HCI: Model-based Design (GOMS Family of Models)

Professor Ram Mohana Reddy Guddeti
Information Technology Department
NITK Surathkal, Mangalore, India

Learning Objective

- In the previous lectures, we learned the process involved in interactive system design
- We learned that interactive systems are designed following the Interactive System Design Life Cycle
  - Consisting of the stages for requirement identification, design, prototyping and evaluation
  - Highly iterative (iterative life cycle is time consuming and also requires money (for coding and testing))
- It is always good if we have an alternative method that reduces the time and effort required for the design life cycle
- Model-based design provides one such alternative

Learning Objective

- In this lecture, we will learn the model-based design in HCI
- In particular, we will learn the following
  - Motivation for model-based design approach
  - The idea of models
  - Types of models used in HCI

Idea of a Model

- A ‘model’ in HCI refers to “a representation of the user’s interaction behavior under certain assumptions”

- The representation is typically obtained from empirical studies (collecting and analyzing data from end users)
  - The model represents behavior of average users, not individuals

Motivation

- Suppose we are trying to design an interactive system
- First, let us identify requirements ("know the user") using the methods such as contextual inquiry
  - Time consuming and tedious process
- Instead of going through the process, it would have been better if we have a "model of the user"
- By encompassing information about user behaviour, a model helps in alleviating the need for requirement identification process
- Such requirements are already known from the model
- Once the requirements are identified, designer 'propose' design(s)

Motivation Contd…

- Typically, more than one designs are proposed
  - The competing designs need to be evaluated
- This can be done by evaluating either prototypes (in the early design phase) or the full system (at the final stages of the design) with end users
  - End user evaluation is a must in user centered design
- Like requirement identification stage, the continuous evaluation with end users is also money and time consuming
- If we have a model of end users as before, we can employ the model to evaluate design
  - Because the model already captures the end user characteristics, no need to go for real users

Summary

- A model is assumed to capture behavior of an average user of interactive system
- User behavior and responses are what we are interested in knowing during ISLC
- Thus by using models, we can fulfill the key requirement of interactive computing system design (without actually going to the user)
  - Saves lots of time, effort and money

Types of Models

- For the purpose of this lecture, we shall discuss two broad categories of the models used in HCI

➢ Descriptive/Prescriptive Models: Some models in HCI are used to explain/describe user behavior during interaction in qualitative terms. An example is the Norman's model of interaction (to be discussed later). These models help in formulating (prescribing) guidelines for interface design

➢ Predictive Engineering Models: these models can "predict" behaviour of a user in quantitative terms. An example is the GOMS model (to be discussed later in this module), which can predict the task completion time of an average user for a given system. We can actually "compute" such behaviour.

Predictive Engineering Models

- The predictive engineering models used in HCI are of three types
  - Formal (System) Models
  - Cognitive (User) Models
  - Syndetic (Hybrid) Model

Formal (System) Model

- In these models, the interactive system (interface and interaction) is represented using ‘formal specifications’
  - For example, the interaction modeling using state transition networks
- Essentially models of the ‘external aspects’ of interactive system (what is seen from outside)

Formal (System) Model

- Interaction is assumed to be a transition between states in a ‘system state space’
  - A ‘system state’ is characterized by the state of the interface (what the user sees)
- It is assumed that certain state transitions increase usability while the others do not
- The models try to predict if the proposed design allows the users to make usability-enhancing transitions
  - By applying ‘reasoning’ (manually or using tools) on the formal specification.

Cognitive (User) Models

- These models capture the user's thought (cognitive) process during interaction
  - For example, a GOMS model tells us the series of cognitive steps involved in typing a word
- Essentially models are the 'internal aspects' of interaction (what goes on inside user's mind)
- Usability is assumed to depend on the 'complexity' of the thought process (cognitive activities)
  - Higher complexity implies less usability

Cognitive (User) Models

- Cognitive activities involved in interacting with a system is assumed to be composed of a series of steps (serial or parallel)
  - More the number of steps (or more the amount of parallelism involved), the more complex the cognitive activities are
- These models try to predict the number of cognitive steps involved in executing the 'representative' tasks with the proposed designs
  - Which leads to an estimation of usability of the proposed design

Syndetic (Hybrid) Model

- HCI literature mentions one more type of model, called ‘Syndetic’ model
- In this model, both the system (external aspect) and the cognitive activities (internal aspect) are combined and represented using formal specification
- This model is rather complex and rarely used –, hence it is outside the scope of this HCI course.

