artificielle, Systèmes cognitifs et
Interaction homme-machine

TOWARDS AN EXPERT SYSTEM BASED MENU INTERFACE EVALUATION TOOL

Stephen C. GIBBONS

Janvier 1992

RR-1581

Programme 3

Intelligence Artificielle, Systèmes Cognitifs et Interaction Homme-Machine

TOWARDS AN EXPERT SYSTEM BASED MENU INTERFACE EVALUATION TOOL

Stephen C. GIBBONS

Rapport de recherche INRIA

Towards an Expert System Based Menu Interface Evaluation Tool

ABSTRACT

This paper reports on a series of efforts conducted in the development of a functioning mock-up of an interface evaluation tool for menu systems. The paper offers a definition and organizational pattern for menu systems which divides the domain of menus into three areas : Menu System Structure, Menu System Objects, and Menu System Sequence Control aspects. This organizational pattern is then used to develop a highly specific, non-redundant, and integrated knowledge source which contains the existing design knowledge on menu systems. The mock-up is discussed in terms of its development stages (from flowcharting of the reasoning paths to the writing of user questions), its constituent parts (evaluation criteria, inputs, and outputs), and its functional aspects. The development of the mock-up has helped to examine the knowledge source for completeness, identify dependent design rules, and develop a reasoning path that can be used to create a prototype tool for the evaluation of user interfaces.

Keywords : Design Criteria, Evaluation, Knowledge Representation, Menu Dialogue, Human Factors Engineering.

Vers un outil d'évaluation des interfaces de type système expert

RÉSUMÉ

Cet article expose les étapes réalisées lors de la conception de la maquette d'un outil d'évaluation de menus d'interfaces. Une définition et une structure d'organisation sont également proposées. Cette organisation comporte une structure de menus, des objets de menus et des séquences de contrôle de menus.
Cette organisation est utilisée pour le développement d'une base de connaissances actuelles intégrées, hautement spécifiques et non redondantes sur la conception des menus.
Les étapes de développement (des organigrammes de stratégie de raisonnement à la rédaction des questions des utilisateurs) de cette maquette, ses parties constituantes (les critères d'évaluation, les entrées, les sorties) et ses aspects fonctionnels sont également présentés.
La réalisation de cette maquette a permis d'évaluer la complétude de la base de connaissances, d'identifier la dépendance de certaines règles ou recommandations de conception et de développer une stratégie de raisonnement qui peut être utilisée pour la création d'un prototype d'évaluation des interfaces utilisateurs.

Mots-clés : Critères de Conception, Evaluation, Représentation de Connaissances, Dialogue par Menu, Ergonomie.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

The following paper presents a series of efforts conducted under the general perspective of developing tools and methods for the evaluation of user interfaces. The goal of this paper is to investigate the feasibility of developing an evaluation tool which can effectively transfer context dependent human factors recommendations, in autonomous formats, that do not directly rely upon the human factors expert. This coincides with the far reaching aim of human factors to find ways to make results from basic research available to the diverse groups of individuals who can apply these results for the benefit of the human. It is not enough to simply identify and research areas of possible human factors contribution ; the next step must be taken -- that of applying what has been discovered and make the transition to the area of human engineering.

In making this transition there are many different paths that can be taken, and may be considered to lie on a continuum from high to low expert involvement. On the high side, the human factors specialist can become directly involved in the design, development, and testing of new products. He can personally interject his concerns and knowledge in a dynamic fashion within the normal system development process. While at the other end of the continuum are autonomous tools which perform the job of the human factors specialist when he is not available. Knowledge can be represented in the form of checklists, procedures to be followed, and tools which directly implement expert knowledge. This approach is of great value considering the numerous application areas where human factors knowledge could contribute and the limited number of trained human factors professionals. Somewhere in the middle of this continuum are the methods and tools designed for use by the human factors specialist to help them carry out their design and evaluation responsibilities.

In the field of human computer interaction the present human factors trend, which lies in the middle of the expert involvement continuum, has been to develop extensive cross referenced or indexed design guidelines (Brown, 1988 ; Rivlin, Lewis, and Cooper, 1990 ; Scapin, 1987 ; Smith and Mosier, 1986) , handbooks (Helander, 1988 ; Salvendy, 1987), and standards (ANSI/HFS, 1988 ; DIN, 1987) for use by designers, engineers and human factors professionals. The problem with the practical application of these knowledge sources is that users have difficulty locating relevant guidelines, choosing which guidelines should be used, establishing priorities among selected guidelines, and translating generally worded guidelines into specific design rules (Mosier and Smith, 1986). Thus, it can be said that this approach to knowledge transfer is not efficient, because it requires the presence of the human factors expert to select and apply the appropriate recommendation within the proper context.

Increasing the involvement of the human factors expert is counter to the original intention of design guidelines and recommendations for human computer interaction. Thus, a second solution is preferable -- taking this available body of information to the next lower step along the continuum.

Since the domain of human-computer interaction is wide and diverse, a first step requires the identification of an appropriate sub-domain. The recent interest in direct manipulation interfaces has produced many formal guidelines which address the design of menu systems (Brown, 1988 ; Rivlin, Lewis, and Cooper, 1990 ; Norman, 1991 ; Shneiderman, 1987 ; Smith and Mosier, 1986 ; Williges and Williges, 1984). Additionally, the International Standards Organization (ISO) is presently circulating a draft proposal of standards for the design of menu systems (Williams, 1990). The apparent abundance of guidelines and recent attempts at standardization lead to the belief that there exists a corresponding level of empirical evidence in support of them, and that the domain area is well established. Based on these observations the domain of menu dialogue design was selected to develop a prototype interface evaluation tool.

Contradictory to the existence of this large body of available literature is the fact that not much is really known about menu systems. Norman and Chin (1989) stated it nicely when they said "Although menu selection is widely used, its scope is currently limited, ill defined, and information lean." (p. 125). The problem stems from researchers who announce that they are investigating this part or that part of a menu system, often providing sufficient definitions of their examined portion, yet these researchers do not take the time to define or integrate their conception of a menu system as a whole. Even in Shneiderman's (1987) highly referenced book "Designing the User Interface", where an entire chapter is devoted to this topic, the presentation of a clear definition of menu systems is neglected. Many parts of the menu and its mode of utilization were nicely covered, yet the reader is still left with a vague feeling of what a menu is.

While many research studies exist, they often do not adequately address the critical issues within the domain. Past research is narrow and limited in scope, while the newer areas of investigation appear to be haphazard and incomplete. Past research mainly addresses three main areas (the depth vs. breadth trade-off, the sequencing of menu options, and the role of option designators), and generally provides inconclusive or mixed results. The continuing work on some of these older topics only succeeds in increasing the level of confusion by highlighting previous methodological problems, posing questions from slightly different perspectives, while presenting no conclusive results. The selection of topics for more recent research studies is not driven by previous research concerns in the domain, but rather follows a haphazard course defined by advances in functional capabilities of computing system. This wide variety of topics addressed by newer area of research makes it difficult to determine which of these deserve more attention. Paramount to consider, when reviewing the literature results, is that none of the research studies examined provide definite answers to the questions put forth. (For a more detailed review of the literature in the domain of menus, see Gibbons, 1991).

All this clearly points to the fact that while there appears to be a consensus on what a menu system does, on how it may be organized, and even agreement on some of its parts, nobody presents a clear definition of menu systems in all their complexity. Although the concept of menus appears intuitive to many individuals, intuition alone is not enough to base the development of a prototype evaluation system upon. Some form of specific, even if limiting, definition of the domain is required.

Thus, before work could be done specifying possible ways of designing an evaluation tool for menu system, an explicit definition and organizational pattern for menu systems and menu objects needed to be developed. The first section of this paper presents a complete definition for menu systems. The proffered definition divides the domain of menus into three areas : Menu System Structure, Menu System Objects, and Menu System Sequence Control aspects. Each of these three areas are expanded in the paper in support of the definition and function as an organizational framework for the menu domain.

Two other subsequent efforts, which rely highly on this organizational pattern, are also reported in this paper : a) the development of a knowledge source containing design rules for menu systems, and b) the development of a functioning mock-up of a proposed interface evaluation tool. The development of the knowledge source is traced from the identification of pertinent knowledge bases, through a series of modifications made to the set of identified design recommendations. The resulting knowledge source is the most highly specific, non-redundant, and integrated knowledge source available for the design and evaluation of menu systems. The development effort of a functioning mock-up is discussed in terms of its development stages (from flowcharting of the reasoning paths to the writing of user questions) and its constituent parts (evaluation criteria, inputs, and outputs). This is followed by an explanation of the way the mock-up functions and and how the evaluative outputs may be utilized. The conclusions of this report quickly review these three efforts and stress the value of automating the evaluation process.

2 Organizing the Domain of Menus

The first section of this report presents a global definition for menu dialogues and propose a domain structure in support of this definition. In formulating the definitions for the menu objects, existing definitions contained in the ISO draft standards document for the design of menu systems (Williams, 1990) were used and/or extended. Since ISO is preparing to release a design standard, it seemed reasonable to use their existing definitions when possible. Implicit in the presentation of this organizational pattern is the standardization of the terminology used when identifying menu system items. In addition to supporting a definition of menus, a

standardized language is presented for use in referring to specific menu objects, user actions, and menu structures.

The identification and organization of the knowledge into these three categories was conducted iteratively with a series of collected guidelines and recommendations. The identification of the three main classes is the result of initially noting static (structure and objects) and dynamic (sequence control) elements of menu systems. The identification of the various objects within these classes was accomplished by directly using the terms found in the design guidelines and recommendations. Individual design recommendations were examined to identify the object and the action item contained in the text. The objects and action items were compiled into a single list and reduced by eliminating redundancy. Each item within the list found its way into the formal structure as either a menu system object or as one of its attributes.

Menu Definition *:* Many authors of design guides and recommendations published over the past several years seem to agree with the general statement that -- a menu system is simply a presented list of selectable options which provide control functions to the user. While this definition in essence is correct it does not adequately specify menu systems. Thus, in the following section a step closer to standardizing the organization of the domain of menus is taken by presenting a definition for menu dialogues.

> *Menu dialogues are a human-computer interaction style, which provides information retrieval and command control functions to the user through the selection and execution of menu option selection targets. The menu dialogue is represented within the user environment by a menu system. The menu system is the aggregate sum of menu panel organization (structure), menu panel elements (objects), user and system actions (sequence control), and the interrelationship of these items in the presentation of a menu dialogue.*

From the definition given above it should be clear that a distinction between three main aspects within a menu system is being made : Menu System Structure, Menu System Objects, and Menu System Sequence Control. The Menu System Structure refers to the overall relationships present in the menu system between the user, computer system, menu objects and sequence control aspects. The Menu Objects are the identifiable parts of a menu system, and Sequence Control refers to the possible user and system actions that can be carried out on menu system objects. These three domain aspects will be further defined and their corresponding objects and attributes identified in the following sections. An outline of the organizational framwork for the domain of menus is presented in Table 1 (from Gibbons, 1991).

2.1 Menu System Structure

The first step in organizing the menu domain is to make the global distinction between menu systems as Command interfaces and as Information-Retrieval interfaces (Card, 1982 ; Giroux and Belleau, 1986). In command interface menus, the user must choose the command he wishes to execute. Due to the limited number of commands available and frequent use of the

Table 1 : Organizational framework for the domain of menus

Menu Domain Organization
(Organisation des Systèmes de Menu)

1.0 Menu System Structure :

1.1 *Type of Menu System*
- 1.1.1 Command Menu
- 1.1.2 Information Retrieval

1.2 *Class of Menu Structure*
- 1.2.1 Single Menu
- 1.2.2 Linear Sequence
- 1.2.3 Hierarchical (cyclic, acyclic)
- 1.2.4 Network

2.0 Menu System Objects :

2.1 *Menu Panel*
- 2.1.1 Titles
  - 2.1.1.1 Main Titles
  - 2.1.1.2 Group Label
  - 2.1.1.3 Menu Panel Desgnators
- 2.1.2 Menu Panel List
- 2.1.3 Menu Option Line
  - 2.1.3.1 Menu Option
  - 2.1.3.2 Option Designator
  - 2.1.3.3 Selection Area

2.2 *Optional Menu Panel Objects*
- 2.2.1 Command Line (Entry Area)
- 2.2.2 Cursor (position. shape)
- 2.2.3 Command Icon
- 2.2.4 Page Indicator
- 2.2.5 Menu Panel Messages
- 2.2.6 Menu Maps

3.0 Menu System Sequence Control :

3.1 *Control Methods*
- 3.1.1 Cursor Control
- 3.1.2 Point Control
- 3.1.3 Direct Control

3.2 *Menu Option Selection*
- 3.2.1 Access
  - 3.2.1.1 User Acess
  - 3.2.1.2 System Acess
- 3.2.2 Navigation
- 3.2.3 Search
- 3.2.4 Selection Indication
  - 3.2.4.1 Single Indication
  - 3.2.4.2 Multiple Indication
  - 3.2.4.3 Selection Correction

3.3 *Menu Option Execution*
- 3.3.1 Execution Indication
- 3.3.2 Execution and Selection
- 3.3.3 Execution Verification
  - 3.3.3.1 Conformation Action
  - 3.3.3.2 Rejection Action
- 3.3.4 Execution Edit Routine

3.4 *Menu Option Processing*
- 3.4.1 Messages
  - 3.4.1.1 Delay Notification
  - 3.4.1.2 Processing Status
  - 3.4.1.3 Interrupt Notification
- 3.4.1 Processing Interruptions
  - 3.4.1.1 User Interruption
  - 3.4.1.2 System Interruption
- 3.4.2 Processing Edit Routine

menus, the menu items are usually known and the task becomes a simple locating process. Whereas in the information-retrieval menu system the selection targets must be categorized under one of the menu items, these items being classes of information that can be found in the data base. The distinction between these two types of menu systems is gaining more support as more researchers accept this distinction (McDonald, Molander and Noel, 1988 ; Norman and Chin, 1988) and use it within their experimental methodologies (MacGregor, Lee and Lam, 1986).

Secondly, the presented structure only address menu systems organized into one of four different structures : Single Menu ; Linear Sequence ; Hierarchical Structure, cyclic or acyclic ; or Networked. With single menu structures all menu options are placed on one menu panel ; additional menu options are accessed through "scrolling" or "page turn" operations. Linear sequence menus present a series of independent menus in a predefined order ; users always see the same sequence of menus. Tree structured menus are hierarchically arranged

semantic groupings of menu options. This is the most common structure and is used when the number of menu options is large. Networked structures provide users access to disparate sections of a menu system and is utilized when the order of access can not be preestablished. For a good discussion of these structures see Shneiderman's 1987 book entitled "Designing the User Interface."

2.2 Menu System Objects

Menu System Objects are the actual portions of the menu system displayed to the user at a given point in time (it also pertains to the portion of a voice menu (sequence of options) presented to a user in a time segment). There are two basic classifications of menu system objects : Menu Panel, and Optional Menu Panel Objects. Critical to defining menu systems is the concept that these two objects represent. It must be clear that the menu panel refers to the display field area and graphical features which separate it from the other functional areas of the screen. This assumes that no menu system would ever superimpose portions of the menu system over other screen items without first removing these items. In cases where the menu panel and its objects are the only pieces of displayed information, the menu panel refers to the full screen. The optional menu panel objects are those objects within the menu system that may or may not be present within the menu panel. In other words, optional menu panel objects my appear in different locations across the display screen or within some portion of the menu panel.

Within the menu panel there exist three sub-classes of objects : Menu Panel Titles ; Menu Panel List ; and Menu Option Line. Menu panel titles come in three types, main titles, group labels, and menu panel designators. The menu panel list is the full set of presented menu option lines within the menu panel and is itself composed of individual menu option lines. The menu panel list presents one group of menu options or several related groupings of menu option lines. The menu option line contains the menu option, its designator, and the selection area. The menu option is the portion of the menu option line which identifies the information to be retrieved, or the control action to be taken. The option designator, although not always present, is a code or abbreviation used to uniquely designate each menu option on a menu panel for direct selection techniques. The selection area, used with cursor and point selection methods, is the active screen area surrounding the menu option line or a screen button used for selection indication.

The final class of menu panel objects, Optional Menu Panel Objects, refers to those objects that can be present within a menu system, yet may not always be displayed within the menu panel. A command line may be presented with direct control methods to echo user commands. In cursor control methods, an on-screen cursor will be provided in the form of an object (an arrow or an X), or as some distinguishable coding technique such as an inverse video or bright coded selection bar. If a menu list exceeds the length of the menu panel, a

"scroll" or "page turn" command icon will be present. In those systems which present multiple page menu lists (sequential menu panel lists considered as being one list of menu options) a page number indicator may be provided. At various times menu system messages will appear either within the menu panel or in an adjacent screen location. Graphical menu maps may also be presented within certain systems as navigational aids.

Attributes : Attributes are the inherent operational and presentation characteristics of menu objects. Presentation characteristics refer to format and graphical properties, while operational refers to the functional and performance aspects of menu objects. To illustrate the various types of attributes that can exist, those found within the object "Menu Option" are presented. The presentation of menu options can be textual, iconic, auditory, or any combination. The menu options can be positioned in a horizontal or vertical orientation or placed on the screen in random or geometric positions. They can perform command or information retrieval functions. They may be explicit, presented within a menu panel, or implicit, not presented but considered as available (selectable) menu options. They may be identified by various operational labels or classes : most frequently accessed, last accessed, position in list (first, second, last), critical option, appropriate option, unavailable option, scenario target, and explicit targets.

2.3 Menu System Sequence Control

In addition to the various menu system objects, a wide range of sequence control characteristics exist which can make similar menu system function quite diferently. There exist different methods of sequence control (Cursor Control, Point Control, Direct Control), various processing stages (Selection, Execution, and Processing), and types of system access. In order to address each of these areas and relate them to one another a definition proposed by Smith and Mosier (1986) is extended to make it more specific to menu systems.

> *Sequence control refers to the sequence of user actions and computer logic that is used to control the selection, execution and processing of menu options.*

Sequence Control Methods : Before individually discussing the different portions of sequence control (Selection, Execution, Processing), it needs to be made clear that there are different types of user control methods for performing these functions. All forms of sequence control by the user (user actions) are carried out using one, or a combination, of the following three control methods : Cursor Command, Point Command, or Direct Command.

Cursor command methods require the user to guide an on screen cursor to indicate selection, execution, and processing of menu options. Computer logic is employed in this case to match the cursor speed to that of the input devices' movement rates, control the position accuracy or point size of the cursor and the size of the selection area.

Point command methods require the use of a point-indication input device (light pens, fingers) in conjunction with touch screen interactions to indicate selection, execution, and processing of menu options. Computer logic is concerned with recognizing an on-screen touch as a user indication action and controlling the size of the selection area.

Direct command methods use keyboard input, or in the case of voice menus, a users auditory responses to input menu commands to indicate selection, execution, and processing of menu options. Direct selection does not require that a menu panel be displayed to the user. Computer logic is required to match user inputs to the appropriate menu operation.

Any one of these methods of sequence control, used individually or in combination with another, can be used to accomplish the users' sequence control objectives. Now that the different methods of sequence control have been described, the three major categories within sequence control will be presented : Selection, Execution, Processing.

Menu Option Selection: Menu option selection is the phase in sequence control where users indicate their selection of a desired selection target. The selection process in menu system sequence control consists of menu system : Access, Navigation, Search, and Selection Indication. Menu system access refers to the user or system actions which are executed to present the first menu panel on-screen. One special case of access exist when direct control methods are employed. In direct control methods there is no requirement that a menu panel be displayed when executing a menu option, and thus no reason to explicitly access the menu system.

Once the menu system is accessed, the user must navigate within the menu structure to locate the menu panel which contains the selection target he is seeking. Navigation refers to the paths followed in order to arrive at the specific menu panel which contains the selection target. Movement along the path can be conducted in one of two ways : 1) the selection, execution and processing of intermediate selection targets which present adjacent and linearly aligned menu panels ; 2) the use of navigational commands. Navigational commands are sequence control mechanisms that allow experienced users to directly access the menu panel which contain the menu option target. These include such functions as keyword access, function key movement, or other menu panel access techniques.

Search is the user activity conducted within a displayed menu panel to locate the appropriate selection target or intermediate selection target . The selection target is the desired menu option(s) within the menu system that the user is searching for. Intermediate selection targets are menu options that must be executed and processed prior to arriving at the menu panel which presents the selection target. Once the user locates a selection target within a menu panel the next step of selection indication is taken.

Selection indication is a user action conducted to identify which menu option(s) is to be executed. Within selection indication, single or multiple menu options can be selected and errors in selection corrected. Following the user's selection indication action the selection target is considered a selected menu option.

Multiple selection is the concurrent or sequential selection indication of multiple selection targets prior to a single execution indication action. Multiple selection indications are made from a single menu panel (concurrent) or across menu panels (sequential) through a command stack. The command stack is a series of keyed command inputs for selection indication of multiple selection targets. The command stack provide users with a means to speed up the menu dialogue by avoiding the need to display each menu panel. Selection correction actions allow users to correct selection indication errors prior to execution.

Menu Option Execution: The menu option execution phase in sequence control is where users indicate execution of the selected menu option(s), verify the execution indication action, and make necessary corrections. Execution indication is a user performed action which signals that the selected menu option(s) should be processed. Execution verification requests user conformation that the execution action taken was correct and that processing should be carried out on the selected menu option(s). This may be present on those occasions when the processing of the selected menu option(s) will cause destructive system consequences, produce irreversible results, or when processing may interfere with other concurrent task requirements.

In execution verification, users may confirm their execution indication action and continue to the processing phase of sequence control, or conduct a rejection action. A rejection action, performed to indicate a negative confirmation, returns the user to the menu option selection phase or to an execution edit routine. The execution edit routine is a selection correction routine embedded in the menu option execution phase of sequence control which allows users to modify (add, delete, change) the selected menu option(s) without the need to return to the option selection phase.

