



CH 7: Command and Natural Languages



The Basic Goals of Language Design

- Precision
- Compactness
- Ease in writing and reading
- Speed in learning
- Simplicity to reduce errors
- Ease of retention over time

➤ Higher-Level Goals of Language Design

- Close correspondence between reality and the notation
- Convenience in carrying out manipulations relevant to user's tasks
- Compatibility with existing notations
- Flexibility to accommodate novice and expert users
- Expressiveness to encourage creativity
- Visual appeal



Functionality to Support User's Tasks

Users do wide range of work:

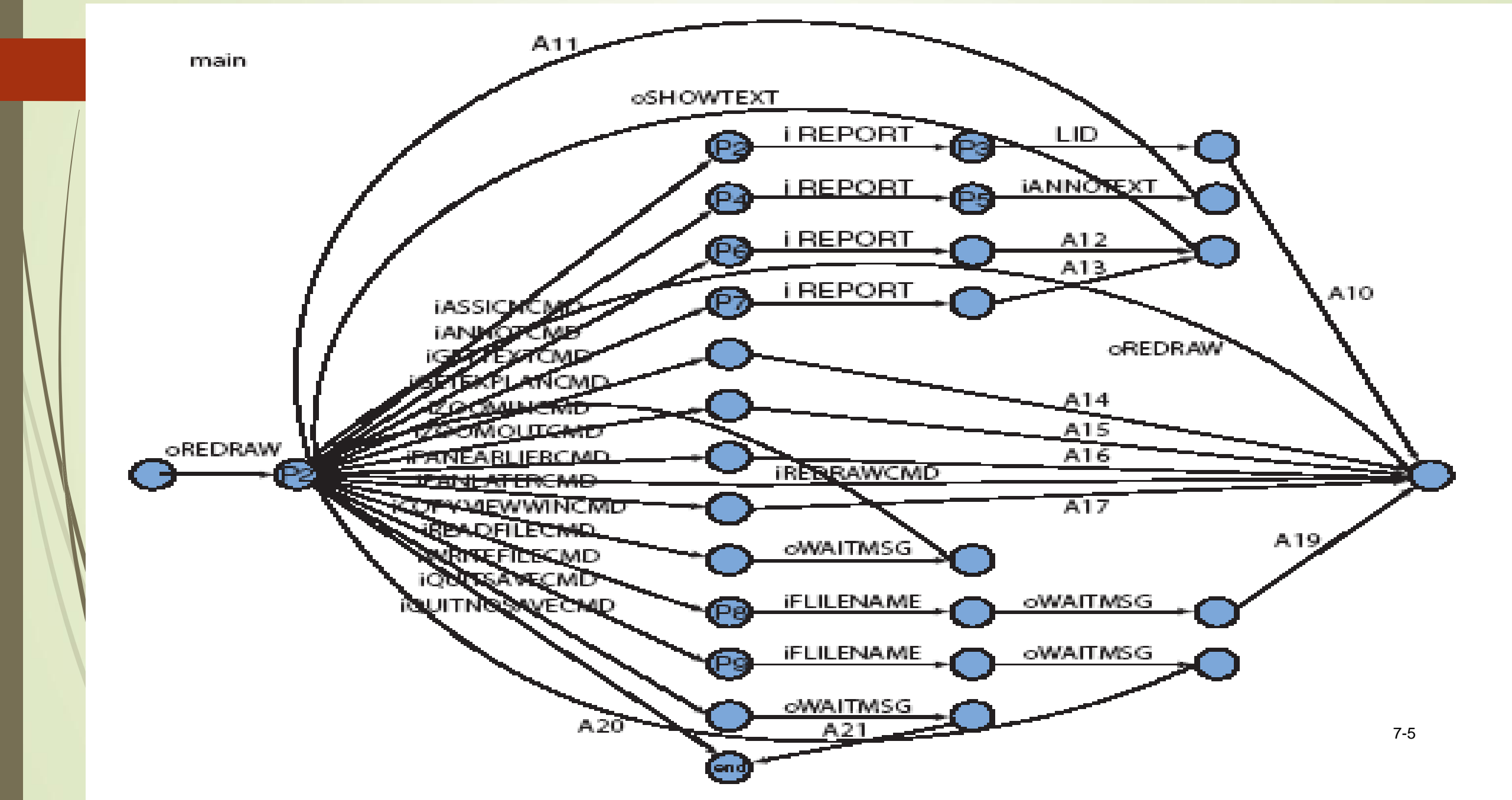
- text editing
- electronic mail
- financial management
- airline or hotel reservations
- inventory
- manufacturing process control
- gaming

Functionality to Support User's Tasks (cont.)