Cognitive Models in HCI

- Although we said before that cognitive models are models of human thinking process, they are not exactly treated as the same in HCI

- Since interaction is involved, cognitive models in HCI not only model human cognition (thinking) alone, but the perception and motor actions also (as interaction requires ‘perceiving what is in front’ and ‘acting’ after decision making)

Cognitive Models in HCI

- Thus cognitive models in HCI should be considered as the models of human perception (perceiving the surrounding), cognition (thinking in the 'mind') and motor action (result of thinking such as hand movement, eye movement etc.)
- In HCI, broadly three different approaches are used to model cognition
  - Simple models of human information processing
  - Individual models of human factors
  - Integrated cognitive architectures

Simple Models of Human Information Processing

- These are the earliest cognitive models used in HCI
- These model complex cognition as a series of simple (primitive/atomic) cognitive steps
  - Most well-known and widely used models based on this approach is the GOMS family of models
- Due to its nature, application of such models to identify usability issues is also known as the "Cognitive Task Analysis (CTA)"

Individual Models of Human Factors

- In this approach, individual human factors such as manual (motor) movement, eye movement, decision time in the presence of visual stimuli etc. are modeled
  - These models are analytical expressions to compute task execution times in terms of interface and cognitive parameters
- Examples are: the Fitts' Law, the Hick-Hyman Law

Integrated Cognitive Architectures

- In this approach, the whole human cognition process (including perception and motor actions) is modelled
  - These models capture the complex interaction between different components of the cognitive mechanism unlike the first approach
  - Combines all human factors in a single model unlike the second approach
- Examples are MHP, ACT-R/PM, SOAR

Model-based Design Limitations

- As we mentioned before, model-based design reduce the need for real users in ISLC

- However, they can not completely eliminate the role played by the real users

- We still need to evaluate the designs with real users, albeit during the final stages
  - Model-based design can be employed in the initial stages

Model-based Design Limitations

- The following are the key limitations:
  - The present models are not complete in representing average end user (they are very crude approximations only)
  - The models can not capture individual user's characteristics (only models the average user behavior)

HCI:Model-based Design (GOMS Family of Models)

Professor Ram Mohana Reddy Guddeti
Information Technology Department
NITK Surathkal, Mangalore, India

Learning Objective

- Earlier, we learned the idea of model-based design in HCI
- We also discussed the type of models used in HCI
  - The concepts of prescriptive and predictive models
  - Different types of predictive engineering models
- Here, we will be dealing with a type of predictive engineering models known as "simple model of human information processing"
- GOMS family of models is the best known examples of the above type of predictive engineering model used in the "Interface Design"
  - GOMS stands for **G**oals, **O**perators, **M**ethods and **S**election Rules

GOMS Family of Models

- GOMS is a modeling technique (more specifically, a family of modeling techniques) that analyzes the user complexity of interactive systems design. It is used by software designers to model the user behaviour. The user's behaviour is modelled in terms of Goals, Operators, Methods and Selection rules
- A GOMS model consists of *Methods that are used to achieve* ***G****oals.*
- A *Method is a sequential list of* ***O****perators that the user performs and (sub)Goals that must be* achieved
- If there is more than one ***M****ethod which may be employed to achieve a Goal, a* ***S****election rule is invoked to determine what Method to choose, depending on the context.*

GOMS Family of Models

- **What is GOMS?**
  - Description of the knowledge that a user must have to carry out tasks on a device or system
  - Representation of the "how to do it" knowledge that is required by a system in order to get the intended tasks accomplished.
- **What does a GOMS task analysis involve?**
  - Involves defining and then describing the user's
    - Goals:
      - Something that the user tries to accomplish (action-object pair, e.g. delete word)
      - Include context
    - Methods:
      - Well learned sequence of steps that accomplish a task
      - How do you do it on this system? (could be long and tedious…)
    - Selection Rules:
      - Only when there are clear multiple methods for the same goal.
    - Operators:
      - Elementary perceptual, cognitive and motor acts that cause change (external vs. mental)
      - Also uses action-object pair (e.g. press key, select menu, make gesture, speak command...)
      - mostly defined by hardware and lower-level software.