Selection indication and execution of a menu option may often be combined in a single input action (i.e., execution immediately follows any selection indication action). Two special cases of this are macro commands and type-ahead features. A macro command is a single pre-defined command input which provides execution of multiple section targets for frequent transaction sequences. Typeahead features combine selection and execution of menu options and are often based on option designator codes. Both of these methods provide the user with the means to speed up the menu dialogue by avoiding the need to display menu panels and the need to perform separate selection and execution indication actions.

Menu Option Processing The menu option processing phase in sequence control is where the menu system processes the selected menu option(s) following a final execution indication action. Within processing, there exist three possible system responses (delay notification, processing status, or interruption) and one user control response (interruption). Delay notification is a system message indicating that a processing constraint makes the immediate processing of the executed option impossible. Process status information is a form of feedback, in the form of an icon or a textual message, which tells the user that processing is being conducted, or lets the user know how much time remains before processing is complete.

Interruption, the halting of an ongoing processing sequence in order to redirect the course of the processing, can be user or system initiated. System interruption halts an on-going processing sequence because of an identified processing error or for the purpose of verifying the direction of the present processing sequence. User interruption halts an on-going processing sequence for the purpose of redirecting the transaction sequence or verifying the present direction. System feedback in the form of an interruption notification is required to indicate that processing interruption has occurred and that processing has indeed been halted.

There may be embedded in the menu option processing phase of sequence control a processing edit routine. This allows users to modify (add, delete, change) the executed menu option(s), after an interruption has occurred, without the need to return to the option selection or execution phase. This routine accomplishes the same goals as that of the execution edit routine and may in fact be the same routine.

3 Knowledge Source Development

This section presents the efforts conducted to develop a highly specific knowledge source for the design and evaluation of menu systems. During the course of a preliminary investigation (Gibbons, 1990) two knowledge bases, that contain much of the documented expertise in the domain of menus, were identified (i.e., Aschehoug, and Scapin, 1990 ; Williams, 1990). These two knowledge bases contain the original set of design rules and guidelines that have been modified for use in the development of the interface evaluation tool proposed in section four of this report.

The Aschehoug and Scapin (1990) report contains 831 design recommendations which span a wide range of interface topics from general screen design to specific recommendation for the presentation of windows. The knowledge base was developed by modifying a series of design recommendations found in design standards and design guides. The natural language of each identified recommendation was modified to clearly state its premiss and conclusion (Scapin, 1990a). Additional information was included in the report for each recommendation

about its original source, references in support or disagreement, and some clarifying examples when necessary.

The Williams (1990) report is a portion of a design standard that is being circulated by the International Standards Organization (ISO) for the design of user interfaces. This emerging standards document provides 136 guidelines for design of menus used in user-computer dialogues to accomplish typical office tasks. Each section of the document specifies the ergonomic design goal, and each guideline lists the dialogue principles which pertain to that guideline. To assist the designer in making tradeoff decisions dialogue principles are provided in addition to information concerning the ergonomic rationale for each guideline. Many of the guidelines provide an example of an implementation that embodies that guideline. Original reference sources are provided, as well as a means to evaluate the menu systems conformance to any given guideline.

Together these two documents make up the preliminary knowledge source of 967 design recommendations identified for use in the development of our interface evaluation tool. Because these two knowledge bases were developed independently of each other, and contain redundancies and extra information not associated with menu systems, steps needed to be taken to streamline these knowledge bases. In the following section of this report steps taken to reduce, rewrite, and classify these two sets of design recommendations will be presented.

3.1 Streamlining the Knowledge Source

The first step taken to streamline the preliminary knowledge source was to eliminate those recommendations in the Aschehoug and Scapin (1990) report that did not apply to identifiable aspects of menu systems. A separate effort, independent from consideration of the ISO knowledge base, was conducted to accomplish this task (Gibbons, 1990). The Aschehoug and Scapin (1990) knowledge base was reduced from the original 831 design recommendations down to 526 by identifying recommendations which corresponded to the categories of Dialogue, Command Language, Menus, and Individual Screens. Design recommendations within these categories were subsequently examined to identify recommendations that could not be applied to a menu evaluation system. Recommendations were excluded if they did not have a connection with menu system. This further reduced the number of the recommendations down to approximately 350. A final reduction was conducted by eliminating those recommendations determined to be too general, ambiguous, or not generalizable to a menu system evaluation framework. Contradictory and redundant recommendations were also examined and one selected as the standard. This reduction left the Aschehoug and Scapin (1990) knowledge base with 261 design recommendations that either applied directly to menu dialogues or inferred secondary properties of menus. For a more detailed explanation of how this process was conducted see Gibbons (1990).

The second step in streamlining the preliminary knowledge source involved integrating the original knowledge base, followed by the removal of redundant, and inconsistent design rules. Because these two knowledge bases were developed independently, yet relied on the same source documents, many of the design recommendations were similar. Additionally, the quality of the design recommendations were not consistent, some of the recommendations were either to general in nature or worded in such a way that the intended meaning was lost. Removal of these rules (32 redundant, and 46 unclear), from the combined set of 397 design recommendations, resulted in the evolving knowledge source being reduced to 319 design recommendations deemed suitable for our application.

The result of these two streamlining steps left a combined knowledge source containing 319 design recommendations for menu systems. At this point a rather large percentage of the information on menu systems, available in the literature, had been compiled into one source document. The next step in preparing the knowledge source for use in a computer based application was to : a) rewrite the design recommendations using the standardized vocabulary determined in the organization of the domain elements (see section 2); and b) categorize the design recommendations according to an evaluation scheme. Each of these steps will now be presented : Rewriting, and Categorization.

3.2 Rewriting the Design Recommendations

Since the design recommendations were collected in their original format, many inconsistency in the use of terms were found. For example : coded entry = keyed codes = option designators = selection number = sequential number coding ; option phrasing = option wording ; option order = sequencing = numbering schemes ; selected item = selected option ; command line = command area. While these examples are not extreme they do demonstrate the mixed terminology found in the domain. Secondly, the design recommendations themselves were often written in too general of a fashion. A single recommendation often referred to several different menu objects, or contained multiple premisses and conclusion statements.

In order to alleviate these problems, and to standardize the terminology that would be used, all of the recommendations within the knowledge source were rewritten into design rules. Each design recommendation was individually examined to determine which menu object, attribute, or sequence control aspect was being addressed. Concurrent with this analysis recommendations were examined to determine if they would be better stated by multiple design rules rather than as one. In this fashion the design rules were written such that they contained only one premiss statement, one conclusion statement, and referred to only one menu object or one of its attributes.

In the rewriting of the design recommendations the concept of rule independence was highly stressed. Achieving this was done by using clear natural language terms in the premiss

and conclusion statements of the design rules, and avoiding the insertion of procedural aspects. "In their purest form, rules are independent chunks of knowledge with no necessarily implied serial chain among them." (Pederson, 1989). By structuring independent design rules the transparency of the proposed application will be improved, development time will be shortened, and the final knowledge source will be easier to maintain.

The rewriting of the preliminary knowledge source resulted in the addition of 118 design rules. These additional design rules are those added after design recommendation identified as having multiple premiss or conclusion statements were rewritten. During the rewriting of the design recommendations, redundancies were easier to locate, resulting in 26 of the design recommendations being eliminated from the knowledge source.

Following these steps the knowledge source contained 397 highly specific design rules related to menu systems. This list of design rules presents the most clearly worded, redundant free, knowledge source available concerning menu systems. Each design rule in the knowledge source is an independent chunk of knowledge that can be applied to menu systems. The design rules are written in a production rule like format that reflects the organization pattern of menu systems previously determined. It is with this knowledge source that the next phase of development has taken place -- classification of individual design rules.

3.3 Classification

Now that a streamlined and highly independent set of design rules exist in a single knowledge source the next step was to classify these design rules in such a way that development of an interface evaluation tool, which relies on this knowledge, would be facilitated. The set of design rules were classified according to three items : Abstract Level, Evaluation Criteria, and Menu Object, according to Scapin (1990a). Each rule was once again reexamined and assigned a value for each of these three categories.

The design rules were first classified according to four different levels of abstraction, or points of view, based on distinctions made by Foley and Van Dam (1984): Conceptual, Semantic, Syntactic, and Lexical. The conceptual level was assigned to design rules that define key application concepts of the menu system. These design rules can be considered as referring to the user model of the application. These rules define the existence of object, the relationships between objects, and the operations performed on these objects. The semantic level was assigned to design rules that addressed the meaning placed on objects and on their operations. The syntactic level label was assigned to design rules that define sequence of inputs and outputs. For inputs it refers to the design rules that pertain to the sequence grammar of how and in what order user actions are taken. For output this refers to the spatial and temporal factors of objects. The lexical level label was assigned to rules that address the actual primitive shapes (lines, characters), and their attributes (such as color and font) displayed in a menu systems.

Table 2. Example of design rules for menu appropriateness

Menu Appropriateness : Design Rules
83) Menu dialogues can be selected if initial entry accuracy is critical. (appropriate - conceptual - error protection)
84) Menu dialogues should be selected if some user will not be familiar with all functions of the software system. (appropriate - conceptual - prompting)
85) Menu dialogues can be selected if tasks are routine, with fixed procedures that require only minimal entry. (appropriate - conceptual - minimal actions)
86) Menu dialogues should be selected if the future users will be inexperienced. (appropriate - conceptual - compatibility)

Secondly, the design rules were classified according to a set of design criteria identified by Scapin (1990b). He identified 8 main criteria , and 10 sub-criteria dimensions (for a total of 18 elementary criteria), which lead to a better, more efficient, less-error prone behavior of interface user. The 8 main criteria are as follows : Compatibility, Consistency, User Workload, Adaptability, Explicit control, Significance of codes, Guidance, Error management. These criteria were identified by using existing definitions in the literature (Scapin, 1987 ; Smith and Mosier, 1986 ; Williges and Williges, 1984). The process of identifying criteria was conducted in an iterative fashion using an empirical approach based on the progressive classification of recommendations. Pollier (1990a) has found that these design criteria account for 83% of the problems identified during the evaluation of an interface. Additional validation efforts are presently being conducted upon these criteria (Bastien, 1991, Bastien and Scapin, 1991). A full listing and definition of the various criteria, and some examples, can be found in Scapin (1990b).

The third classification of the design rules is based on the menu system structure presented in section 2 of this report. During the development of the knowledge source each design rule had been rewritten to reflect this organizational pattern. Thus, it was just a matter of identifying which menu object the design rules were addressing. The design rules were then given a label which reflected the menu object it referred to. For a partial listing of the labeled and organized knowledge source see appendix A : Knowledge Source. Table 2 presents a small subset of the labled design rules for the category "Menu Appropriateness".

Up to this point the objective of the project has been stated-- designing an evaluation tool ; a definition of menu systems which includes its structure, objects, and sequence control capabilities has been provided ; and a highly specific knowledge source of human factors

design rules has been developed. Armed with this information the next step in the investigation will be taken - that of proposing a working model of a prototype evaluation system. The remaining sections of this paper will change gears from speaking of issues in the domain of menus and begin to address the development of a functioning mock-up. The mock-up will be discussed in terms of its unique development goals, the construction of flowcharts used to map out a reasoning path, and of its constituent parts.

4 Mock-Up Development

After completing the knowledge source development effort, the next step has been to examine the feasibility of a computer based application of this knowledge. To test the feasibility of a computer based system and to examine the knowledge source for missing information a paper walk-through, or as it is often called a mock-up version of the proposed system, was developed. This intermediary step was deemed necessary because of the large number (397) of independent rules identified during the development of the knowledge source.

The goal of the mock-up has been to examine our knowledge source for completeness, identify dependent design rules, and develop a reasoning path that the proposed system could follow. The development of the mock-up required a two phase effort. The first phase was the determination of a reasoning path. This was accomplished by constructing detailed flowcharts of the relationship between individual design rules in the knowledge source. The second phase composed the actual writing of user questions, which will be asked of an evaluator, in a sequence order dictated by the paths identified in the flowcharts.

The remainder of this report presents these two development phases and discusses the mock-up in terms of the evaluators input, the sequence order of questioning, and concludes by providing a detailed discussion of the output of the proposed tool.

4.1 Flowcharting of the Reasoning Paths

The first step taken in the development of the mock-up was to flowchart the relationships between groups of related design rules. Up until this point the design rules have been organized into groups of design rules for individual menu objects (i.e., according to the identified menu structure). Further decomposition of the design rules into subgroups which address similar attributes within a menu object had yet to occur. Thus, in order to organize the knowledge source into smaller subgroups of related design rules, and to link these subgroups together in a logical structure, flowcharts of the relationships between design rules within menu objects were constructed.

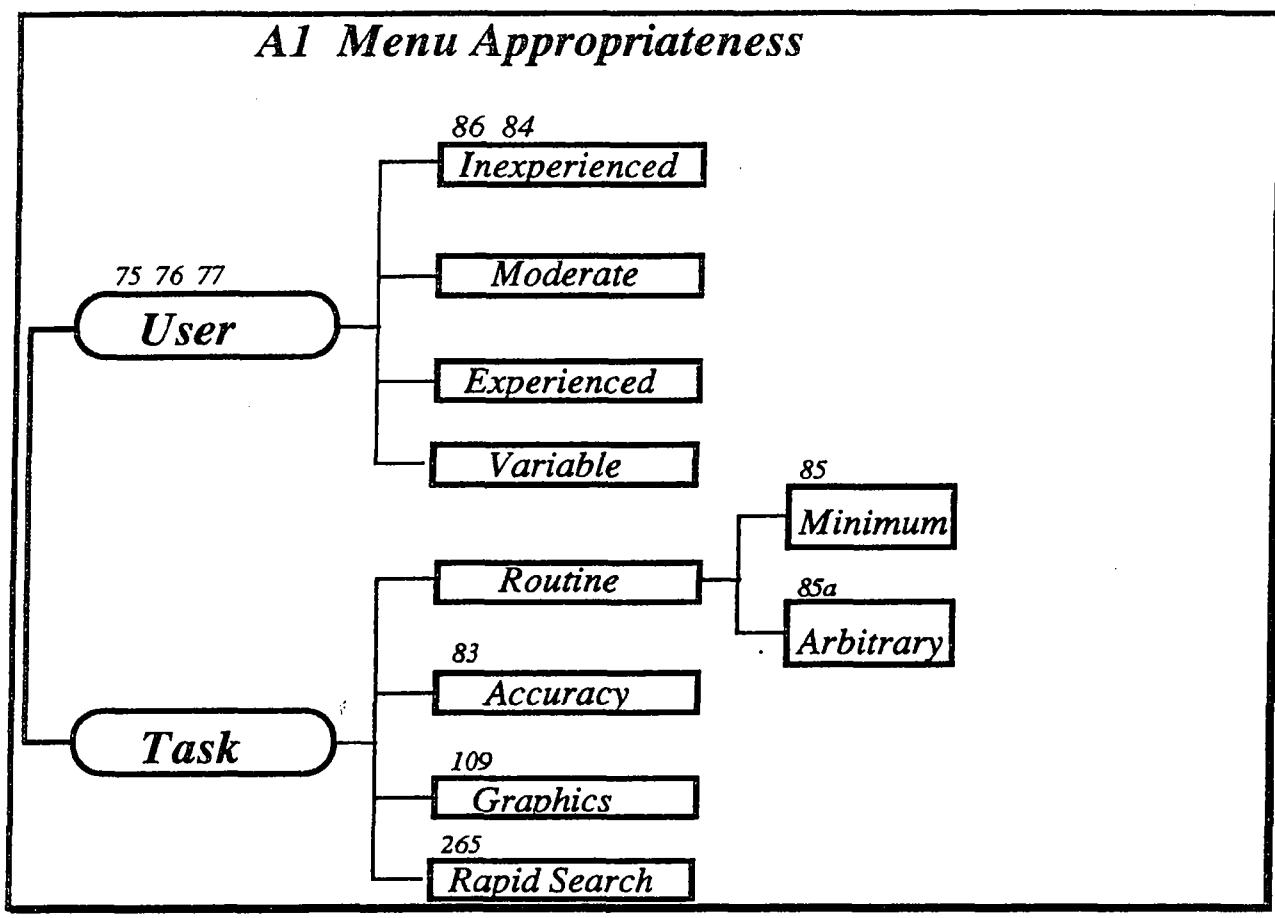
Initially, the knowledge source was organized such that all of the design rules addressing a given menu object were placed together for easy reference (see section 3.3). The development of the flowcharts was accomplished by extending this organization into even smaller subgroups of design rules. Design rules, within a single menu object, were examined to identify major attributes labels that could be used to classify subgroups. For example, within the object "Menu Appropriateness" all the design rules were identified as addressing either task or user aspects. Thus, design rules pertaining to this menu object were decomposed into two subgroups of design rules and given the attribute labels of "task" and "user". Each of these subgroups were subsequently examined to determine if any other logical second level sub-groupings within this attribute were needed. This process of dividing and examining was carried out until all the design rules within a menu object were hierarchical related.

Once these relationships were identified, flowcharts were constructed using the attribute labels. Identification number for each of the design rules, pertaining to these attribute labels, were also included as part of the flowcharts. In this fashion every design rule in the knowledge source has been assigned to an identified node within the flowcharts. Nodes are the identified subgroups of design rules within a given menu object. Nodes are represented in the flowcharts by rectangular boxes. Thus, in addition to providing attribute labels for the design rules, the assignment of identification numbers to the appropriate subgroup makes it easier to identify the corresponding design rules for each node. These flowcharts are presented in appendix B. An example of a flowchart for the category of "Menu Appriateness" is presented in figure 1.

The final step in the development of the flowcharts was to reorganize the knowledge source according to the attribute labels assigned to the subgroups. The design rules within each menu object were reordered to reflect the same left-to-right, top-to-bottom organization found in the flowcharts. The first and second level attribute labels assigned to nodes (i.e., subgroups) were added to the knowledge source to facilitate the location of corresponding design rules within the menu object. In this manner the ordering of the design rules in the knowledge source (appendix A) correspond exactly to the sequence order and attribute labels presented in the flowchart of menu objects (appendix B).

The result of this flowcharting effort is a highly detailed organization, and graphical representation of the knowledge source. All of design rules for each menu object are now classified into related subgroups, with hierarchical inheritance explicitly identified. The reorganization of the knowledge source makes it easier to locate design rules and identify which of the subgroups lack sufficient information. This will also make it easier when it becomes necessary to up-date the knowledge source.

Figure 1. Flowchart of the menu appropriateness category



4.2 Question Sequence

The second step in the development of the mock-up has been to rewrite the design rules into a series of user questions whose responses could be used to determine adherence or contradiction, of the menu system being evaluated, to the design criteria identified in Scapin (1990). There exists several reasons why the design rules need to be rewritten into a question-response format. The most important reason is that the ability to more accurately estimate the number of questions that would need when utilizing the proposed evaluation tool was needed. The writing of potential questions, prior to developing a prototype, would also help identify those design rules whose instantiated conclusion statement could be used to determine the premiss statement of a dependent rule. Identifying these types of dependency would mean that responses to one user question would help determine the conclusion for multiple design rules (i.e., limiting the number of questions needing to be asked). Presently the knowledge source contains 397 independent rules that must have their premiss statement determined. If every design rule must be addressed by a user question (i.e., there exists no means to indirectly instantiate design rules) then the size of the proposed system would be too clumsy and unwieldy to be efficiently utilized by an evaluator.

The second major reason for writing the design rules into user questions is that the relationships between menu objects could be examined and further specified. Although the flowcharts identify the logical flow of design rule instantiation within a menu object, they don't adequately express the existence of relationships between menu objects. This type of design rule dependency will be easier to identify by actually asking potential evaluative questions. Design rules which address one menu object may need information about other menu objects to help instantiate the premiss statement of a design rule. For example rules number 204 (The selection area, with cursor control methods, should be at least 10 times the size of the positioning accuracy of the cursor or 0,6 cm (1/4 inch)- square, whichever is smaller.) and 326 (The selection area for point control methods can be at least the same size as the menu option line, plus a half character distance around the label if the finger is used as the point device.) require the knowledge of which type of sequence control method is utilized. The identification of these dependencies will help later development by explicitly outlining the relationships between menu objects.

Lastly, the asking of potential evaluative questions brings into focus the sequencing of the question "Which question to ask 1st, 2nd, or last", and provides an impetus for explicitly determining a reasoning path. Rather than just rewriting the design rules in the form of user questions, to determine what the questions may look like and how many of them there may be, the development of a mock-up gives a functional aspect to the static design rules. Design rules that stand independently must function in a dynamic system. The branching within the sequence of question, to access or ignore a series of related questions, can be visualized as a modeling of the reasoning paths that might be followed in a computer based tool. This can be more readily visualized by reexamining the example of rules number 326 and 204 given above. In this example the question associated with rule number 326 would only be asked if one of the sequence control methods available to the user was point control. If the user of the menu system is not provided with point control methods of sequence control then the question associated with rule number 326 would not need to be asked. Consequently all design rules, and thus questions, concerning point control methods of sequence control would not be addressed.

The writing of the questions was thus conducted with two main goals in mind : limiting the number of questions required, by identifying dependencies between design rules, and explicitly stating the sequencing of questions to be asked through the use of a syntax that could be used later when encoding the proposed tool. A portion of the user questions are presented in appendix C. An example of the questions asked under the menu category of "Menu Appropriateness" is presented in table 3.

Table 3. Example of user questions for the menu appropriateness category

A1 Menu Appropriateness
Existance : 1. Does the software being evaluated contain a menu system as one of its dialogue types ? Yes or No
User : 2. Does the use of menu systems as a dialogue type effectively correspond to the characteristics of the intended users : level of skill, and training ? Yes or No
3. What is the level of user experience with the menu system ? Inexperienced, Moderately Experienced, Experienced, or Variable
Task : 4. Is the principle task routine with fixed proceduresw ? Yes or No
5. Does the principle task require minimal entry by the user ? Yes or No

4.4 Functioning of the Mock-Up

It should be clear that the proposed tool will be orientated towards the evaluation of a menu system. Yet, one thing that has yet to be specified - the scope of this evaluation. The prototype system, and consequently the focus of the mock-up, will be designed to identify positive and negative features present in the menu system being evaluated. The prototype system will guide a user through an evaluation such that all positive and negative aspects of menu system objects are recorded ; no evaluation as to the overall goodness or badness of the system will be presented. The evaluation of the proposed tool will be limited to the identification of individual objects within the menu system being evaluated. The individual menu objects evaluated will be identified as agreeing or contradicting a specific design recommendation and evaluation criteria. The tool will identify deficiencies and present a list of the criteria that were violated while giving no weight to faults. It will simply note deficiencies such that an evaluator can determine the seriousness of the problems at the end of the evaluation through the examination of the output results.