Designers should

- determine functionality of the system by studying users' task domain
- create a list of task actions and objects
- abstract this list into a set of interface actions and objects
- represent low-level interface syntax
- create a table of user communities and tasks, with expected use frequency
- determine hierarchy of importance of user communities (i.e. prime users)
- evaluate destructive actions (e.g. deleting objects) to ensure reversibility
- identify error conditions and prepare error messages
- allow shortcuts for expert users, such as macros and customizing system parameters

- Transition diagram – 'i' with input and 'o' with output





Command-Organization Strategies

A unifying interface concept or metaphor aids

- learning
- problem solving
- retention

Designers often err by choosing a metaphor closer to machine domain than to the user's task domain.

Simple command set

- i. Each command is chosen to carry out a single task. The number of commands match the number of tasks.
- For small number of tasks, this can produce a system easy to learn and use.
- E.g. the vi editor of Unix.

Command plus arguments/options

Command plus arguments

- (ii) Follow each command by one or more arguments that indicate objects to be manipulated, e.g.
 - COPY FILE A, FILE B
 - DELETE FILE A
 - PRINT FILE A, FILE B, FILE C
- Keyword labels for arguments are helpful for some users, e.g. COPY FROM = FILE A TO = FILE B.
- Commands may also have options to indicate special cases, e.g.:
 - PRINT/3,HQ FILE A
 - PRINT (3, HQ) FILE A
 - PRINT FILE A -3, HQ ---- to produce 3 copies of FILE A on the printer in the headquarters building.

The arguments may also have version number, privacy keys or disk address

- Error rates and the need for extensive training increase with the number of possible options.

(iii) Command is organised into a tree structure - Hierarchical command structure can be for a set of tasks

- The full set of commands is organized into a tree structure
- $5 \times 3 \times 4 = 60$ tasks with 5 command names and 1 rule of formation

Action	Object	Destination
CREATE	File	File
DISPLAY	Process	Local printer
REMOVE	Directory	Screen
COPY		Remote printer
MOVE		

The Benefits of Structure

Human learning, problem solving, and memory are greatly facilitated by meaningful structure.

Users can recognize the structure and can easily encode

- Beneficial for
 - task concepts
 - computer concepts
 - syntactic details of command languages

Consistent Argument Ordering

Inconsistent order of arguments

SEARCH	file no, message id
TRIM	message id, segment size
REPLACE	message id, code no
INVERT	group size, message id

Consistent order of arguments

SEARCH	message id, file no
TRIM	message id, segment size
REPLACE	message id, code no
INVERT	message id, group size

Symbols versus Keywords

Command structure affects performance

Symbol Editor

FIND:/TOOTH/;-1

LIST;10

RS:/KO/,/OK/*

Keyword Editor

BACKWARD TO "TOOTH"

LIST 10 LINES

CHANGE ALL "KO" TO "OK"

	Percentage of Task Completed		Percentage of Erroneous Commands	
	Symbol	Keyword	Symbol	Keyword
Inexperienced users	28	42	19.0	11.0
Familiar users	43	62	18.0	6.4
Experienced users	74	84	9.9	5.6

Naming and Abbreviations

There is often a lack of consistency or obvious strategy for construction of command abbreviations.