GOMS: An Example

- **File & Directory Operations:**
  - Delete a file, move a file, delete a directory, move a directory.

- **GOMS Analysis – File & Directory Operations:**
  - Method for goal: delete a file.
    - Step 1. drag file to trash.
    - Step 2. Return with goal accomplished.
  - Method for goal: move a file.
    - Step 1. drag file to destination.
    - Step 2. Return with goal accomplished.
  - Method for goal: delete a directory.
    - Step 1. drag directory to trash.
    - Step 2. Return with goal accomplished.
  - Method for goal: move a directory.
    - Step 1. drag directory to destination.
    - Step 2. Return with goal accomplished.

GOMS: An Example

- **GOMS Analysis – File & Directory Operations - A Better Version:**
  - Method for goal: delete an object.
    - Step 1. drag object to trash.
    - Step 2. Return with goal accomplished.
  - Method for goal: move an object.
    - Step 1. drag object to destination.
    - Step 2. Return with goal accomplished.

- **GOMS Analysis – the Drag Operation**
  - Method for goal: drag item to destination.
    - Step 1. Locate icon for item on screen.
    - Step 2. Move cursor to item icon location.
    - Step 3. Hold mouse button down.
    - Step 4. Locate destination icon on screen.
    - Step 5. Move cursor to destination icon.
    - Step 6. Verify the destination icon.
    - Step 7. Release mouse button.
    - Step 8. Return with goal accomplished.

Learning Objective

- The GOMS family consists of FOUR models
  - **K**eystroke **L**evel **M**odel or KLM
  - Original GOMS proposed by **C**ard, **M**oran and **N**ewell, popularly known as (CMN) GOMS
  - **N**atural GOMS **L**anguage or NGOMSL
  - **C**ognitive **P**erceptual **M**otor or (CPM)GOMS [also known as **C**ritical **P**ath **M**ethod GOMS]

Keystroke Level Model (KLM)

- The KLM - GOMS model was proposed way back in 1980 by Card, Moran and Newell; retains its popularity even today
- This is the earliest model to be proposed in the GOMS family (and one of the first predictive models used in HCI)
- The KLM model provides a quantitative tool (like other predictive engineering models)
  - The model allows a designer to 'predict' the time it takes for an average user to execute a task using an interface and interaction method
  - For example, the model can predict how long it takes to close this PPT using the "close" menu option

How KLM Works

- In KLM, it is assumed that any decision-making task is composed of a series of ‘elementary’ cognitive (mental) steps, that are executed in sequence
- These ‘elementary’ steps essentially represent low-level cognitive activities, which can not be decomposed any further
- The method of breaking down a higher-level cognitive activity into a sequence of elementary steps is simple to understand, provides a good level of accuracy and enough flexibility to apply in practical design situations

How KLM Works

- In KLM, we build a model for task execution in terms of operators
  - That is why KLM belongs to the cognitive task analysis (CTA) approach to design
- For this, we need to choose one or more representative task scenarios for the proposed design
- Next, we need to specify the design to the point where keystroke (operator)-level actions can be listed for the specific task scenarios
- Then, we have to figure out the best way to do the task or the way the users will do it
- Next, we have to list the keystroke-level actions and the corresponding physical operators involved in doing the task

How KLM Works

- If necessary, we may have to include operators when the user must wait for the system to respond (as we discussed before, this step may not be ignored most of the times for modern-day computing systems)

- In the listing, we have to insert mental operator M when user has to stop and think (or when the designer feels that the user has to think before taking next action)

- Once we list in proper sequence all the operators involved in executing the task, we have to do the following

  - Look up the standard execution time for each operator
  - Add the execution times of the operators in the list

How KLM Works

- The total of the operator times obtained in the previous step is "the time estimated for an average user to complete the task with the proposed design"

- If there are more than one design, we can estimate the completion time of the same task with the alternative designs
  - The design with least estimated task completion time will be the best

The Idea of Operators

- To understand how the model works, we first have to understand this concept of 'elementary' cognitive steps
- These elementary cognitive steps are known as ***operators***
  - For example, a key press, mouse button press and release etc.
- Each operator takes a pre-determined amount of time to perform
- The operator times are determined from the empirical data (i.e., data collected from several users over a period of time under different experimental conditions)
  - That means, operator times represent average user behaviour (not the exact behaviour of an individual)