The mock-up is based on an algorithmic approach which follows a strictly defined sequence, resulting in a non-flexible system. The unique development goals of the mock-up (i.e., identify relational rules, examine the knowledge source, and identify a reasoning path) do not require that the mock-up be fully functional. The need to quickly verify the feasibility of an evaluation tool of this sort out weighed the need to provide a highly interactive system. As a result the interaction with the mock-up is cumbersome. There exists no way to streamline the

question sequence, select specific variable to examine, or request specialized output formats. With this in mind, the following paragraphs we will outline how the evaluator will input his responses while utilizing the mock-up, and in what format the output will be presented.

Input Sequence:The input from the evaluator (i.e., the users of the proposed tool) using the mock-up system will be in the form of responses to a list of question whose content is guided by the evaluator's responses. In order to reduce vagueness, in the responces of the evaluator, each question presents a limited set of specific responce values. The overall structure and ordering of the question will follow the sequence order determined by the developed flowcharts. A top-down sequence order will be followed for the menu objects. Within each menu object a depth first approach based on the hierarchy of attributes will be followed and completed before continuing to the next menu object in the menu structure.

Output Matrix:The evaluators responses will be recorded into the form of an 18 X 4 evaluation matrix between abstract levels and evaluation criteria (figure 2). The output matrix will be a count of the design rules within the prototype that directly apply to the four different abstract level and the eight evaluation criteria. The classification of the design rules during the development of the knowledge source allows the identification of both the criteria and the abstract level for each design rule. Each matrix cell will contain four numbers that present relevant evaluation counts : System Rules, Instantiated Rules, Positive Aspects, Negative Aspects. The first number within each cell, System Rules Present, will identify the total number of rules within the knowledge source that evaluate the menu structure under this intersection. This number informs the user of the richness of information present by category and is not a function of the evaluation. This number is static and will typically change only when the knowledge source is upgraded or changed.

The second number within each cell, Instantiated Rules, will identify the number of design rules in the category that the sequence of questions has addressed and whose conclusion has been triggered (instantiated). Typically this will be a subset of the first category - design rules present. It is only a subset, because not all rules are expected to be accessed. If certain objects do not exist in the menu system then the corresponding design rules that evaluate its attributes would not be accessed. For example : if the menu system did not employ grouping within Menu Panel Lists, then questions concerning the sequencing of groups, group labels, and of their format would not take place. Thus, it is obvious that not all available design rules may be triggered (instantiated). When a given design rule has been instantiated this indicates that the mock-up has made some determination about a menu object's agreement with or violation of a preferred or recommended state.

These first two matrix numbers are more informative than evaluative and only give information as to how complete the knowledge source is (design rules present) and how thoroughly it was queried (instantiated) by the mock-up. These two numbers, although not

greatly important for evaluation purposes, provide additional information that can be seen as a means to increase user confidence that a significant amount of the available domain knowledge has been addressed.

The remaining two numbers in each box will identify the number of instantiated design rules that indicate Positive Aspects and Negative Aspects of the menu system. The sum of these two categories should always equal that of the number of instantiate rules. Design rules within the knowledge source, designed to locate and label potential system faults, cannot be considered equivalent since the priority of the relationship between various system faults is not known. Therefore, the magnitude of the numbers within these last two matrix cells can not be considered to give a clear indication of goodness or badness of the menu system.

The output matrix provides only a count of what is present in the system and is not itself an indication of goodness or badness. Since each design rule can not be considered equivalent the evaluator will need to survey the original text of rules found in each cell category. Rules written in clear natural language will allow the user to do his own filtering to determine the level of system effectiveness. This approach, though not optimal, allows the user to conduct an examination independent from the evaluation results. The matrix will provide a historical summary of the examination, identifying positive and negative points, for later review.

In addition to the matrix a separate chart will be provided which lists the objects and attribute values of the menu system which were identified, yet for which the positive or negative correlation with the design rules or design criteria could not be determined. Typically this will mean that the preferred value for these aspects has not been acceptable determined through prior research. For example the prototype system might ask a user : Which sequencing rule was used to determine option order : natural order, alphabetic, frequency ? The prototype could easily determine which rule is used, but it is unknown which is the best. Is important for the user to know the positive, negative, and unknown aspects of the menu system being evaluated.

This matrix may be the corner stone of other more efficient evaluation techniques to be applied later. Yet, before additional evaluation outputs can be proposed (graphics, summaries, comparison techniques) the prototype system must first function in a consistent and understandable fashion when identifying areas of discrepancy. Incorporating techniques that make evaluative comparisons between identified deficiencies (i.e., identifying which problem is more seveer) will require additional knowledge acquisition to determine priority structures. Thus, the first developmental stages of the mock-up have concentrated on identifying and counting design deficiencies.

The matrix also serves a second important function of acting as a development tool. What this does for development is to provide a mechanism which we can be used to determine

Figure 2. Example of an evaluation matrix for the mock-up

Evaluation Matrix

Abstract Level	Compatibility	Consistency	Workload	Adaptibility	User Control	Sig Codes	Guidance	Error Manage
Conceptual	pre 37 ins 25 neg 15 pos 10	pre 40 ins 15 neg 5 pos 10	pre 60 ins 45 neg 35 pos 10	pre 37 ins 22 neg 17 pos 5	pre 55 ins 55 neg 12 pos 43	pre 12 ins 12 neg 5 pos 7	pre 30 ins 15 neg 12 pos 3	pre 33 ins 33 neg 21 pos 12
Semantic	pre 42 ins 39 neg 15 pos 24	pre 22 ins 5 neg 5 pos 0	pre 8 ins 5 neg 3 pos 2	pre 6 ins 1 neg 1 pos 0	pre 16 ins 15 neg 5 pos 10	pre 47 ins 25 neg 15 pos 10	pre 20 ins 15 neg 15 pos 5	pre 30 ins 15 neg 5 pos 10
Syntatic	pre 84 ins 55 neg 33 pos 22	pre 38 ins 36 neg 25 pos 11	pre 12 ins 0 neg 0 pos 0	pre 20 ins 0 neg 0 pos 0	pre 33 ins 0 neg 0 pos 0	pre 28 ins 0 neg 0 pos 0	pre 20 ins 0 neg 0 pos 0	pre 22 ins 18 neg 6 pos 12
Lexical	pre 47 ins 15 neg 10 pos 5	pre 32 ins 20 neg 15 pos 5	pre 62 ins 37 neg 12 pos 25	pre 37 ins 12 neg 1 pos 11	pre 41 ins 33 neg 10 pos 23	pre 18 ins 18 neg 10 pos 8	pre 27 ins 5 neg 0 pos 5	pre 20 ins 15 neg 15 pos 5

Evaluation Criteria

if a reasoning paths is not being followed, making it easier to make subsequent corrections. For example, consider the case where a significant number of rules have been developed and the mock-up is run with the results matrix presented (fig. 3). A cursory look at this matrix tells us that a problem with the question sequence may exist in the syntactic row. This can be said because, although five of the eight criteria have values of zero in the instantiated category, a significant number of design rules exist that may address these aspects. It is hard to imagine a menu system that has no syntactic components to its objects, thus a problem is identified in the system and the developer can work to determine the cause. The design rules, initially identified as applying to the category, can be viewed in their original text to help quickly identify the problem. This will make it easier to identify whether the problem lies in the syntax used in the development of the design rules, or if it is a problem of the sequence of questions not reaching these design rules.

5 Conclusions

This paper has presented three efforts that have been conducted under the general perspective of developing tools and methods for the evaluation of the user interface. The goal of these efforts has been to investigate the feasibility of developing an evaluation tool which can effectively transfer context dependent human factors recommendation - in more autonomous formats.

Before efforts to specify a design for an evaluation tool could take place an explicit definition, and organizational pattern, for menu systems and menu objects needed to be developed. Thus, the first section of this report presented the results of an investigation into the domain of menus and proposed a complete definition for menu systems. The proffered definition divided the domain into three areas : allowable menu system structures ; identifiable menu system objects ; and menu system sequence control aspects. Each of these three areas were expanded in the paper in support of the definition, and function as an organizational framework for the menu domain. The menu objects and attributed that could be used to trigger design rules in a functioning expert system were also identified at this stage. Although this organizational pattern is still new, it serves the purpose of organizing the domain knowledge necessary for the development of a prototype system.

Two other subsequent efforts, which rely highly on this organizational pattern, were also reported in this paper : 1) the development of a knowledge source containing design rules for menu systems, and 2) the development of a functioning mock-up of a proposed interface evaluation tool. The development of the knowledge source began by the identification of two knowledge bases that contain much of the documented expertise in the domain of menus (i.e., Aschehoug and Scapin, 1990 ; Williams, 1990). These two knowledge bases contain the original set of design recommendations that have been modified. Because these two knowledge bases were developed independently of each other, and contained redundancies and extra information not associated with menu systems, steps were taken to streamline these knowledge bases. The development of the design rules in the knowledge source is traced from the identification of pertinent knowledge bases through a series of modifications made to the set of identified design recommendations. The resulting knowledge source is the most highly specific, non-redundant, and integrated knowledge source available for the design and evaluation of menu systems.

The third effort reported in this paper was development of a functioning mock-up. The goal of the mock-up was to examine the knowledge source for completeness, identify dependent design rules, and develop a reasoning path that the proposed system could follow. This was accomplished by constructing detailed flowcharts of the relationships between individual design rules in the knowledge source, and writing actual user questions in a sequence order dictated by the paths identified in the flowcharts.

The mock-up has been designed to identify positive and negative features present in the menu system being evaluated. It functions by asking a set of predefined questions to the evaluator of a menu system. The sequence order of the question follows the reasoning paths determined in the flowcharts. The evaluator's responses are recorded into an 18 X 4 evaluation matrix between abstract levels and evaluation criteria. The output matrix presents a count of the design rules within the prototype that directly apply to the four different abstract level and the

eight evaluation criteria. Each of the cells in the matrix contain four numbers that present relevant evaluation counts : System Rules Present, Instantiated Rules, Positive Aspects, Negative Aspects. The evaluative output of the mock-up is limited to the identification of individual objects, and making decisions about the objects conformance or non-conformance to the design rules and evaluation criteria. The mock-up is capable of identifying deficiencies, presenting the list of violated design criteria, while giving no weight to faults. It notes deficiencies in a menu system such that an evaluator can independently determine the seriousness of the problems.

Having a completed mock-up helps to visualize how a prototype evaluation system could function. The writing of actual user questions has helped to accurately estimate the number of questions that would be needed in the proposed evaluation tool. It also has assisted in the identification of design rules whose instantiated conclusion statement could be used to determine the premiss statement of a dependent rule - identified design rule dependencies. Additionally, relationships between menu objects could be examined and further specified, allowing us to determine a sequence order for the user questioning.

Since this report represents the first stages in what is expected to be a long development effort, caution needs to be observed when evaluating its merits. Obviously additional testing and validation needs to be conducted. The organizational pattern presented in section two of this report needs further clarification and empirical validation. Some user testing needs to be done to ensure the validity of the proposed organization. Within the knowledge source large amounts of information concerning the design of menu still needs to be added. This is because of the general lack of literature and empirical research. For example, presently there is no information in the knowledge source concerning the users model of the overall menu structure. Most of the developed design rules address the functionality and lexical aspects of menu objects. Before an expert system could function effectively (i.e., gaining the confidence of its users) these areas must be addressed.

Presently the mock-up is based on an algorithmic approach which follows a strictly defined sequence which results in a non-flexible system. The need to quickly verify the feasibility of an evaluation tool of this sort out weighed the need to provide a highly interactive system. As a result the interaction with the mock-up is cumbersome and needs to be upgraded. In addition to further testing and user validation, certain functional characteristics need to be added to make the tool more flexible. Users should be given the flexibility to evaluate the menu system from multiple points of view. For example, users should be able to investigate all aspects of the menu system according to one design criteria (i.e., compatibility, or error protection, etc...), by a single level of abstraction (i.e., selecting all aspects that refer to the conceptual or lexical levels), or by examining a single menu object without the need examine all

menu objects. This type of flexibility, which could be provided in the next prototype development, would result in a more usable system.

Additional investigations should also determine if the proposed tool could be include as part of a tool box, providing the evaluation tool with the ability to directly extract certain facts from the menu system without requiring user inputs. This would reduce the number of questions that would need to be asked of a user. Technical aspects could also be improved by allowing the user to respond to questions while actually using the menu system being evaluated. The user question could be presented on screen, while the user is given the ability to manipulate the menu system, thereby concurrently verifying his responses.

Even though the mock-up is limmited in flexibility it has successfuly demonstrated the feasibility of a computer based interface evaluation tool. The domain knowledge necessary for the data base of an expert system now exists in a organized and highly detailed source ; representing the knowledge acquisition effort. The development of a functioning mock-up has shown that independent design rules can be integrated to perform an actual evaluation. The development of a mock-up has provided an explicit guide for developing the reasoning paths that would be followed in an actual expert system. These results provide the necessary support for continuing research on providing autonomous tools for the design and evaluation of the user interface.

6 References

American National Standard for Human Factors Engineering of Visual Display Terminal Workstations. Human Factors Society, the society, Santa Monica, CA, USA, ANSI/HFS Standard No. 100-1988, 1988 (1-90).

Aschehoug, F., & Scapin, D.L. (1990). *Reference rules data base.* Unpublished Manuscript.

Bastien, C. (1991). *Validation de critères Ergonomiques pour l'évaluation d'interfaces Utilisateurs.* (Rapport de recherche n° 1427). Rocquencourt, France : Institut National de Recherche en Informatique et en Automatique.

Bastien, C., & Scapin, D. (1991). *A validation of ergonomic criteria for the evaluation of user interfaces.* International Journal of Human Computer Interaction, in press.

Brown (1988). *Human-Computer interface design guidlines.* Norwood, NJ : Ablex.

DIN (1987). DIN 66234 Part 8 : Principles of dialogue design.

Foley, J.D., & Van Dam A. (1984). *Fundamentals of interactive computer graphics.* Reading, MA : Addison-Wesley.

Gibbons, S.C. (1990). *Feasibility of an expert system based menu interface evaluation tool.* (Progress report, . Rocquencourt, France : Institut National de Recherche en Informatique et en Automatique.

Gibbons, S.C. (1991). *Organizing the domain of menus.* Manuscript submited for publication.

Helander, M. (Ed.). (1988). *Handbook of Human-Computer Interaction.* Amsterdam : Elsevier.

MacGregor, J., Lee, E., & Lam, N. (1986). Optimizing the structure of database menu indexes : A Decision Model of Menu Search. *Human Factors, 28* (4), 387-399.

McDonald, J.E., Molander, M.E, & Noel, R.W. (1988). Color-Coding categories in menus. In Soloway, E., Frye, D., & Sheppard, S.B. (Eds.) *Proceedings of the CHI '88.* Reading, MA : Addison-Wesley.

Mosier, J. N., & Smith, S. L. (1986). Application of guidelines for designing user interface software. *Behavior and Information Technology,* 5 (1), 39-46.

Norman, K.L. (1991). *The psychology of menu selection.* Norwood, NJ : Ablex.

Norman, K.L. & Chin, J.P. (1988). The effect of tree structure on search in a hierachical menu selection system. *Behavior and Information Technology, 7* (1), 51-65.

Norman, K.L., & Chin, J.P. (1989). The menu metaphor : food for thought. *Behavior and Information Technology, 8* (2), 125-134.

Pedersen, K. (1989). *Expert Systems : programing.* New York : Wiley & Sons.

Pollier, A. (1990). *Evaluation d'une interface par des ergonomes : diagnostics et stratégies.* (Rapport de recherche n° 1391). Rocquencourt, France : Institut National de Recherche en Informatique et en Automatique.

Rivlin, C., Lewis, R., & Cooper, R.D. (1990). *Guidlines for screen design.* Oxford, England : Blackwell Scientific.

Salvendy, G. (Ed.). (1987). *Handbook of human factors.* Canada : Wiley & Sons.

Scapin, D.L. (1990a). Organizing human factors knowledge for the evaluation and design of interfaces. *International Journal of Human-Computer Interaction*, 2 (3), 203-229

Scapin, D.L. (1990b). Des critères ergonomiques pour l'évaluation et la conception d'interfaces. *Proceedings du XXVIème Congrès de la SELF*. Montréal, 3-5 octobre.

Scapin, D.L. (1987). Guide Ergonomique de conception des interfaces homme-machine. (Rapport de recherche n° 77). Rocquencourt, France : Institut National de Recherche en Informatique et en Automatique.

Shneiderman, B. (1987). *Designing the user interface.* Reading, MA : Addison-Wesley.

Smith, S.L., & Mosier, J.N. (1986). Design guidelines for the user interface software. Technical Report ESD-TR-86-278, National Technical Information Service. Springfield, Virginia : (NTIS No. AD A177 198).

Williams, J.R. (1990). *Menu design guidlines.* ISO/CD 9241-14.

Williges, B., and Williges, R.C. (1984). Dialogue design considerations for interactive computer systems. In F.A. Muckler (Ed.) *The Human Factors Review.* The Human Factors Society, Santa Monica.

Appendix A

Knowledge Source*

* The following appendix presents a sub-set of the complete knowledge source.

A1 Menu Appropriatness

75) Selection of a menu dialogue should be based on user characteristics.
(appropriate - conceptual - compatibility)

76) Selection of a menu dialogue should be based on skill of users.
(appropriate - conceptual - compatibility)

77) Selection of a menu dialogue should be based on user training.
(appropriate - conceptual - compatibility)

86) Menu dialogues should be selected if the future users will be inexperienced.
(appropriate - conceptual - compatibility)

84) Menu dialogues should be selected if some user will not be familiar with all functions of the software system.
(appropriate - conceptual - prompting)

85) Menu dialogues can be selected if tasks are routine, with fixed procedures that require only minimal entry.
(appropriate - conceptual - minimal actions)

85a) Menu dialogues can be selected if tasks are routine, with fixed procedures requiring entry of arbitrary data.
(appropriate - conceptual - mental load)

83) Menu dialogues can be selected if initial entry accuracy is critical.
(appropriate - conceptual - error protection)

109) In menu dialogues where graphical input is required, menu options should be displayed as a set of graphical icons.
(appropriate - conceptual - consistency)

265) Within a menu system the menu panels should contain as many menu options as possible if rapid search time is important.
(appropriate - conceptual - minimal action)

A2 Menu System

338a) Within the menu system the availability of individual menu options, the category to which they belong, their names, and the means to select them should always be evident to the user.
(menu system - conceptual - mental load)

78) Menu dialogues should be flexible if user characteristics are variable.
(menu system - conceptual - compatibility)

79) Menu dialogues should be flexible for very experienced computer users.
(menu system- conceptual - compatibility)

80) Menu dialogues should not be flexible for moderately experienced computer users.
(menu system- conceptual - compatibility)

283) Menu panels in a hierarchical structures should be presented in the shortest time possible (less than 500 miliseconds)
(menu system - syntactic - imm feedback)

231) Information should be presented to the user in a directly usable form : the user should not be required to translate, transpose, change units, or interpolate.
(menu system - conceptual - mental load)

261a) The organization of the menu panels within a menu system should facilitate the user's ability to find and to select options relevant for the task
(menu system - conceptual - mental load)

263) In the menu system menu panels should be organized to minimize depth and maximize breadth, if the menu options have no natural grouping, but can be grouped or ordered in a manner which is unambiguous and easily learned by the user population.
(menu system - conceptual - mental load)

261) The organization of the menu panels within a menu system should reflect user expectations.
(menu system - conceptual - compatibility)

276) Within a menu system navigational cues should be provided which can help users learn the menu structure and navigate within the structure.
(menu system - conceptual - mental load)

276a) Within a menu system navigational cues can be provided by : distinctive and compoundable titles, menu panel designators, option designators, graphic techniques, simultaneous display of menu panels, and menu maps.
(menu system - syntactic - prompting)

152) Multiples paths to accommodate both experienced and inexperienced users should be provided.
(menu system - conceptual - flexibility)

288) The menu system should provide multiple pathways for navigation if it is logical to access menu panels from differing points in the menu structure.
(menu system - conceptual - flexibility)

286) The menu system should provide users a means to return the initial (beginning) menu panel from any menu in a menu structure.
(menu system - syntactic - user control)

280) If the menu system utilizes the simultaneously dismay of menu panels the hierarchical relationship between the menu panels should be apparent to users.
(menu system - conceptual - prompting)

B Menu Panel

73) The menu panel should be symmetrically balanced.
(menu panel - syntactic - clarity)

59) The menu panel format should have a standardized organization.
(menu panel - syntactic - consistency)

33) Menu panels should be formatted according to the users standard format, if a users standard format exists.
(menu panel - syntactic - compatibility)

B1 Menu Panel Titles

117) Every menu panel should have a main title.
(main titles- conceptual - mental load)

61b)- Menu panel titles should not include extra or non-relevant information.
(mp titles - semantic - concision)

113a) In the menu panel titles consistent terminology should be used across menu panels..
(mp titles - semantic - consistency)

31) In menu panel titles user jargon should not be used.
(mp titles - semantic - sig or codes)

44) Menu Panel titles should not be contracted to the point where they cannot be read.
(mp titles - syntactic - clarity)

55a) In menu panel titles hyphenation should be avoided.
(mp titles - syntactic - clarity)

B1.1 Main Titles

117a) Main tittles should clearly state the contents of the menu panel.
(main titles - semantic - prompting)

131b) Main titles can be selected to semanticaly indicate current position in the structure.
(main titles - semantic - prompting)

131b1) Main titles can information coding to indicate current position in the structure.
(main titles - syntactic - prompting)