Specificity Versus Generality

Infrequent, discriminating words

Frequent, discriminating words

Infrequent, nondiscriminating words

Frequent, non discriminating words

General words (frequent, nondiscriminating)

Nondiscriminating nonwords (nonsense)

Discriminating nonwords (icons)

insert

delete

add

remove

amble

perceive

walk

view

alter


correct

GAC

MIK

abc-adbc

abc-ab



LOL	Laugh out loud
2G2BT	Too good to be true
BBFN	Bye bye for now
CUL8R	See you later
HAGD	Have a great day
IMHO	In my humble opinion
J/K	Just kidding
AATK	Always at the keyboard
OTO	Out of the office
POV	Point of view
ROTGL	Rolling on the ground laughing
RTSM	Read the silly manual
SWIM	See what I mean?

TABLE 7.1

Sample abbreviations used in online chat, instant messaging, e-mail, blogs, or newsgroup postings (both at work and at home)



Six Potential Abbreviation Strategies

1. **Simple truncation:** The first, second, third, etc. letters of each command.
2. **Vowel drop with simple truncation:** Eliminate vowels and use some of what remains.
3. **First and last letter:** Since the first and last letters are highly visible, use them.
4. **First letter of each word in a phrase:** Use with a hierarchical design plan.
5. **Standard abbreviations from other contexts:** Use familiar abbreviations.
6. **Phonics:** Focus attention on the sound.

Guidelines for using abbreviations

Ehrenreich and Porcu (1982) offer this set of guidelines:

- A *simple* primary rule should be used to generate abbreviations for most items; a *simple* secondary rule should be used for those items where there is a conflict.
- Abbreviations generated by the secondary rule should have a marker (for example, an asterisk) incorporated in them.
- The number of words abbreviated by the secondary rule should be kept to a minimum.
- Users should be familiar with the rules used to generate abbreviations.
- Truncation should be used because it is an easy rule for users to comprehend and remember. However, when it produces a large number of identical abbreviations for different words, adjustments must be found.
- Fixed-length abbreviations should be used in preference to variable-length ones.
- Abbreviations should not be designed to incorporate endings (ING, ED, S).
- Unless there is a critical space problem, abbreviations should not be used in messages generated by the computer and read by the user.



Command-language guidelines

- Create explicit model of objects and actions.
- Choose meaningful, specific, distinctive names.
- Try to achieve hierarchical structure.
- Provide consistent structure (hierarchy, argument order, action-object).
- Support consistent abbreviation rules (prefer truncation to one letter).
- Offer frequent users the ability to create macros.
- Consider command menus on high-speed displays.
- Limit the number of commands and ways of accomplishing a task.



Natural Language in Computing

- **Natural-language interaction**
- **Natural-language queries and question answering**
- **Text-database searching**
- **Natural-language text generation**
- **Adventure games and instructional systems**

Natural Language in Education

The screenshot displays the CognitiveTutor interface with four main windows:

- Scenario Window:** Contains a word problem about a rock climber and four questions. The problem states: "A rock climber is currently on the side of a cliff 67 feet off the ground. She can climb on average about two and one-half feet per minute." The questions are:
 - When will she be 92 feet off the ground?
 - In twenty minutes, how many feet above the ground will she be?
 - In 75 seconds, how far above the ground will she be?
 - Ten minutes ago, how far above the ground would she have been?Below the questions, it says: "To write the expression, define a variable for the climbing time and use this variable to write a rule for her height above the ground."
- Solver Window:** Shows the equation $92 = 67 + 2.5T$ and the steps to solve for T, resulting in $25 = 2.5T$.
- Skills Window:** Lists skills being tracked, including "Entering a given", "Identifying units", "Finding X, any form", "Writing expression, any form", "Placing points", "Changing axis intervals", and "Changing axis bounds".
- Grapher Window:** Displays a graph of Height (Feet) vs. Time (Minutes). The x-axis ranges from -10 to 20, and the y-axis ranges from 0 to 125. A point is plotted at (10, 25).

Below the Scenario window is a **Worksheet for Problem BH1T20** window, which contains a table for tracking student progress:

Quantity Name	Unit	Expression
TIME	MINUTES	T
Question 1	10	
Question 2	20	
Question 3	75	
Question 4		

A tooltip for the value 75 is shown, stating: "75 is the the climbing time in seconds, but we want to use minutes. How many minutes are in 75 seconds?"

CognitiveTutor traces student progress in mastering skills and concepts, then assigns individually tuned problems
Communicating with students via Natural Language