The Idea of Operators

- The empirical nature of operator values indicate that, we can predict the behavior of average user with KLM
  - The model can not predict individual traits
- There are seven operators defined, belonging to three broad groups
- There are seven operator defined, belonging to three broad groups
  - Physical (motor) operators
  - Mental operator
  - System response operator

Physical (Motor) Operators

- There are five operators, that represent five elementary motor actions with respect to an interaction

Operator	Description
K	The motor operator representing a key-press
B	The motor operator representing a mouse-button press or release
P	The task of pointing (moving some pointer to a target)
H	Homing or the task of switching hand between mouse and keyboard
D	Drawing a line using mouse (not used much nowadays)

Mental Operator

- Unlike physical operators, the core thinking process is represented by a single operator **M**, known as the “mental operator”
- Any decision-making (thinking) process is modeled by M

System Response Operator

- KLM originally defined an operator **R**, to model the system response time (e.g., the time between a key press and appearance of the corresponding character on the screen)
- When the KLM model was first proposed (1980), R was significant. However, it is no longer used since we are accustomed to almost instantaneous system response, unless we are dealing with some networked system where network delay may be an issue

KLM- GOMS

- **Calculates the task execution time using pre-established keystroke-level primitive operators (**each operator in KLM refers to an elementary cognitive activity that takes a pre-determined amount of time to perform)**.**
- **Seven Types of Operators:**
  - K: to press a key or a button
  - B: to click (press or release) a mouse button
  - P: to point with a mouse to a target on a display
  - H: to home hands on keyboard or other device
  - D: to draw a line segment on a grid
  - M: to mentally prepare to do an action or closely related series of primitive actions.
  - R: to symbolize the system response time during which the user has to wait for the system.
- **Each of these operators has an estimate time or simple approximation function.**
  - Time to execute is empirically defined:
    - $\mathbf{T_{execute} = T_K + T_B + T_P + T_H + T_D + T_M + T_R}$
- **Heuristics for adding (handling) M**

KLM- GOMS: Operator Times

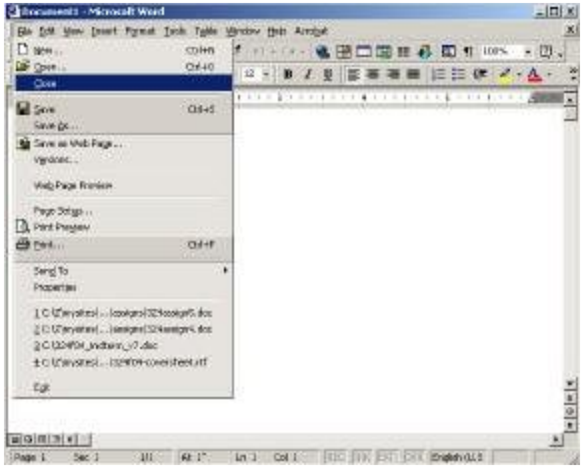
Operator	Description	Time (sec)
K	press a key or button (shift or control key count	
	separately) best typist (135 wpm)	.08
	good typist (90 wpm)	.12
	average typist (55 wpm)	.22
	average typist (40 wpm)	.28
	typing complex codes	.75
B	click (press or release) a mouse button	.10/.20
P	point with mouse to target on display (Fitts's Law)	1.10
H	home hand on keyboard or device	.40
D(n,l)	draw *n* straight-line segments of total length *l* cm	
	(calculated for a square .56 cm grid)	$.9n + .16l$
M	mentally prepare/respond	1.35

KLM: Additional Operator Times

Operator	Description	Time (sec)
	Move eyes to location on screen	2.3
	Retrieve item from memory	12
	Select among methods	12