277) Main titles should be semanticly distinctive and descriptive of the information contained within the menu panel.
(main titles - semantic - sig codes)

277a) Main titles should be compoundable such that they can be put together into multiple word titles.
(main titles - syntactic - sig codes)

73a) Main titles should be centered above the menu panel list.
(main title - syntactic - by location)

B1.2 Group lables

130) Group label should be given to each grouping in a menu panel list.
(labels - conceptual - compatibility)

352) Group Labels should use information coding to make the labels perceptually distinct from the menu options they represent.
(group labels - syntactic - clarity)

73b) Group labels should be left justified above the option group.
(group labels - syntactic - by location)

B1.3 Menu Panel Desingators

131c) Menu panel designators can be used to indicate current position in the structure.
(mp designators - conceptual - prompting)

278) If menu panel designators are utilized the coding scheme should make the menu system structure obvious to the user.
(mp designators - conceptual - sig of codes)

138) Menu panels should appear consistently in the same display location.
(menu panel - syntactic - consistency)

138a) Menu panel should not appear simultaneously in different screen locations.
(menu panel - syntactic - mental load)

C Sequence Control

186) Sequence control actions should be compatible with task aspects.
(seq control - conceptual - compatibility)

186a) In sequence control frequent procedures should be easy
(seq control - conceptual - minimal action)

302) In sequence control the number of keystrokes required to select and execute options should be minimized.
(seq control - syntactic - concision)

286a) The means provided to the user to return the initial (beginning) menu panel from any menu in a menu structure should be simple.
(seq control - syntactic - minimal actions)

186b) In sequence control destructive actions should be difficult.
(seq control - conceptual - err protection)

245a) Common errors in sequence control actions should not produce processing sequences that are different from those intended.
(seq control - syntactic - err protection)

188) In sequence control the ability to return to the previous step in sequence control should be provided.
(seq control- conceptual - err correction)

188a) In sequence control, returning to the previous step in sequence control should be easy.
(seq control - syntactic - err correction)

C1 Sequence Control Methods

290) Several alternatives sequence control methods should be provided for the selection of menu options.
(seq methods - conceptual - flexibility)

331) Direct control methods of -keyed inputs- should be provided in addition to the pointing control methods, If this is consistent with task requirements and system constraints.
(seq methods - conceptual - flexibility)

324) Cursor control methods may be used as a substitute to direct control methods.
(seq methods - conceptual - compatibility)

112) Sequence control methods should be compatible with the user characteristics.
(seq methods - conceptual - compatibility)

112a) Sequence control methods should be compatible with the task characteristics.
(seq methods - conceptual - compatibility)

291) Sequence control methods should require separate actions for selecting and executing menu options.
(seq methods - syntactic - err protection)

289b) Sequence control methods should facilitate the users' operation of the menu system.
(seq methods - conceptual - minimal actions)

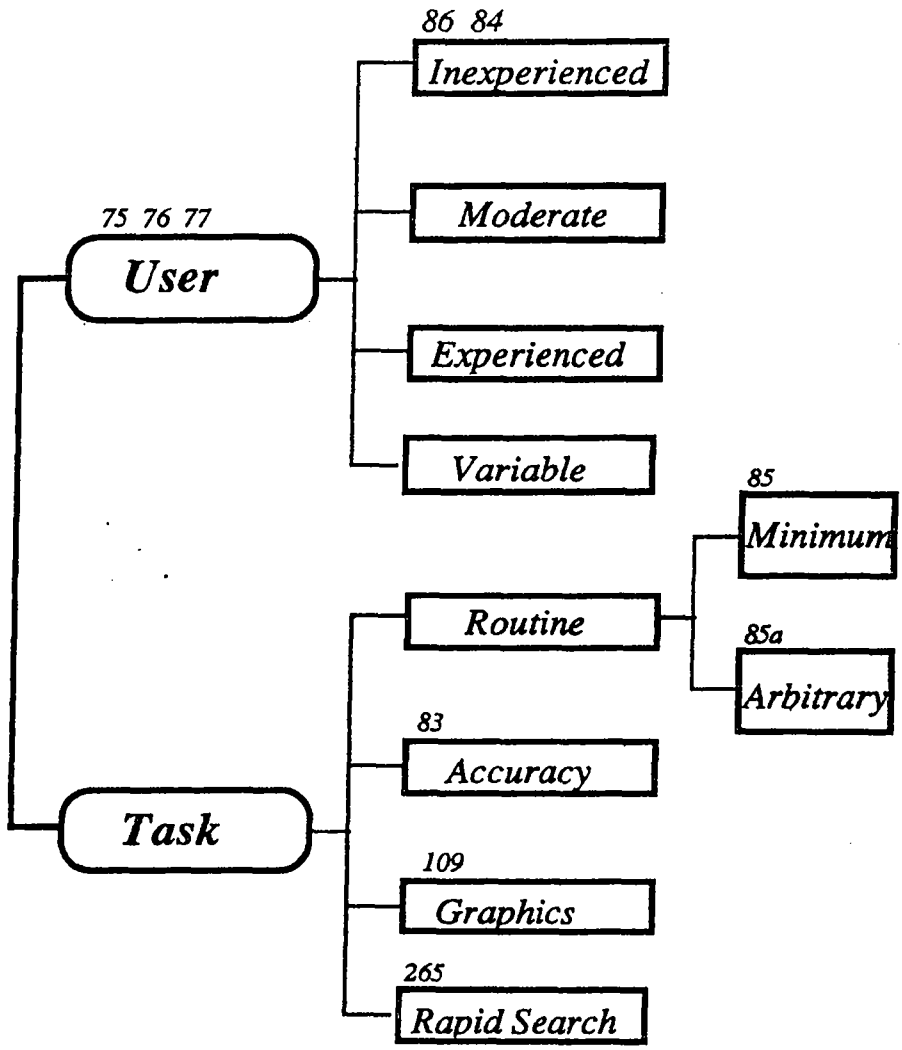
289) The selection of sequence control methods should depend on the task, dialogue requirements, and individual preference.
(seq methods - conceptual - compatibility)

Appendix B

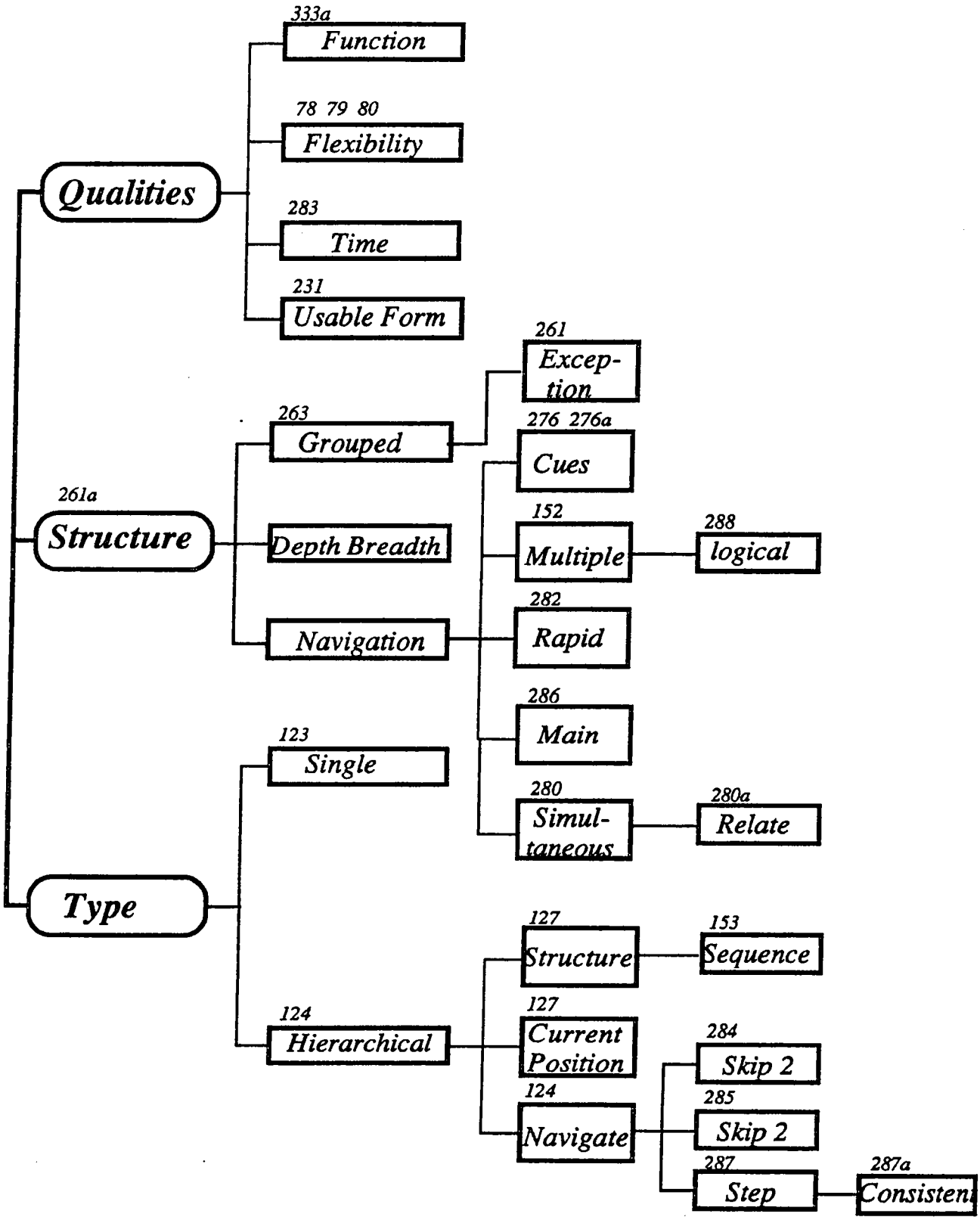
Flowcharts[§]

§ The following appendix presents all the flowcharts created during the development of the mock-up system.

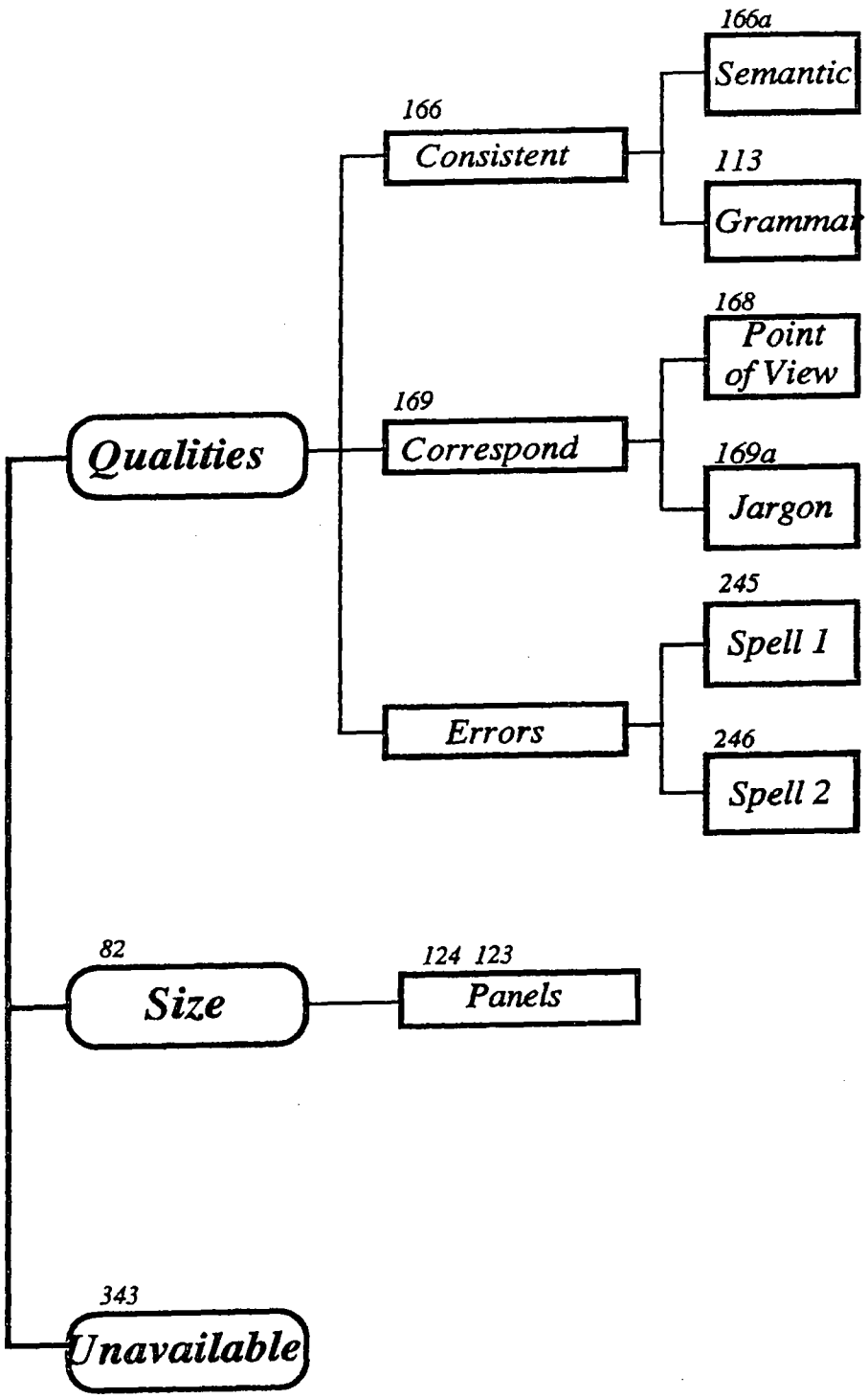
A1 Menu Appropriateness



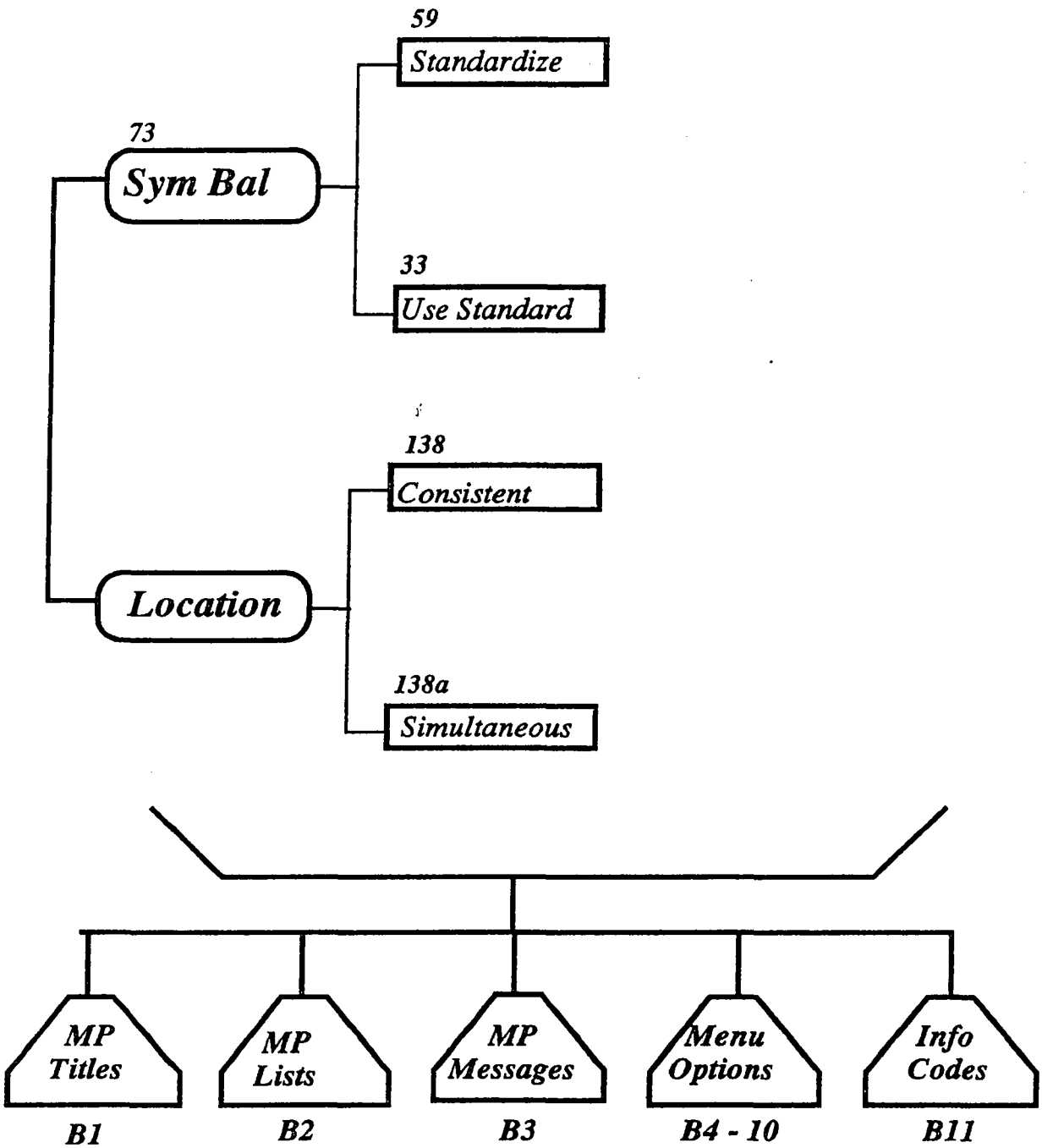
A2 Menu System



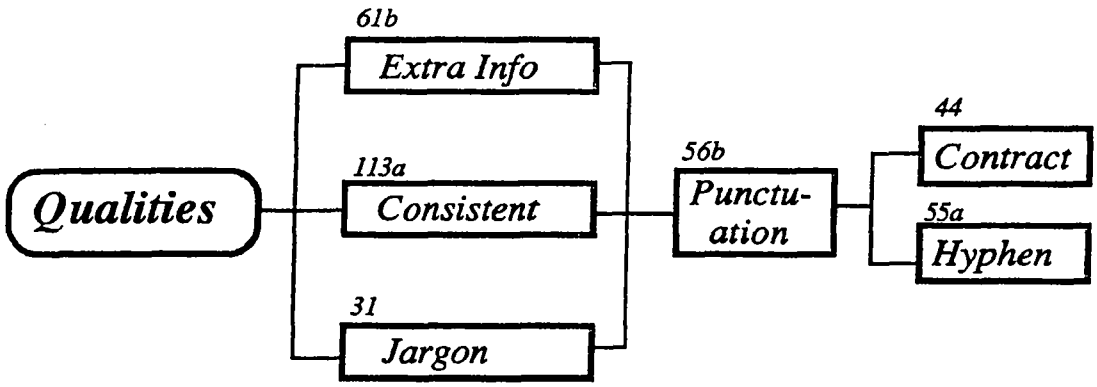
A3 Command Set

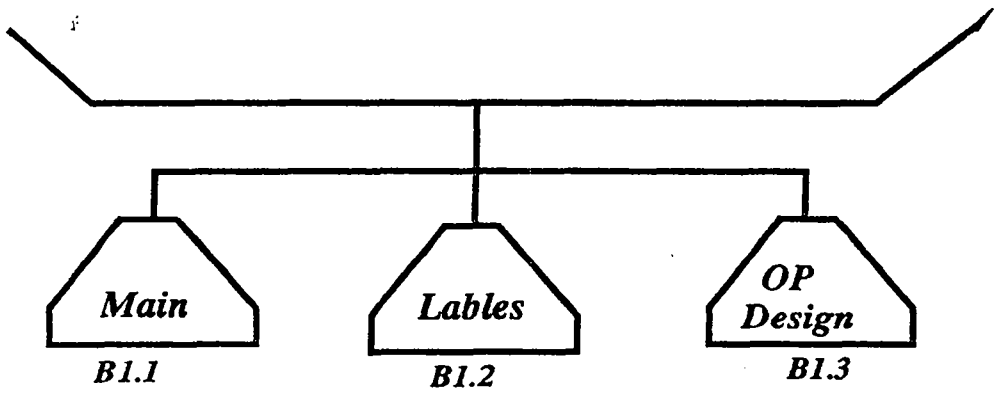


B Menu Panel

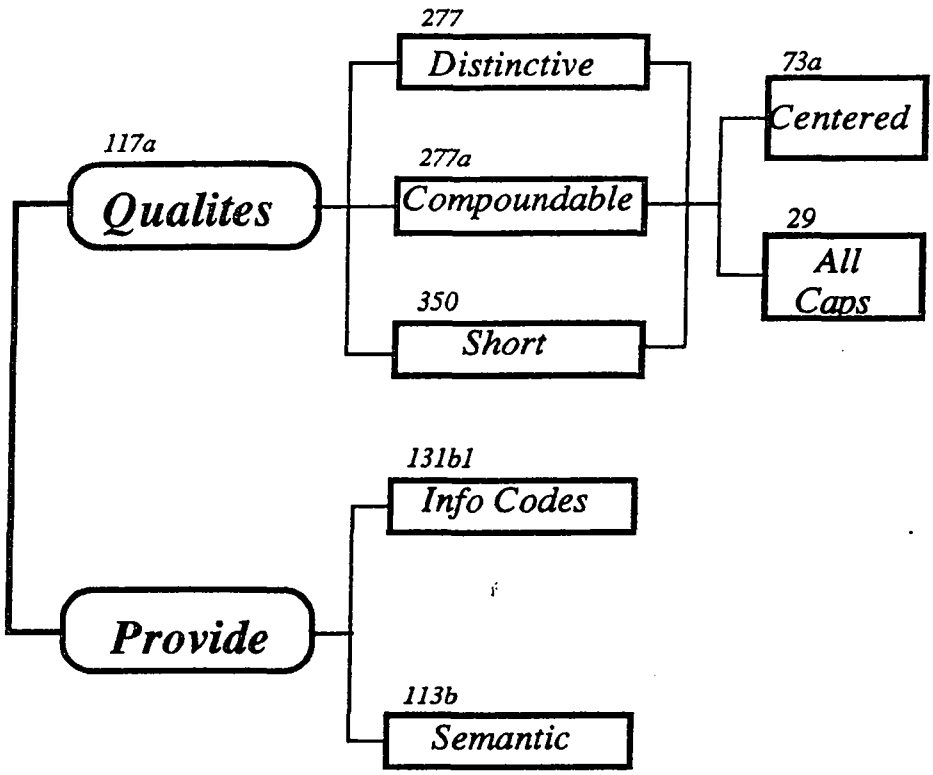

59
Standardize
73
Sym Bal
33
Use Standard
138
Consistent
Location
138a
Simultaneous
MP Titles
B1
MP Lists
B2
MP Messages
B3
Menu Options
B4 - 10
Info Codes
B11