KLM- GOMS: An Example

- **Closing a Window**
  - **Either use the close button, or press Ctrl+W**

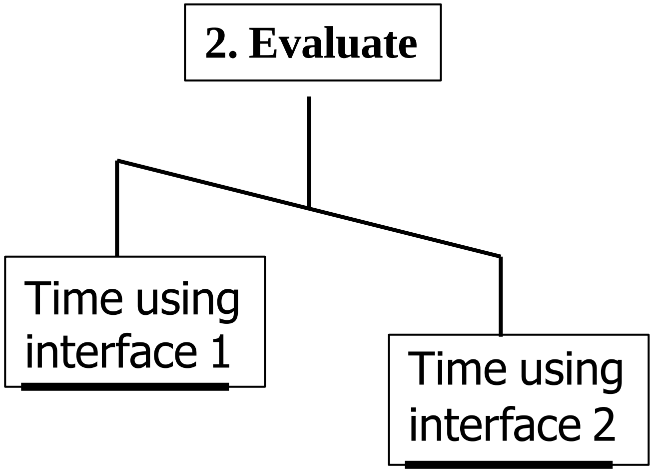


```
GOAL: ICONISE-WINDOW
[select
   GOAL: USE-CLOSE-METHOD
   .      MOVE-MOUSE-TO- FILE-MENU
   .      PULL-DOWN-FILE-MENU
   .      CLICK-OVER-CLOSE-OPTION
   GOAL: USE-CTRL-W-METHOD
          PRESS-CONTROL-W-KEY]
```

  - **Comparing both techniques (assuming hand starts on mouse)**

1. Predict

USE-CTRL-W-METHOD		USE-CLOSE-METHOD	
H[to kbd]	0.40	P[to menu]	1.1
M	1.35	B[LEFT down]	0.1
K[ctrlW key]	0.28	M	1.35
		P[to option]	1.1
		B[LEFT up]	0.1
Total	**2.03 s**	**Total**	**3.75 s**

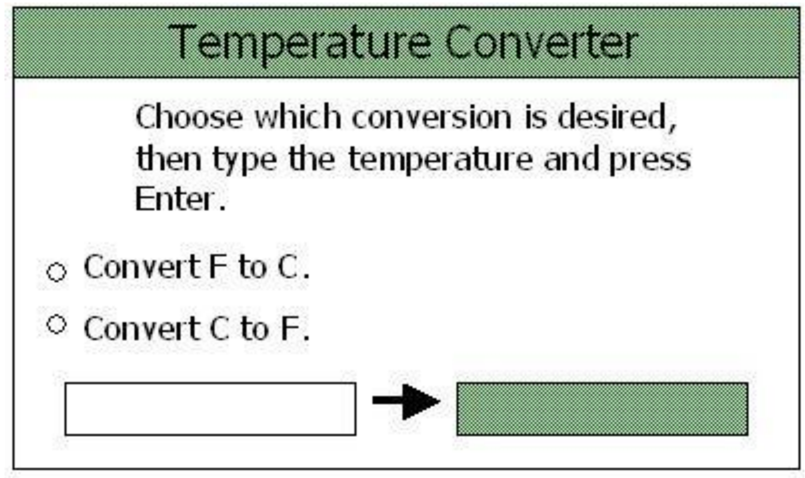


KLM- GOMS: Handling M

- **Rule 0: initial insertion of candidate's M's**
  - **Insert M before K**
  - **Insert M before P iff P selects a command**
- **Rule 1: deletion of anticipated M's**
  - **If an operator following an M is fully anticipated, delete that M**
- **Rule 2: deletion of M's within cognitive units**
  - **If a string of MK's belongs to a cognitive unit, delete all Ms but the first**
- **Rule 3: deletion of M's before consecutive terminators**
  - **If a K is a redundant delimiter, delete the M before it.**
- **Rule 4: deletion of M's that are terminator of commands**
  - **If K is a delimiter that follows a constant string, delete the M infront of it.**
- **Rule 5: deletion of overlapped M's**
  - **Don't count any M that overlaps an R**

KLM: Handling M – An Example

K	0.2
B	.10/.20
P	1.1
H	0.4
D	-
M	1.35
R	-



HPBHKKKKK	Apply Rule 0
HMPMBHMKMKMKMKMK	Apply Rules 1 and 2
HMPBHMKKKKMK	Convert to numbers

.4+1.35+1.1+.20+.4+1.35+4(.2)+1.35+.2

=7.15

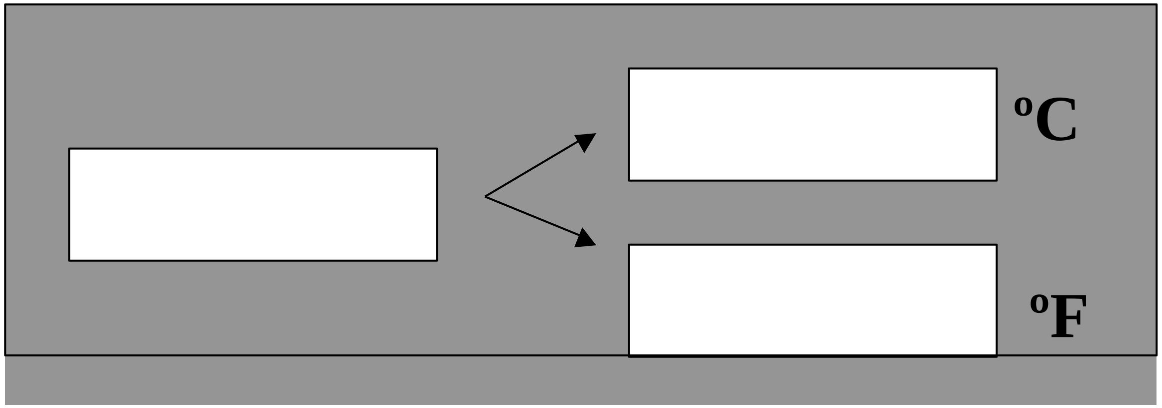
KLM: Handling M – An Example

K	0.2
B	.10/.20
P	1.1
H	0.4
D	-
M	1.35
R	-

To convert temperatures,
Type in the numeric temperature,
Followed by ºC for Celicious or
ºF for Fahrenheit. The converted
Temperature will be displayed.

MKKKKMK = 3.7 sec

KLM: Handling M – An Example



K	0.2
B	.10/.20
P	1.1
H	0.4
D	-
M	1.35
R	-

MKKKK = 2.15 sec

KLM Limitations

- Although KLM provides an easy-to-understand-and-apply predictive tool for interactive system design, it has few significant constraints and limitations
  - It can model only “expert” user behavior
  - User errors can not be modeled
  - Analysis should be done for “representative” tasks; otherwise, the prediction will not be of much use in design. Finding “representative” tasks is not easy

Learning Objective

- In the previous slides, we discussed the KLM
- In KLM, we list the basic (cognitive) steps or operators required to carry out a complex interaction task
  - The listing of operators implies a linear and sequential cognitive behavior
- In this lecture, we shall discuss another model in the GOMS family, referred to as the (CMN)GOMS model
  - CMN stands for **C**ard, **M**oran and **N**ewell (the surname of the three researchers who proposed it)

KLM vs (CMN)GOMS

- In (CMN)GOMS, a hierarchical cognitive (thought) process is assumed, as opposed to the linear thought process of KLM
- Both assumes error-free and ‘logical’ behavior
  - A logical behavior implies that we think logically, rather than driven by emotions

(CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the the task and user actions in terms of four constructs (goals, operators, methods, selection rules)
  - **Goals**: represents what the user wants to achieve, at a higher cognitive level. This is a way to structure a task from cognitive point of view
  - The notion of Goal allows us to model a cognitive process hierarchically

(CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the the task and user actions in terms of four constructs (goals, operators, methods, selection rules)
  - **Operators**: elementary acts that change user's mental (cognitive) state or task environment. This is similar to the operators we have encountered in KLM, but here the concept is more general

(CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the the task and user actions in terms of four constructs (goals, operators, methods, selection rules)
  - **Methods**: these are sets of goal-operator sequences to accomplish a sub-goal

(CMN) GOMS – Basic Idea

- (CMN)GOMS allows us to model the the task and user actions in terms of four constructs (Goals, Operators, Methods, Selection Rules)
  - **Selection Rules**: sometimes there can be more than one method to accomplice a goal. Selection rules provide a mechanism to decide among the methods in a particular context of interaction

Operator in (CMN)GOMS

- As mentioned earlier, the operators in (CMN)GOMS Model are conceptually similar to the operators in KLM
- The major difference is that in KLM, only seven operators are defined. In (CMN)GOMS, the notion of operators is not restricted to those seven operators
  - The modeler has the freedom to define any "elementary" cognitive operation and use it as operator
- The operator can be defined
  - At the keystroke level (as in KLM)
  - At higher levels (for example, the entire cognitive process involved in "closing a file by selecting the close menu option" can be defined as operator)

Operator in (CMN)GOMS

- (CMN)GOMS gives the flexibility of defining operators at any level of cognition and different parts of the model can have operators defined at various levels
- **Example:** Suppose we want to find out the definition of a word from an online dictionary. How can we model this task with (CMN)GOMS?
- **Answer:** We shall list the goals (high level tasks) first

  - Goal: Access online dictionary (first, we need to access the dictionary)
  - Goal: Lookup definition (then, we have to find out the definition)

Answer to the Example (Contd...)