B1 Menu Panel Titles

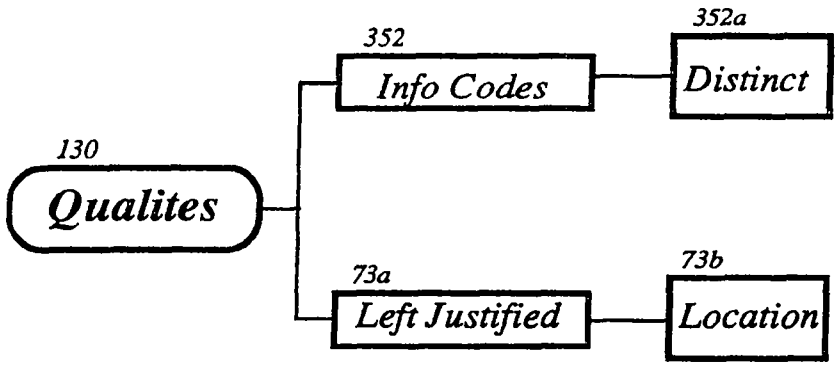




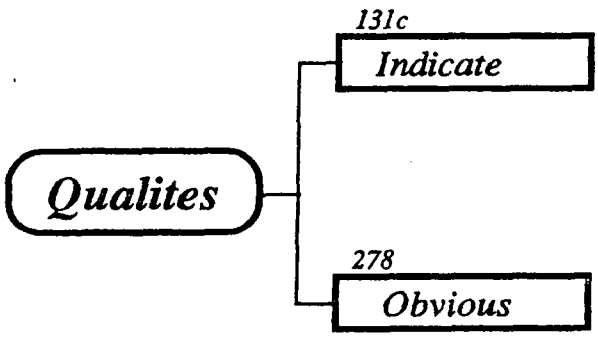
B1.1 Main Titles



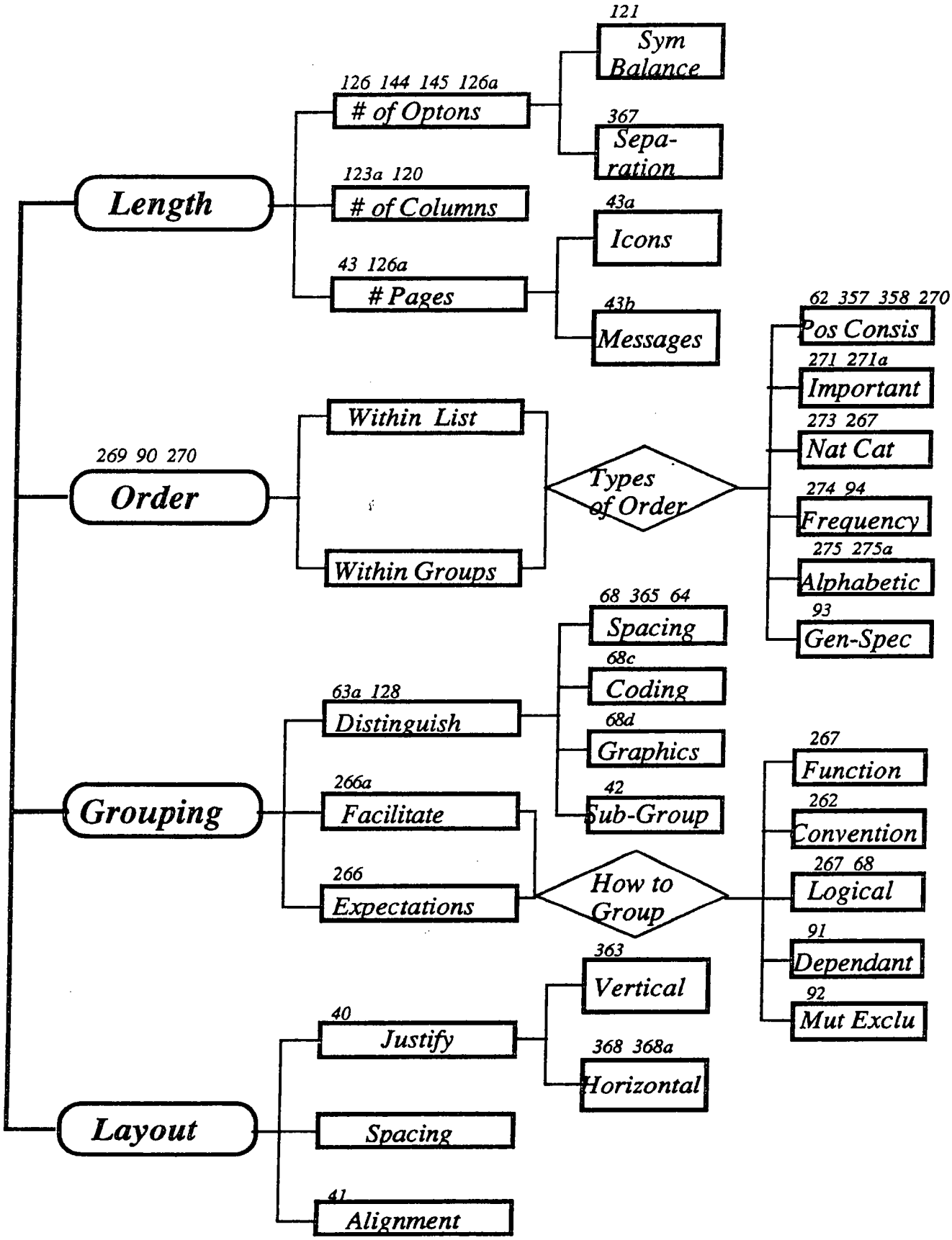
B1.2 Group Labels



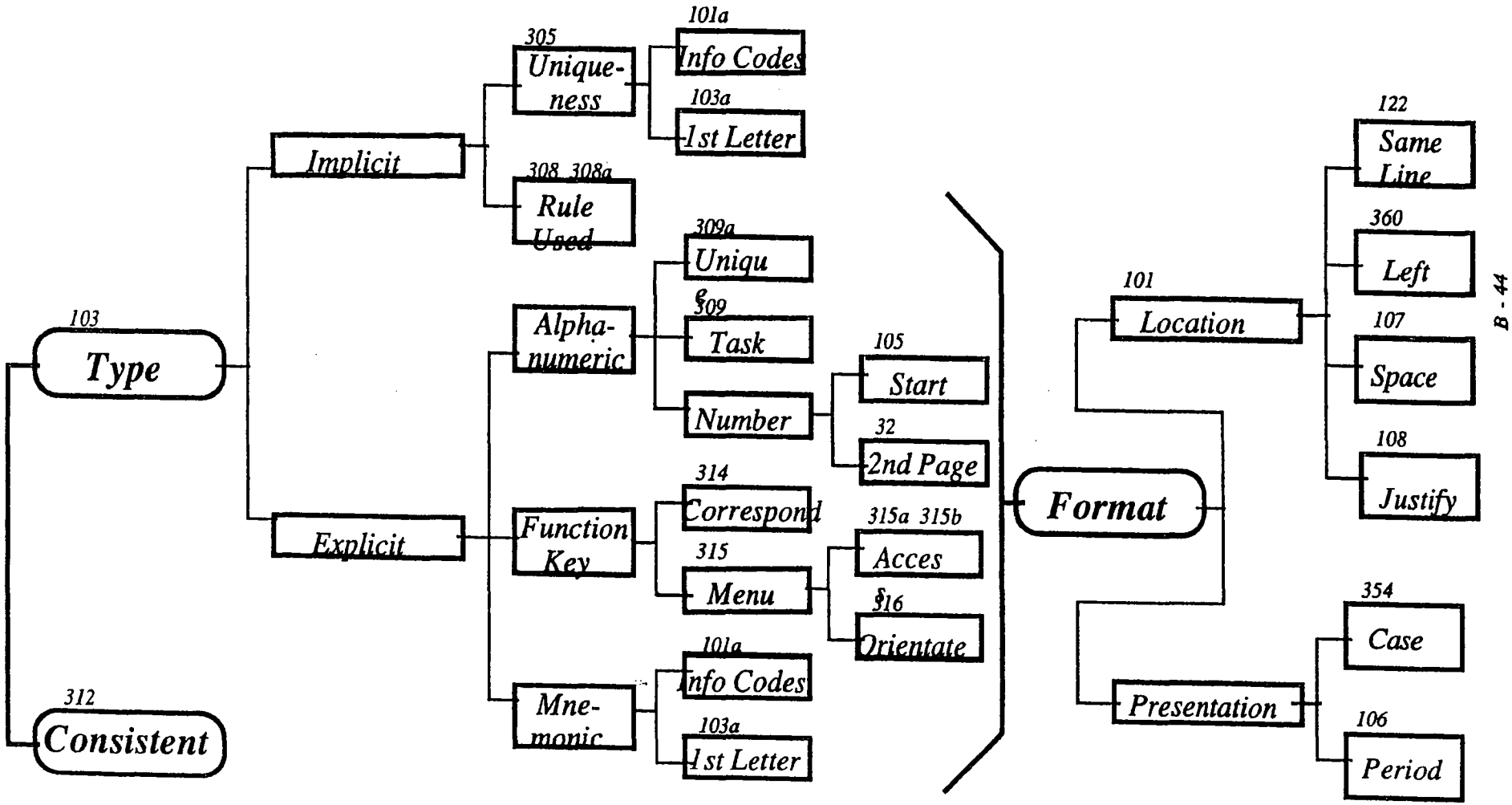
B1.3 Menu Panel Designators



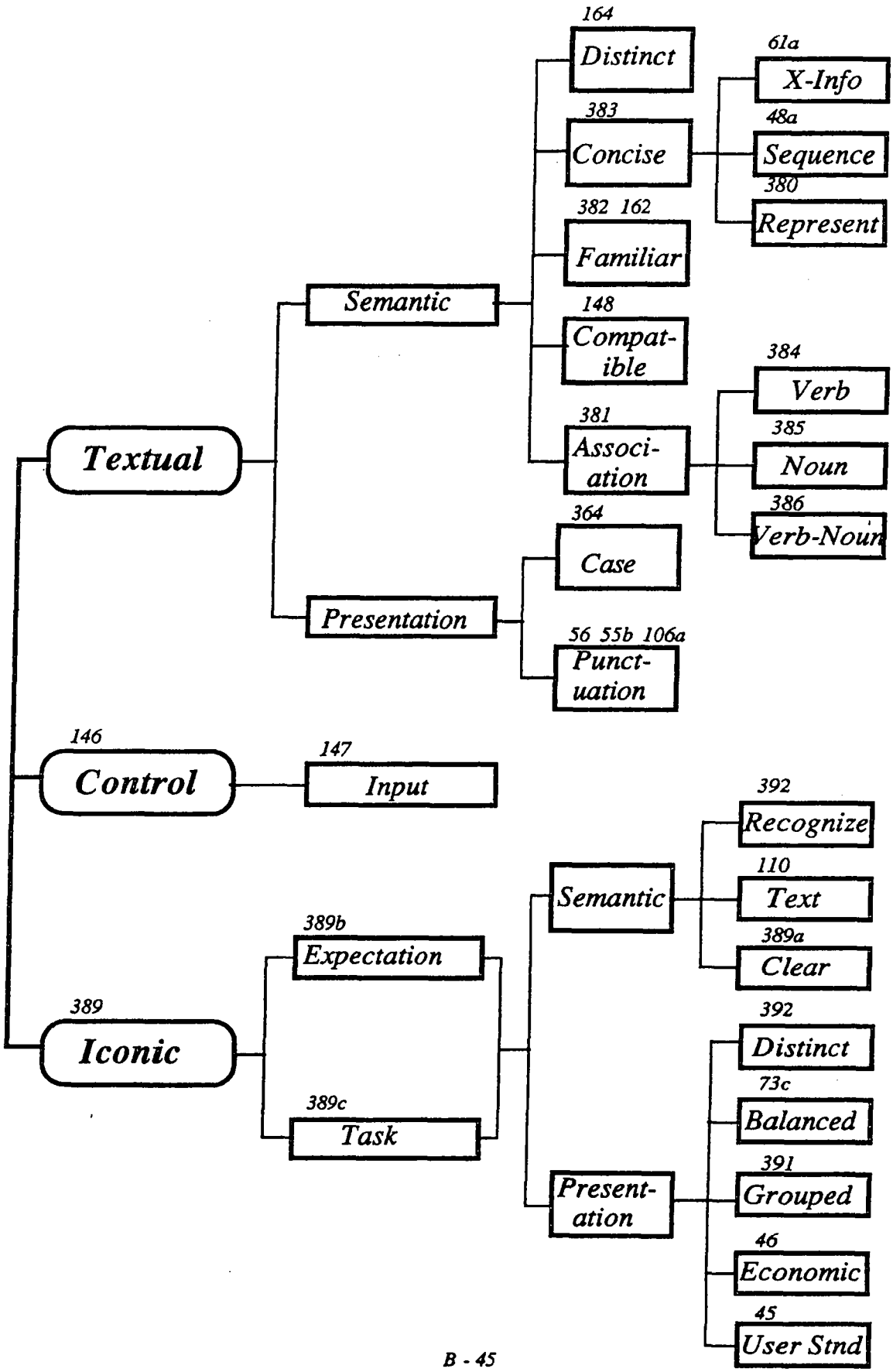
B2 Menu Panel List



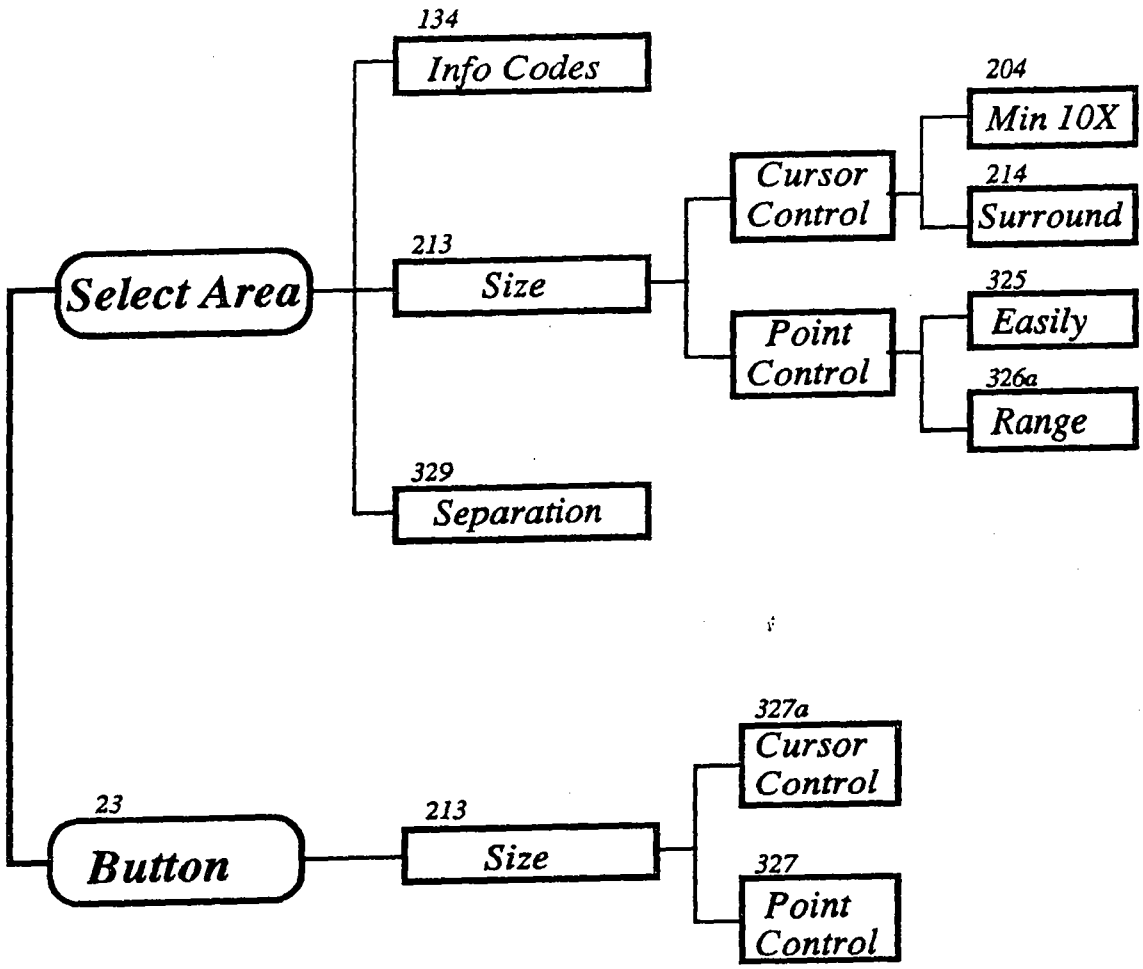
B2.1 Option Designators



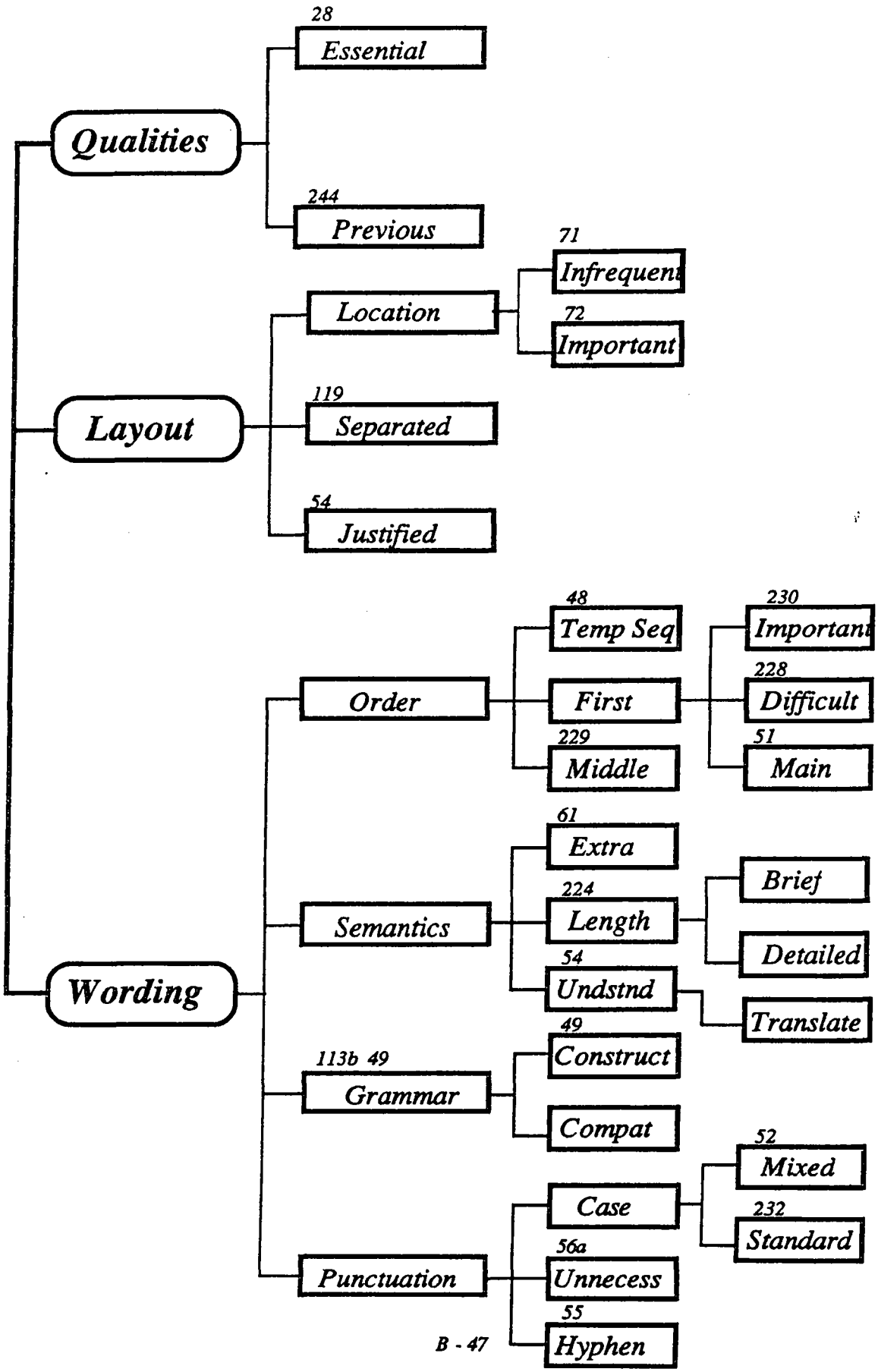
B2.2 Menu Options



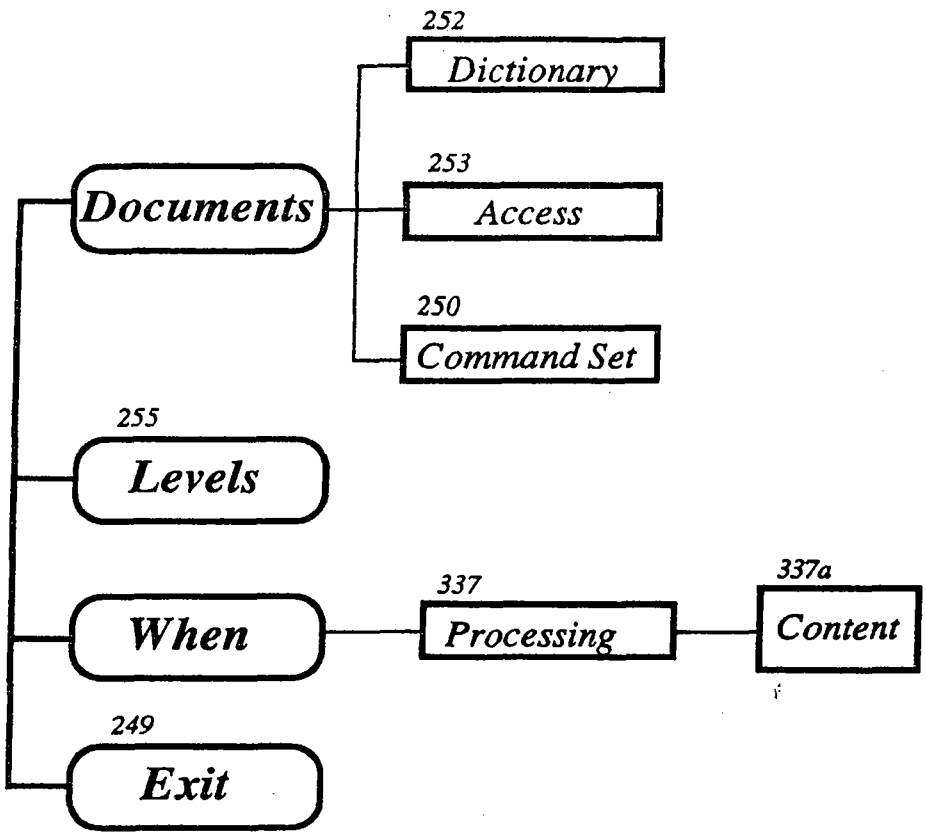
B2.3 Selection Area



B3 Menu Panel Messages

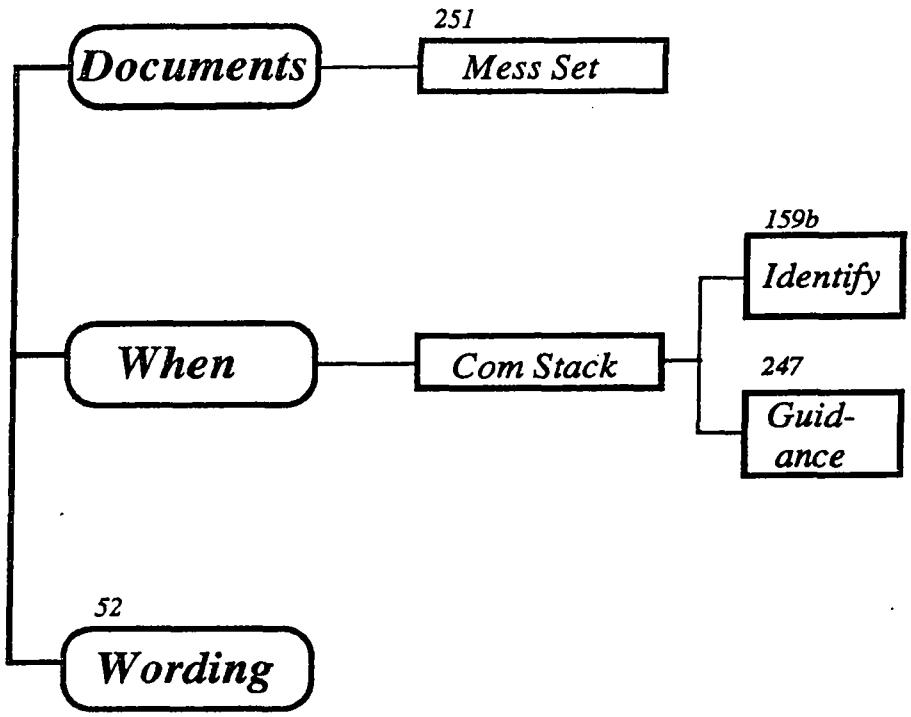