- Next, we have to determine the methods (operator or goal-operator sequence) to achieve each of these goals

  - Goal: Access online dictionary
    - # Operator: Type URL sequence
    - # Operator: Press Enter

- Next, we have to determine the methods (operator or goal-operator sequence) to achieve each of these goals

  - Goal: Lookup definition
    - # Operator: Type word in entry field
    - # Goal: Submit the word
      - Operator: Move cursor from field to Lookup button
      - Operator: Select Lookup
    - # Operator: Read output

Answer to the Example (Contd...)

- Thus, the complete model for the task is
  - Goal: Access online dictionary
    - Operator: Type URL sequence
    - Operator: Press Enter
  - Goal: Lookup definition
    - Operator: Type word in entry field
    - Goal: Submit the word
      - Operator: Move cursor from field to Lookup button
      - Operator: Select Lookup button
    - Operator: Read output

Answer to the Example (Contd...)

- Notice that there is a hierarchical nature of the model
- Note that there are operators of use
  - The operator “type URL sequence” is a high-level operator defined by the modeler
  - “Press Enter” is a keystroke level operator
- Note that how both the low-level and high-level operators co-exist in the same model
- Note that there are methods of use
  - For the first goal, the method consisted of two operators
  - For the second goal, the method consisted of two operators and a sub-goal (which has a two-operators method for itself)

Another Example

- The previous example illustrates the concepts of goals and goal hierarchy, operators and methods
- The other important concept in (CMN)GOMS is the selection rules
- **Example:** Suppose we have a window interface that can be closed in either of the two methods: by selecting the 'close' option from the file menu or by selecting the Ctrl key and the F4 key together. How we can model the task of "closing the window" for this system?

Another Example

- Here, we have the high level goal of “close window” which can be achieved with either of the two methods: “use menu option” and “use Ctrl+F4 keys”
  - This is unlike the previous example where we had only one method for each goal
- We use the “Select” construct to model such situations (please see the next slide)

Another Example

Goal: Close window

- [SelectGoal: Use menu method

    Operator: Move mouse to file menu

    Operator: Pull down file menu

    Operator: Click over close option

  Goal: Use Ctrl+F4 method

    Operator: Press Ctrl and F4 keys together]

Another Example

- The select construct implies that “selection rules” are there to determine a method among the alternatives for a particular usage context
- Example selection rules for the window closing task can be
  - Rule 1: Select “use menu method” unless another rule applies
  - Rule 2: If the application is GAME, then select “use Ctrl+F4 method”
- The rules state that, if the window appears as an interface for a game application, it should be closed using the Ctrl+F4 keys. Otherwise, it should be closed using the close menu option

Steps for Model Construction

- A (CMN)GOMS model for a task is constructed according to the following steps
  - Determine high-level user goals
  - Write method and selection rules (if any) for accomplishing goals
  - This may invoke sub-goals, write methods for sub-goals
  - This is recursive. Stop when operators are reached

Use of the Model

- Like KLM, (CMN)GOMS also makes quantitative prediction about user performance
  - By adding up the operator times, total task execution time can be computed
- However, if the modeler uses operators other than those in KLM, the modeler has to determine the operator times

Use of the Model

- The task completion time can be used to compare competing designs

- In addition to the task completion times, the task hierarchy itself can be used for comparison
  - The deeper the hierarchy (keeping the operators same), the more complex the interface is (since it involves more thinking to operate the interface)

(CMN) GOMS Model Limitations

- Like KLM, (CMN)GOMS also models only skilled (expert) user behavior
  - That means user does not make any errors
- Can not capture the full complexity of human cognition such as learning effect, parallel cognitive activities and emotional behavior