B3.1 Informative Messages

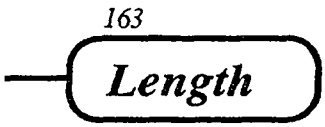


B3.2 Directive Messages

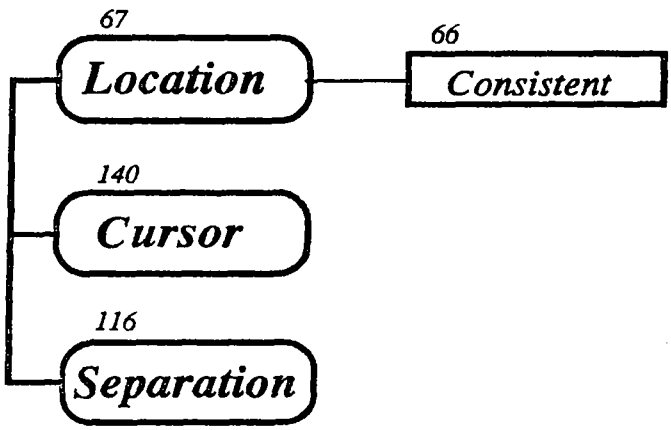
B3.3 Error Messages



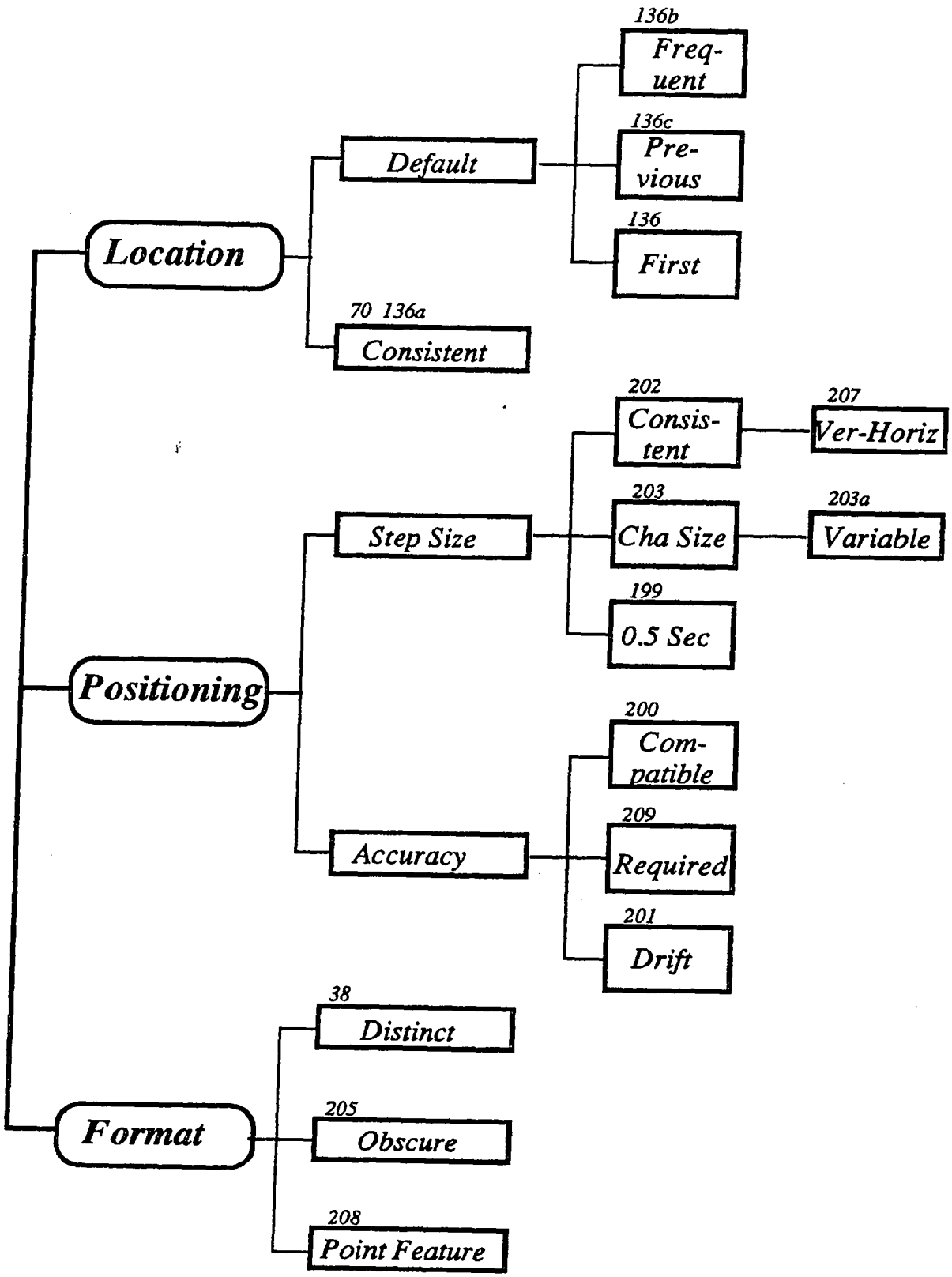
B4 Keywords



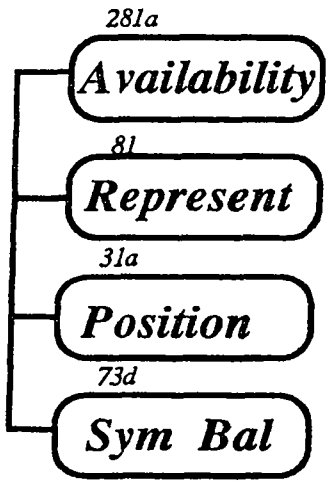
B5 Command Line



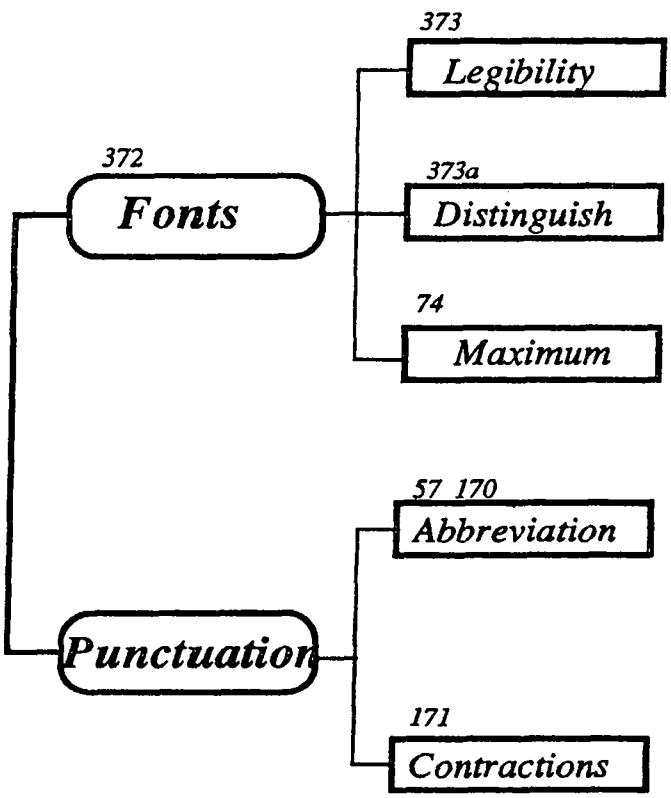
B6 Cursor


Location
Default
136b
Freq-uent
136c
Pre-vious
136
First
70 136a
Consistent
Positioning
Step Size
202
Consis-tent
207
Ver-Horiz
203
Cha Size
203a
Variable
199
0.5 Sec
Accuracy
200
Com-patible
209
Required
201
Drift
Format
38
Distinct
205
Obscure
208
Point Feature

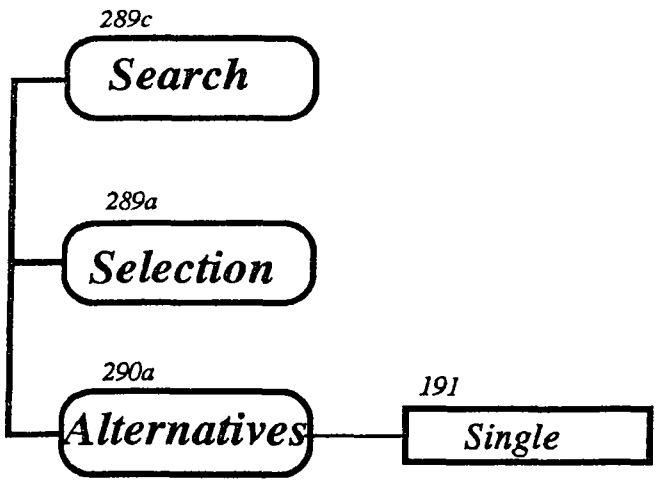
B7 Menu Maps



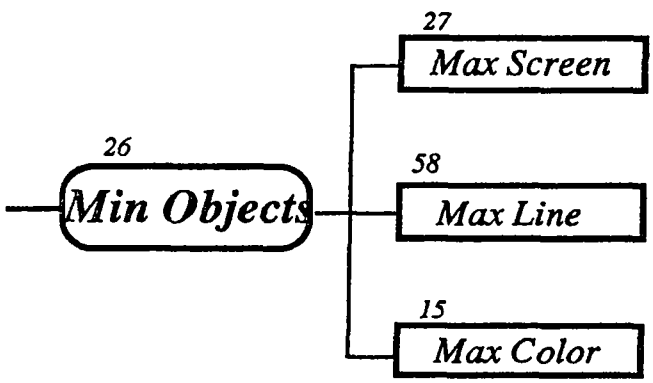
B8 All Text



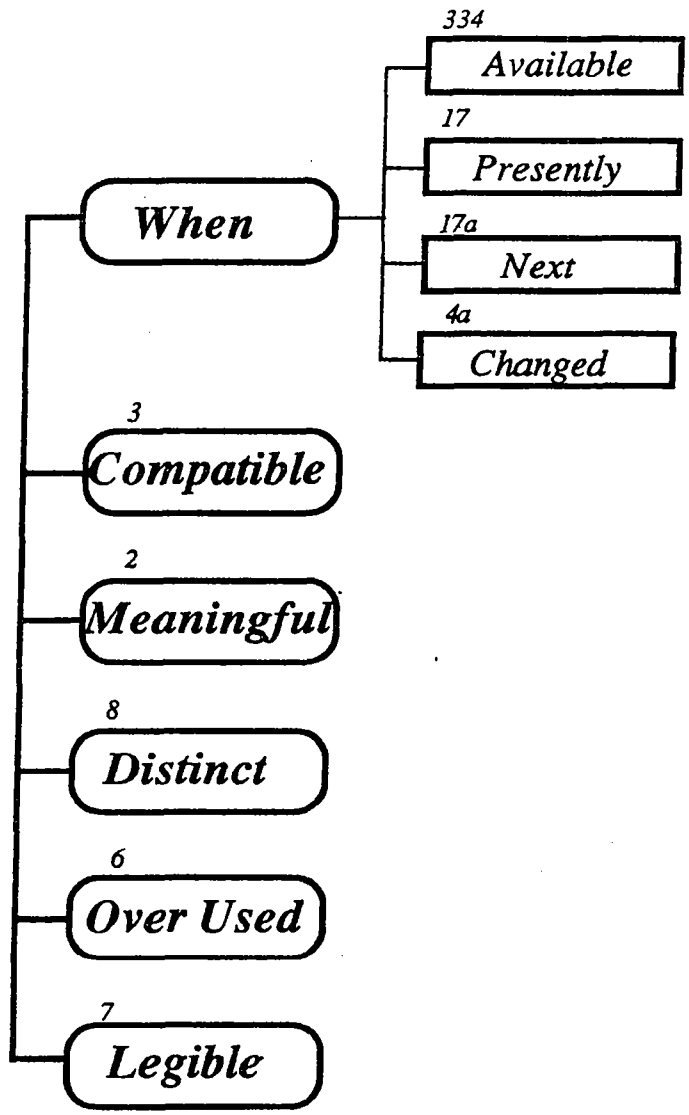
B9 Input Device

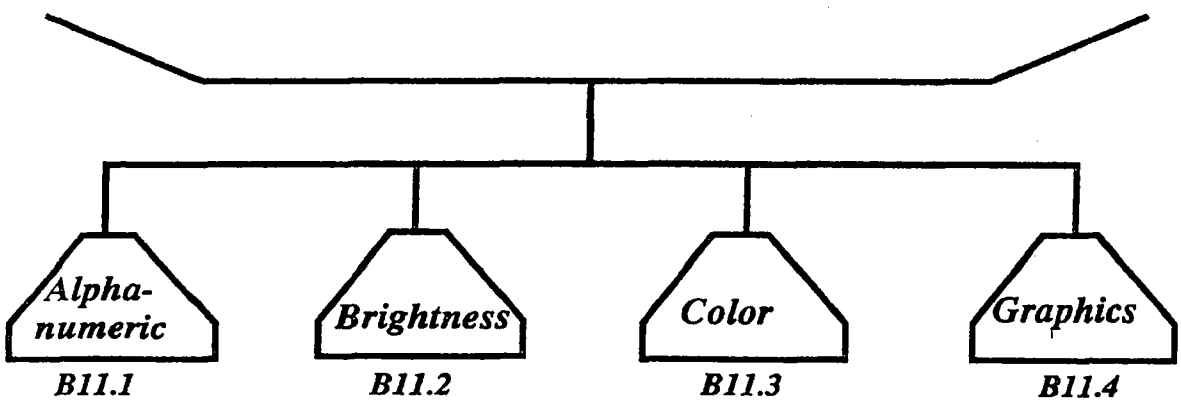


B10 Display Screen

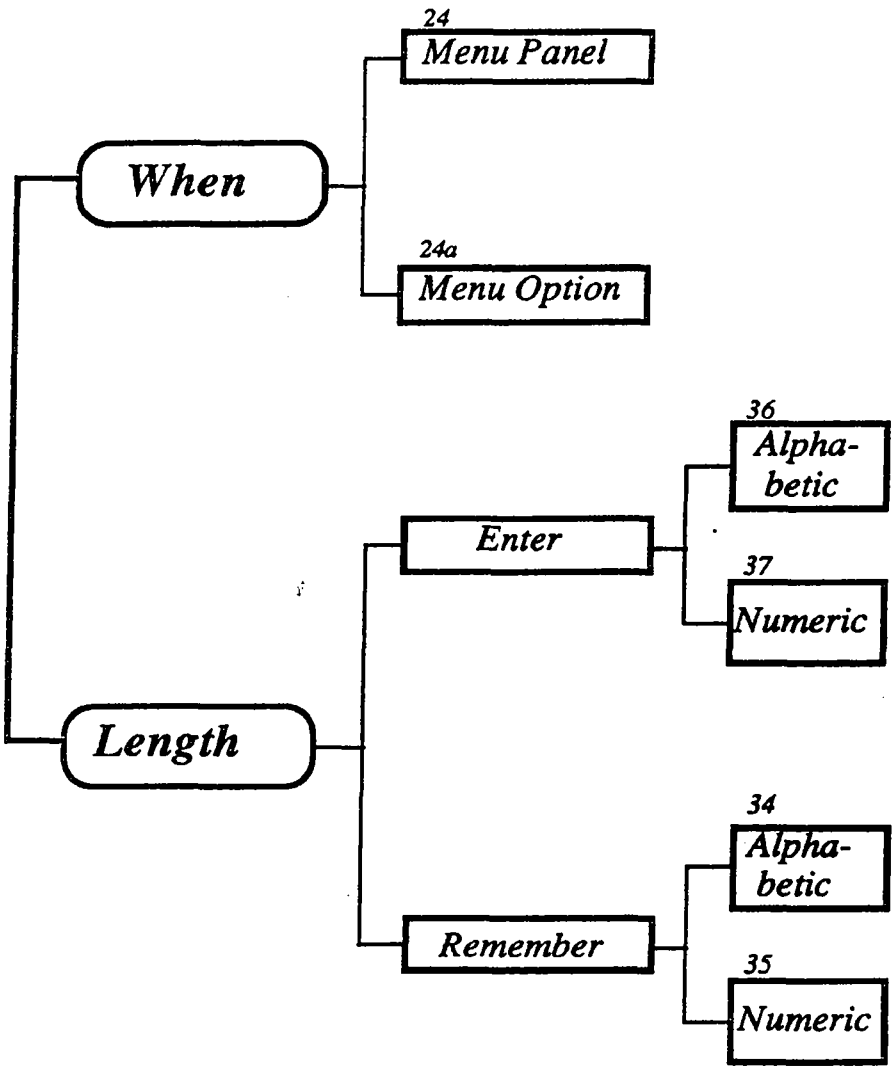


B11 Information Coding

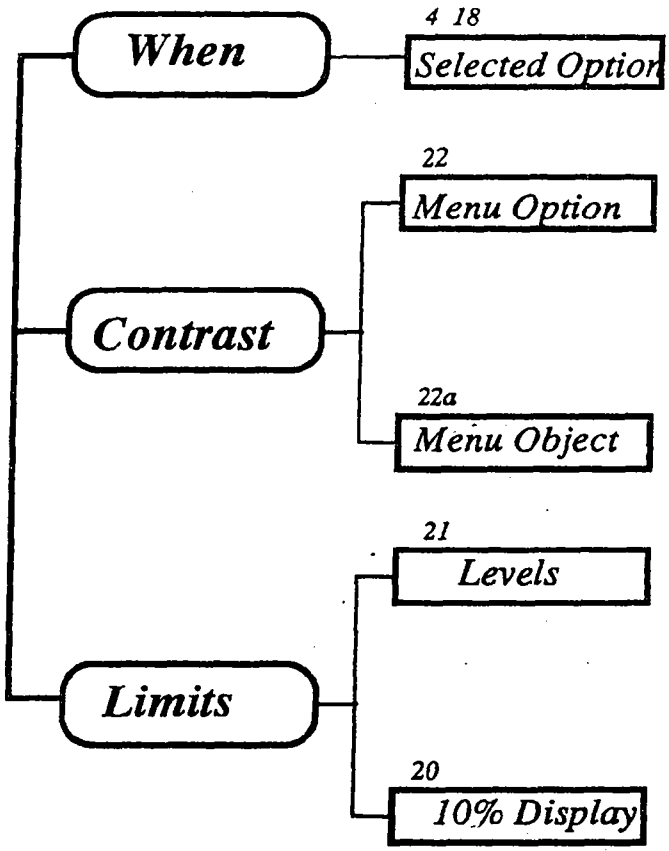




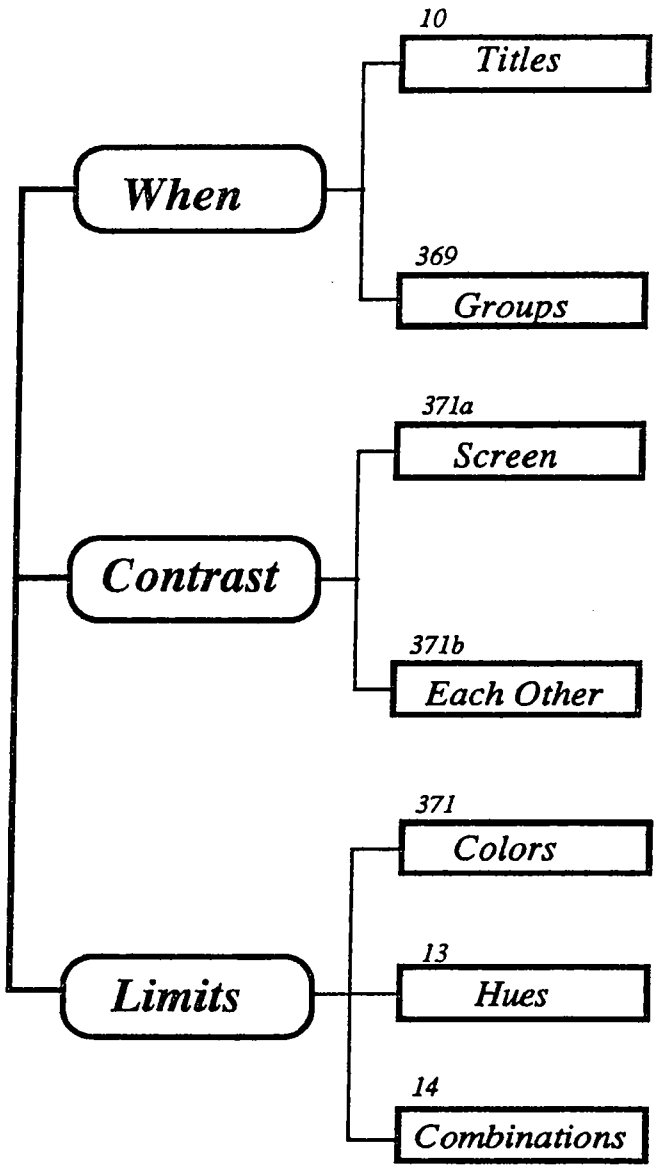
B11.1 Alphanumeric Codes



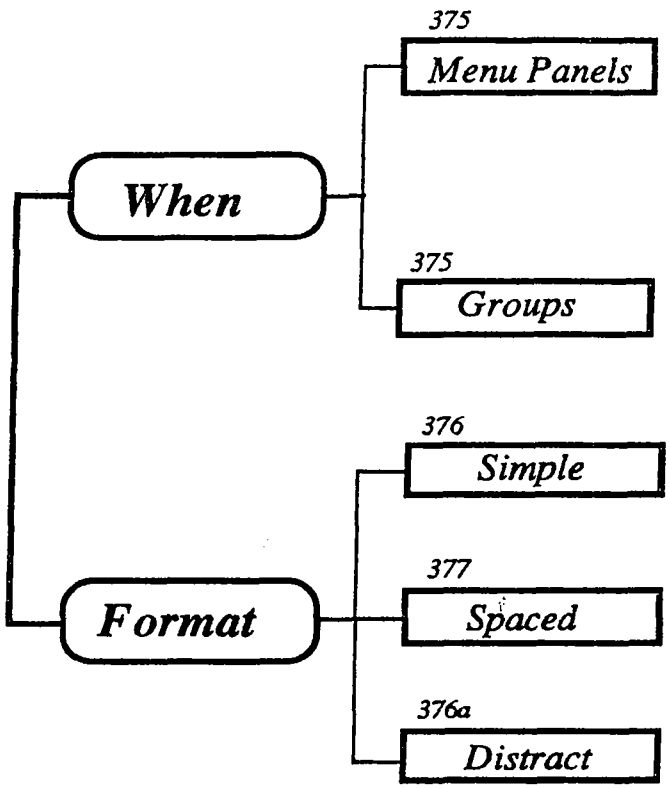
B11.2 Brightness Codes



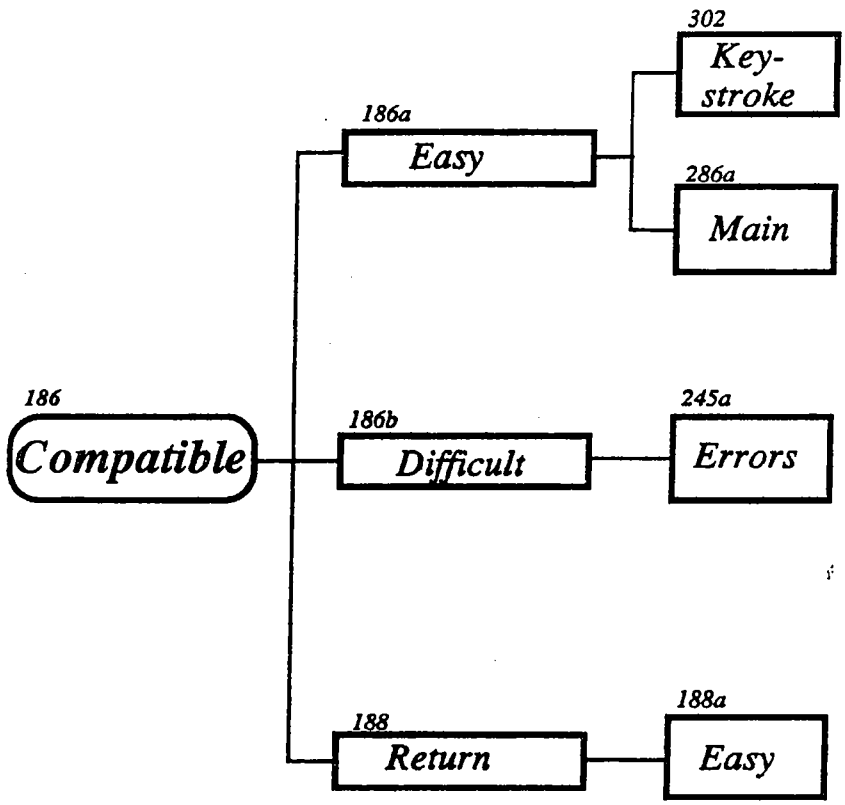
B11.3 Color Codes

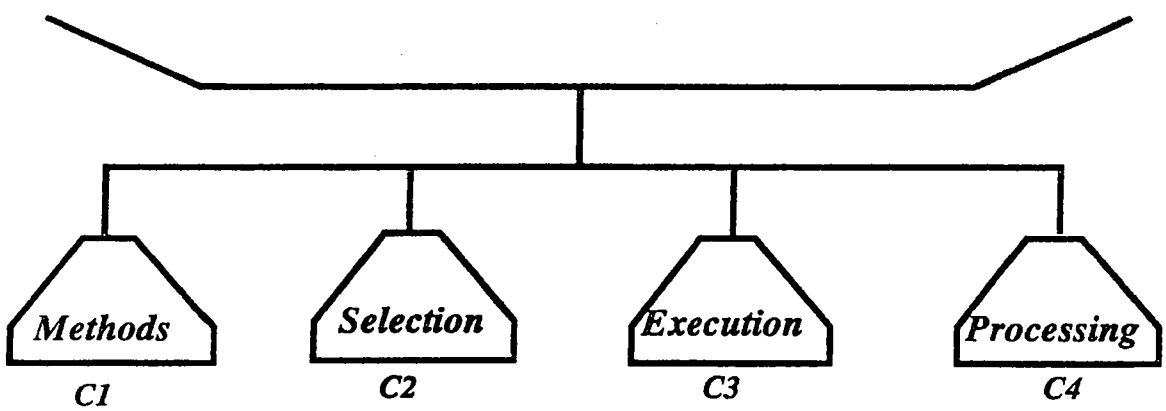


B11.4 Graphic Codes

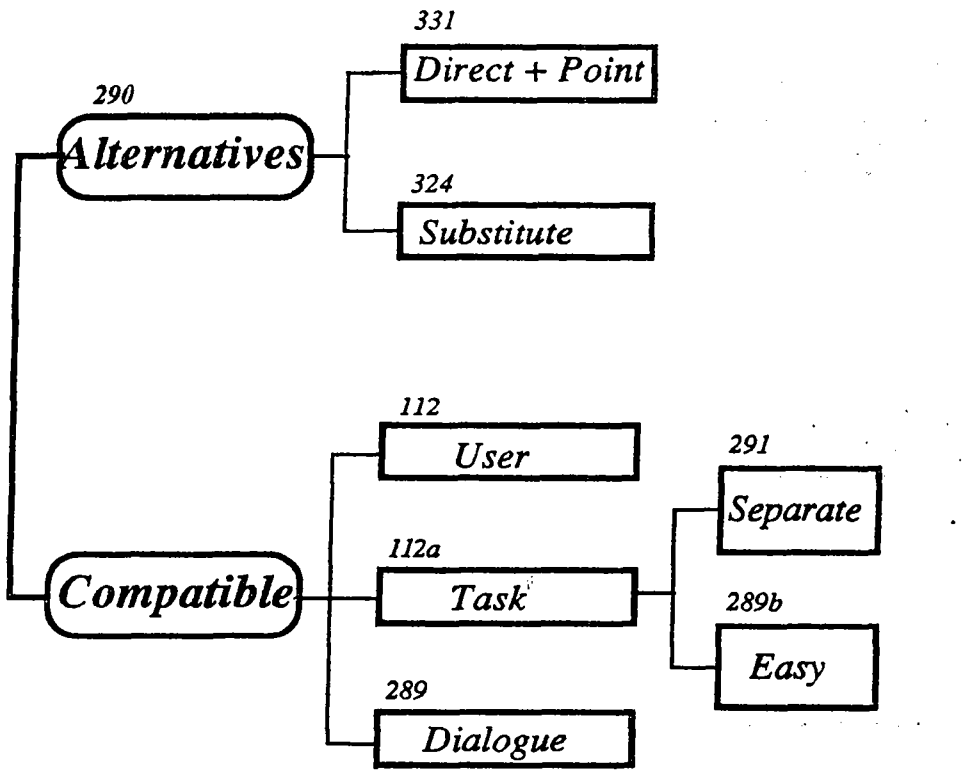


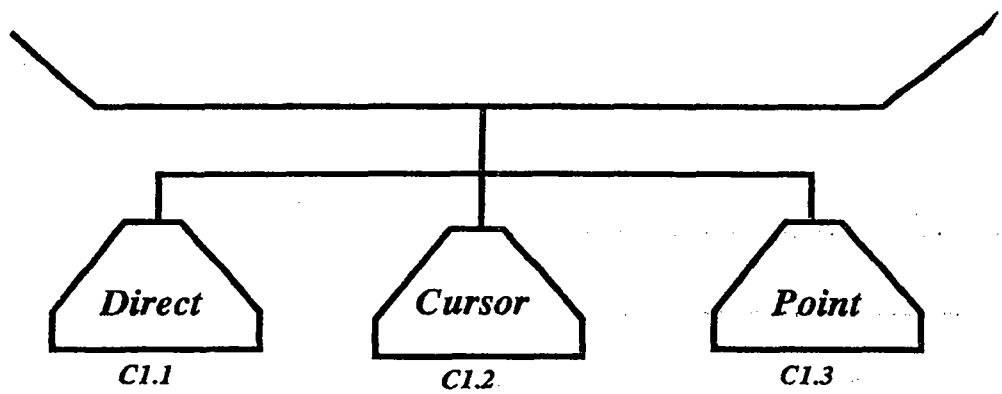
C Sequence Control



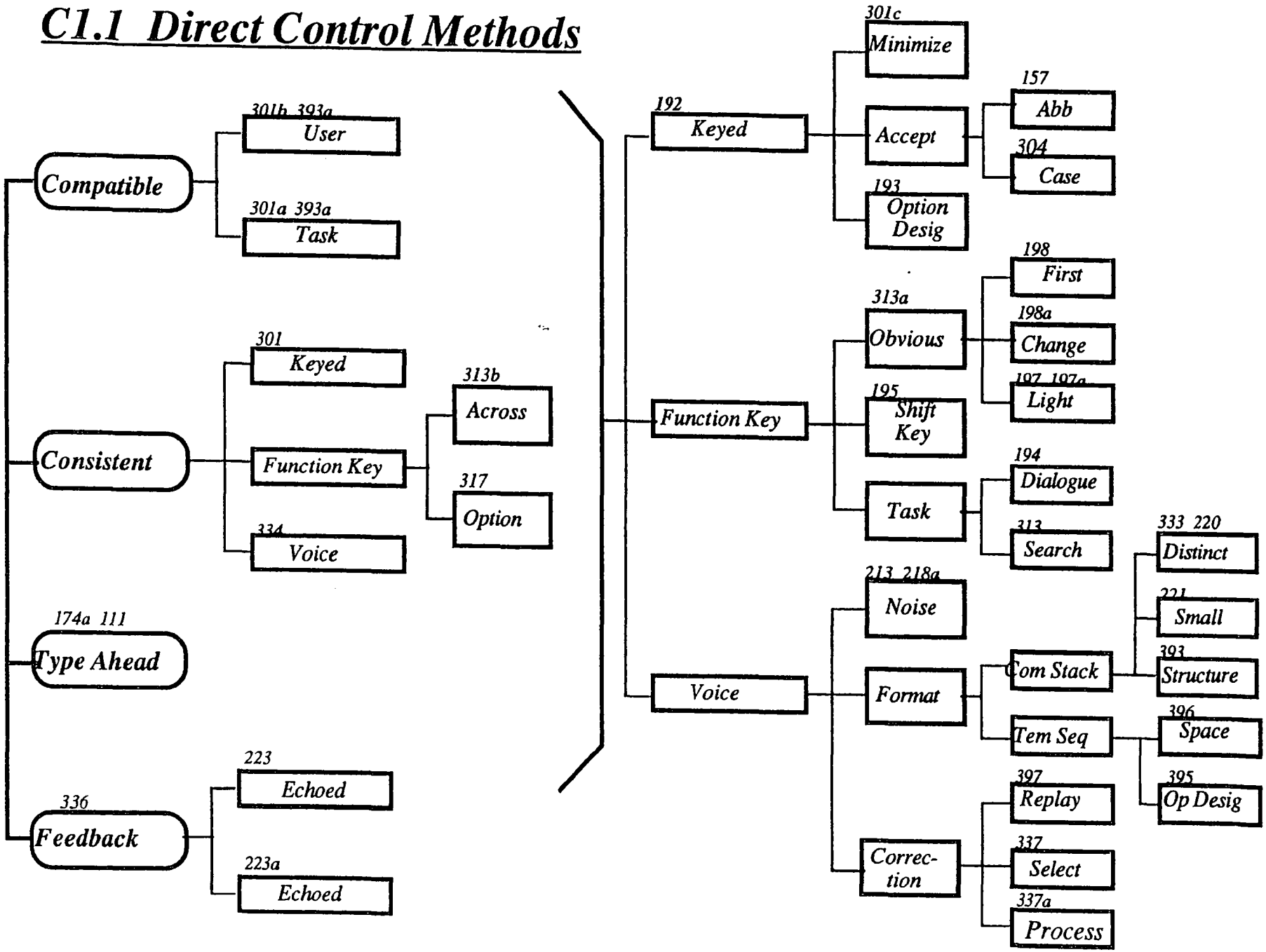


C1 Sequence Control Methods

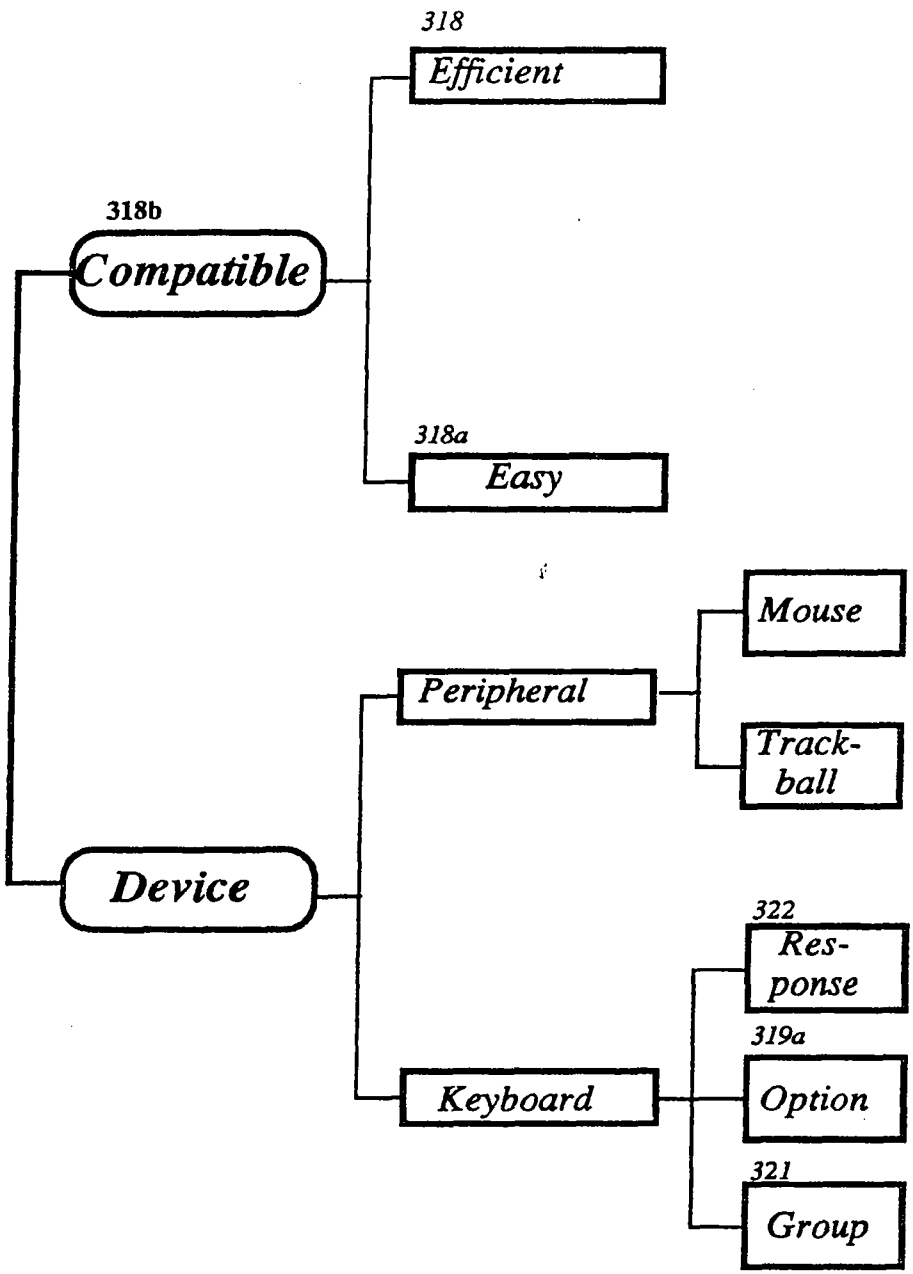




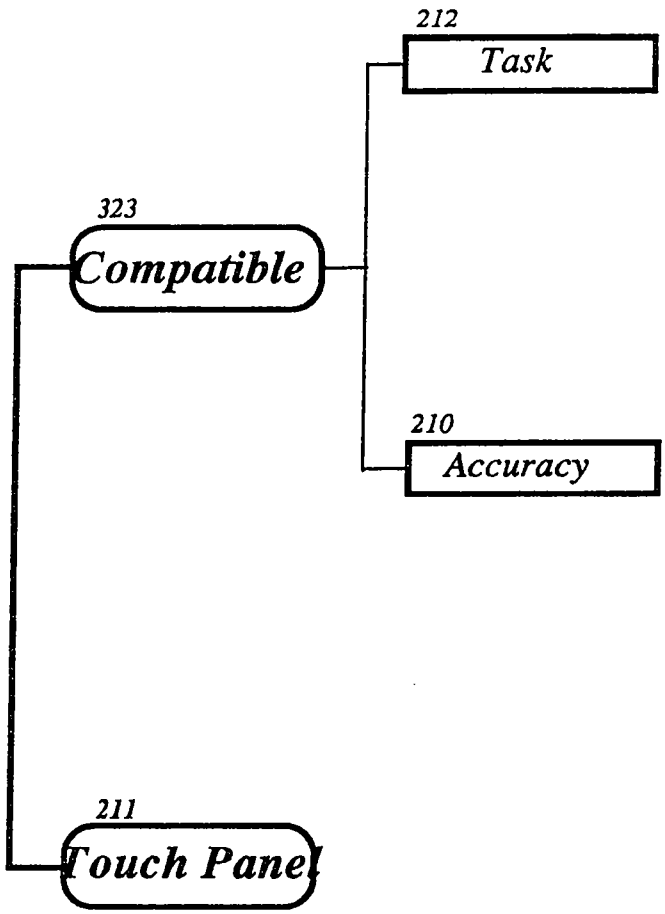
C1.1 Direct Control Methods



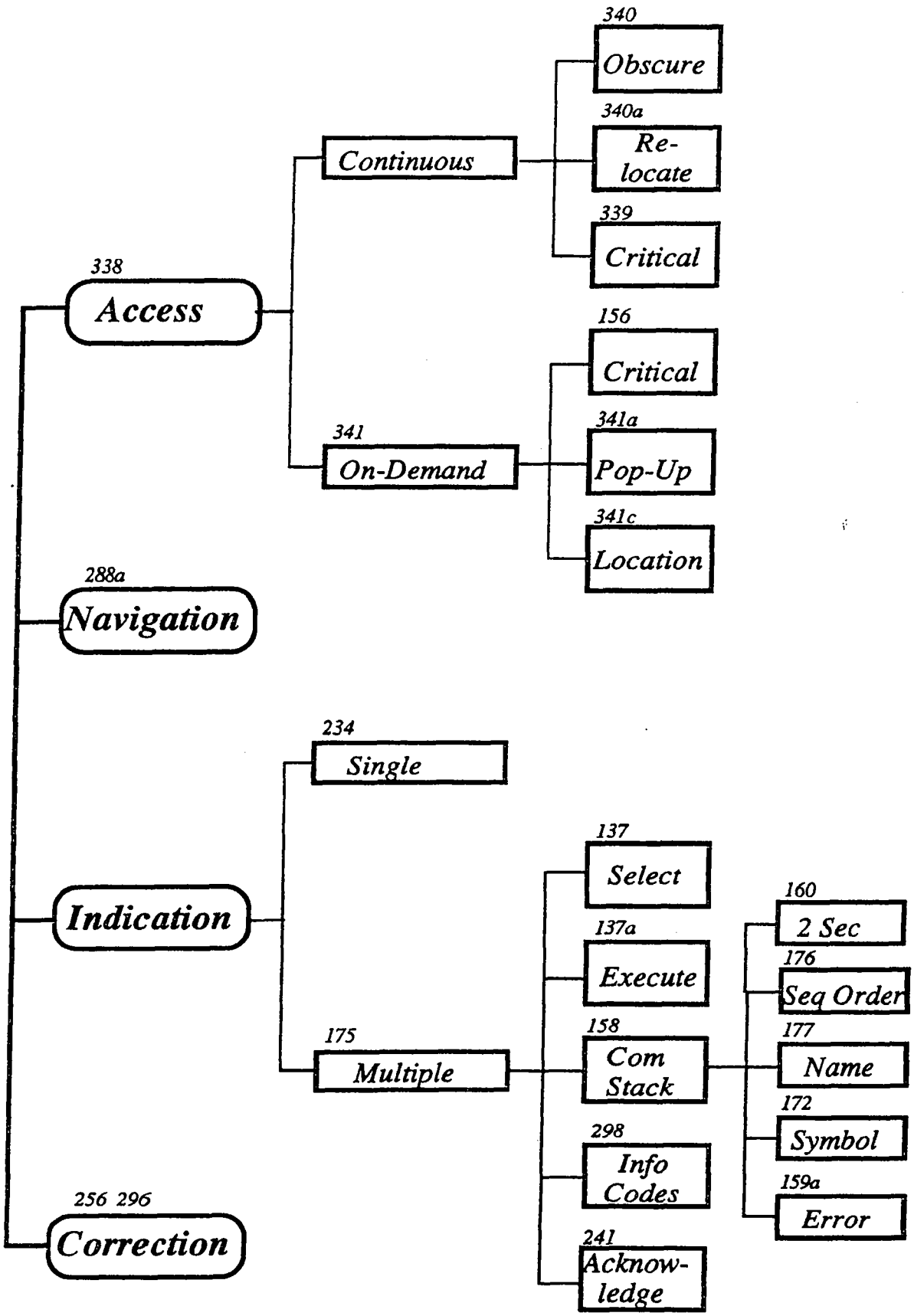
C1.2 Cursor Control Methods



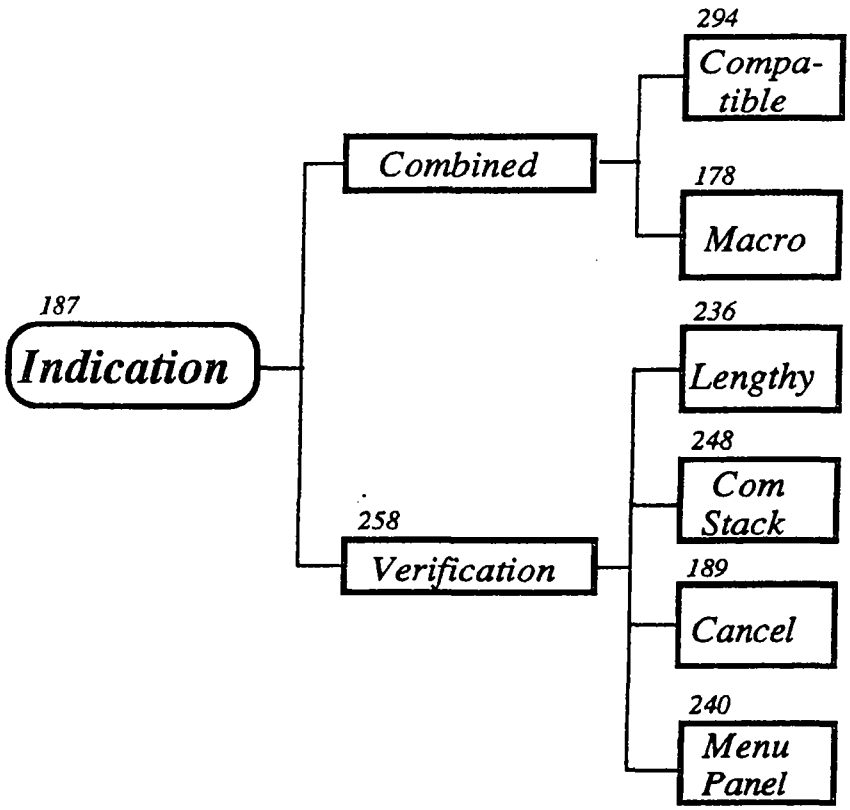
C1.3 Point Control Methods



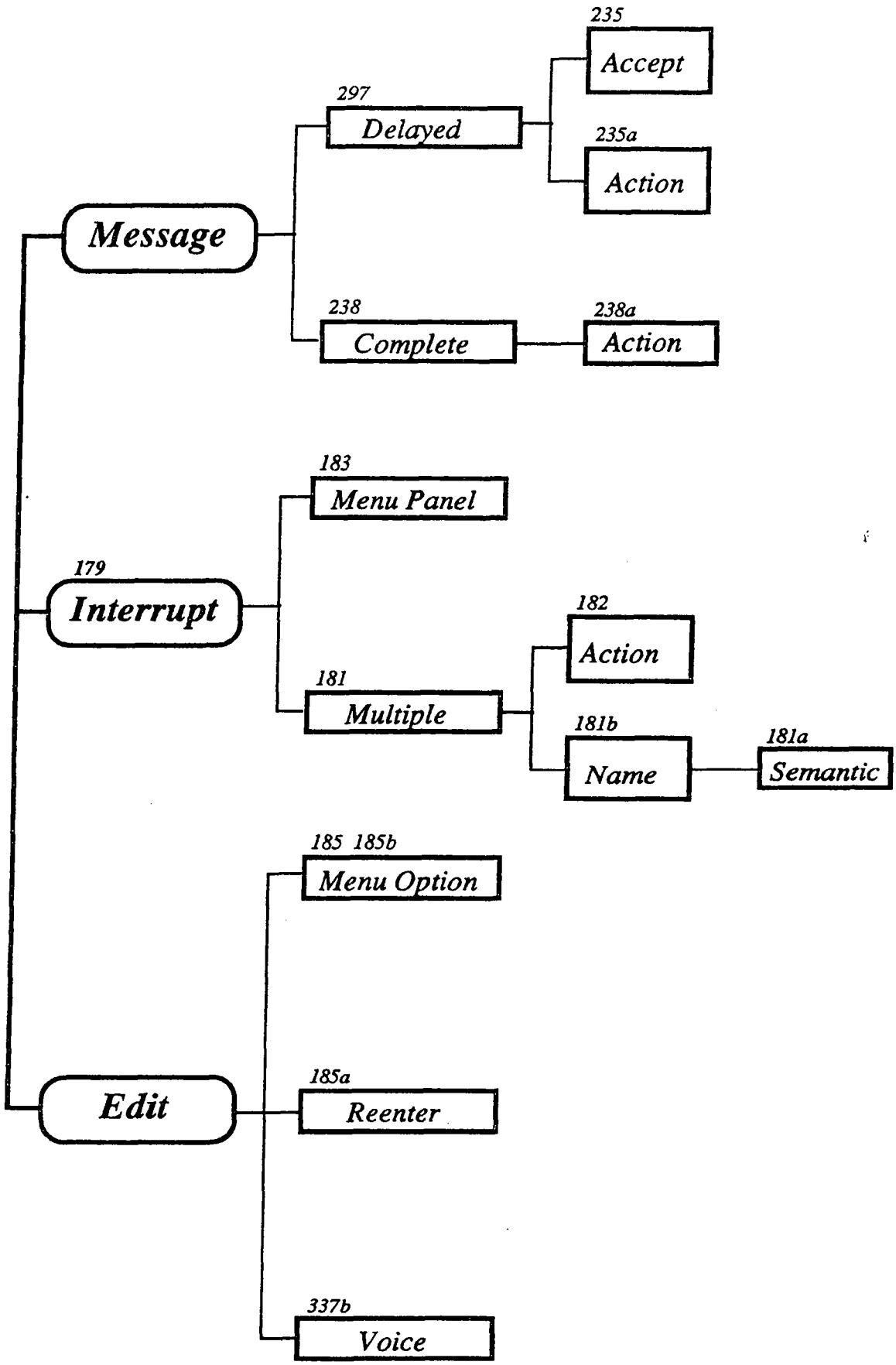
C2 Menu Option Selection



C3 Menu Option Execution



C4 Menu Option Processing



Appendix C

Question Sequence §§

§§ The following appendix presents a sub-set of the full sequence of questions developed for the mock-up system.

A Structure

A1 Menu Appropriateness

Existance :

1. { } Does the software being evaluated contain a menu system as one of its dialogue types ?
   Yes = No Value Vg- Existance
   No = No Value Vg- Existance END MENUS

User:

2. {75, 76, 77} Does the use of menu systems as a dialogue type effectively correspond to the characteristics of the intended users: level of skill, and training ?
   Yes = + compatibility
   No = - compatibility

3. {86, 84} What is the level of user experience with the menu system ?

Inexperienced	= + compatibility	Vg- Experience
Moderately Experienced	= + compatibility	Vg- Experience
Experienced	= No Value	Vg- Experience
Variable or Mixed	= No Value	Vg- Experience

Task:

4. { } Is the principle task routine with fixed procedures ?
   Yes = + compatibility
   No = No Value GOTO 7

5. {85} Does the principle task require minimal entry by the user ?
   Yes = + compatibility
   No = No Value

6. {85a} Does the principle task require the entry arbitrary data by the user ?
   Yes = + compatibility
   No = No Value

7. {83} Does the principle task require a high level of accuracy during initial entry ?
   Yes = + compatibility, + err protection Vg- Accuracy
   No = No Value Vg- Accuracy

8. {109} Does the task require graphical input ?
   Yes = No Value Vg- Graphics
   No = No Value Vg- Graphics

9. {265} Does the task require rapid search times ?
   Yes = No Value Vg- Search time
   No = No Value Vg- Search time

A2 Menu System

Qualities:

1. {338a} Within the menu system is the availability of individual menu options, the category to which they belong, their names, and the means to select them always evident to the user ?
   Yes = + mental load
   No = - mental load

2. {78, 79, 80} *GET A1(3) Experience,* value = { X }
*** Is the menu system flexible ?

IF **Yes,** AND Experience	= Inexperienced	THEN = - flexibility
	= Moderately Experienced	THEN = - flexibility
	= Experienced	THEN = + flexibility
	= Variable or Mixed	THEN = + flexibility
IF **No,** AND IF Experience	= Inexperienced	THEN = + flexibility
	= Moderately Experienced	THEN = + flexibility
	= Experienced	THEN = - flexibility
	= Variable or Mixed	THEN = - flexibility

3. {283} How quickly are the menu panels presented on-screen when accessed ?
   0 - .500 seconds = + imm feedback Vg- Present time
   .600 - 2 seconds = - imm feedback Vg- Present time
   > 2 seconds = - imm feedback Vg- Present time

4. {231} Does the menu system present its information in a directly usable format, such that users are not required to translate, transpose, change units, or interpolate?
   Yes = + mental load
   No = - mental load

Structure:

5. {261a} Does the way in which the menu panels are organized in the menu system facilitate the users' ability to find and select relevant menu options?
   Yes =+ mental load
   No = - mental load

6. {263} Have the menu options in the command set been grouped or ordered within the menu system (i.e., into menu panels) in an unambiguous manner which is easily learned by the user?
   Yes = + compatibility
   No = - compatibility

7. {261} Does the way in which menu options have been grouped into different menu panels reflect user expectations?
   Yes = + compatibility
   No = - compatibility

8. {263} Does the way in which the different menu panels have been organized into a structure minimize depth and maximize breadth?
   Yes = + minimal action
   No = - minimal action

9. {276, 276a} Does the system provide navigational cues to help users learn the menu structure and navigate within that structure (ex., distinctive and compoundable tittles, menu panel designators, option designators, graphic techniques, simultaneous display of menu panels, or menu maps)?
Yes = + mental load
No = - mental load

10. {152} Are multiple navigational paths provided to the users?
Yes = +flexibility Vg- Mult Path GOTO 12
No = No Value Vg- Mult Path

11. {288} Is it logical to access menu panels from differing points in the menu structure?
Yes = - flexibility
No = + flexibility

12. {282} *GET A1(3) Experience*, Value { X }
IF Experience is {experienced, mixed}, THEN Ask Question
IF Experience is {inexperienced, moderate}, THEN SKIP
***Does the menu system provide rapid navigation methods for skilled users?
Yes = + compatibility
No = - compatibility

13. {286} Are users provided with a means to return to the initial menu panel (main menu) from any menu in the menu structure?
Yes = + user control
No = - user control

14. { } Does the menu system utilize the simultaneously display of menu panels?
Yes = No Value
No = No Value GOTO 16

15. {280} When menu panels are simultaneously displayed to the users, is the hierarchical relationship of the menu panels apparent to the users?
Yes = + prompting
No = - prompting

Type:

16. { } Which of the following label best qualifies the type of menu organization present in the menu system?

1. Single	= No Value	Vg- Menu type
2. Linear	= No Value	Vg- Menu type
3. Hierarchical	= No Value	Vg- Menu type
4. Networked	= No Value	Vg- Menu type

17. {131} GET A2 (16) Menu Type, value { X }
IF Menu Type is {hierarchical}, THEN Ask Question
IF Menu Type is { any other }, THEN GOTO A3 (1)
***Does the hierarchical menu structure provide users with some form of indication of their current position in the menu tree?
Yes = + prompting
No = - prompting

18. {127} Has the hierarchical structure been designed to shorten the navigational paths between appropriate menu panels?
Yes = + concision
No = - concision

19. {153} Has the sequence of menu panels been determined by a users task analysis (i.e., does the structure of the menu panels seem to correspond directly to the task at hand?) ?
Yes = + compatibility
No = - compatibility

20. {284} Are users provided with the ability to skip between menu panel nodes without returning to the initial common node?
Yes = + user control
No = - user control

21. {285} Are users provided with the ability to skip intermediate menu panels within the same branch of the hierarchy?
Yes = + user control
No = - user control

22. {287} Are users provided with a simple one keystroke method to move up to the next higher level in the menu structure?
Yes = + minimal action
No = - minimal action GOTO A3 (1)

23. {287a} Is the method provide to users to move up to the next higher level in the menu structure consistent throughout the menu system?
Yes = + consistency
No = - consistency

B Menu Panel Objects

Symetric Balance:

1. {73} Are the menu panels symmetrically balanced?
   Yes= +clarity
   No = -clarity

2. {59} Are the menu panels organized in a standardized format?
   Yes= +format
   No = -format

3. {33} Do there exits users' standard formats for the menu panel layout?
   Yes= No Value
   No = No Value GOTO 5

4. {33} Are the users' standard format of the menu panel layout used?
   Yes= +compatibility
   No = -compatibility

Location:

5. {138} Do the menu panels appear consistently in the same display location?
   Yes= +consistency
   No = -consistency

6. {138a} Do menu panels appear simultaneously in different display locations?
   Yes= +mental load
   No = -mental load

B1 Menu Panel Titles

1. {} Do the menu panels contain menu panel tittles?
   Yes = +mental load
   No = -mental load GOTO B2 (1)

2. {117} What types of menu panel titles appear in the system *(there may exist more than one type of titles, chose as many as apply)* ?

Main titles	= + mental load	Vg- Title Type
Group labels	= No Value	Vg- Title Type
MP Designators	= No Value	Vg- Title Type

3. {61b} Do the menu panel tittles present extra or irrelevant information?
   Yes = +concision
   No = -concision

4. {113a} Do the menu panel titles use consistent terminology throughout the system?
   Yes = +consistency
   No = -consistency

5. {31} Do the menu panel titles use user jargon?
   Yes = -sig of codes
   No = +sig of codes

6. {} Is unnecessary punctuation used in the menu panel tittles?
   Yes = -clarity GOTO 9
   No = +clarity GOTO 9
   Unsure = No Value GOTO 7

7. {44} Are contractions used in the menu panel tittles?
   Yes = -clarity
   No = +clarity

8. {55a} Is hyphenation used in the menu panel tittles?
   Yes = -clarity
   No = +clarity

B1.1 Main Titles

10. {117a} *GET B1 (2) Title Type,* value { X }
    IF Title Type is { main title }, THEN Ask Question
    IF Title Type is {NOT main title}, THEN - mental load
    AND GOTO B1.2 (18)

***Do the main titles clearly state the contents of the menu panel?
   Yes = +prompting
   No = -prompting

11. {131} Do the main titles provide information about the menu systems structure to the user?
    Yes = +prompting
    No = -prompting GOTO 14
    Unknown = No Value

12. {131b} Do the main tittles utilize semantics to indicate current position in structure?
    Yes = +prompting
    No = No Value

13. {131b1} Do the main tittles utilize information coding to indicate current position in structure?
    Yes = +sig of codes
    No = No value

14. {277} Are the main tittles semanticaly distinctive from one another?
    Yes = +sig of codes
    No = -sig of codes

15. {277a} Are the main tittles compoundable, such that they can be put into multiple word titles for lower menu panels? (ex. of semantics)
    Yes = +sig of codes
    No = -sig of codes

16. {73a} Are the main titles centered above the menu panel list?
    Yes = +location
    No = -location

17. {29} Are the main titles presented in all CAPITAL letters?
    Yes = +format
    No = -format

B1.2 Group Labels

18. {130} *GET B1 (2) Title Type,* value { X }
    IF Title Type is { group lables }, THEN Ask Question
    IF Title Type is {NOT group lables}, THEN GOTO B1.3 (22)

***Are group labels provided for each grouping of menu option in the list?
    Yes = +compatibility
    No = -compatibility

19. {352} Do the group label use information coding to make the label perceptually distinct from the menu options they represent?
    Yes = +clarity
    No = -clarity

20. {73b} Are group labels left justified?
    Yes = +location
    No = -location

B1.3 MP Designators

22. {131c} *GET B1 (2) Title Type,* value { X }
    IF Title Type is { mp designators }, THEN Ask Question
    IF Title Type is {NOT mp designators }, THEN B2 (1)

***Do the menu panel designator codes help to indicate current position in the menu structure?
    Yes = +prompting
    No = -prompting

C Sequence Control

1. {186} Are the sequence control actions compatible with task aspects?
   Yes = + compatibility
   No = - compatibility

2. {186a} Are frequent sequence control procedures easy for the users to accomplish?
   Yes = + minimal action
   No = - minimal action

3. {302} Have the number of keystrokes required to execute and select menu options been minimized by the available sequence control features?
   Yes = + concision
   No = - concision

4. {286a} Are the sequence control actions to return to the main menu easy for users to accomplish?
   Yes = + minimal action
   No = - minimal action

5. {186b} Are destructive sequence control procedures difficult for the users to accomplish?

   Yes = + err protection
   No = - err protection

6. {245a} Do common errors in sequence control procedures produce processing sequences that are different from those intended?
   Yes = + err protection
   No = - err protection

7. {188} Are users given the ability to return to the previous step in a sequence control action?
   Yes = + err correction
   No = - err correction GOTO C1

8. {188a} Is returning to the previous step in a sequence control action easy for users to accomplish?
   Yes = + err correction
   No = - err correction

C1 Sequence Control Methods

Alternatives:

2. {290} *GET B2.3 (1)* Seq Methods, value = { X }

IF Seq Methods is {more than one},	THEN + flexibility
IF Seq Methods is {direct },	THEN - flexibility
IF Seq Methods is {cursor },	THEN - flexibility
IF Seq Methods is { point },	THEN - flexibility
IF Seq Methods is { other },	THEN - flexibility

NOTE This is a question that does not need to be asked, but can be determined "Are multiple sequence control methods provided?"

3. {331} *GET B2.3 (1)* Seq Methods, value = { X }

IF Seq Methods is {direct , and point},	THEN + flexibility
IF Seq Methods is {Not direct, and point },	THEN - flexibility

NOTE This is a question that does not need to be asked, but can be determined "Are direct control methods provided in addition to the point control methods?"

4. {324} *GET B2.3 (1)* Seq Methods, value = { X }

IF Seq Methods is {cursor, and Not direct},	THEN + compatibility
IF Seq Methods is { cursor, and direct },	THEN + compatibility
IF Seq Methods is {direct, and Not cursor},	THEN + compatibility

NOTE This is a question that does not need to be asked, but can be determined "Are cursor control methods provided as a substitute to direct control methods?" Yes = + compatibility
No = - compatibility

5. {112} Are the sequence control methods selected compatible with user characteristics?
Yes = + compatibility
No = - compatibility

6. {112a} Are the control methods selected compatible with task characteristics?
Yes = + compatibility
No = - compatibility

7. {291} Are separate user actions required for selection and execution indication?
Yes = + err protection
No = - err protection

8. {289b} Do the sequence control methods selected facilitate the users' operation of the menu system?
Yes = + minimal action
No = - minimal action

9. {289} Are the control methods selected compatible with the dialogue requirements?
Yes = + compatibility
No = - compatibility

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique

ISSN 0249-6399