**UNIT - III**

## Eight Golden Rules of User Interface Design stated by Ben Shneiderman.

Shneiderman's eight golden rules provide a convenient and succinct summary of the key principles of interface design. They are intended to be used during design but can also be applied, like Nielsen's heuristics, to the evaluation of systems.

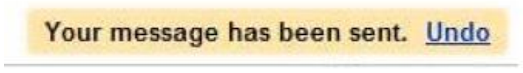
1. **Strive for consistency** by utilizing familiar icons, colors, menu hierarchy, call-to-actions, and user flows when designing similar situations and sequence of actions.
   - Users need to be able to do the same thing the same way that they have been doing.-every time.
   - Interfaces need to exhibit 'consistent' quality across screens/ applications both visually as well as behaviorally.
   - Consistency leads to a pattern which is easier to handle cognitively.
   - Consistency such as 'similar sequence of actions in similar situations' makes it easy to learn.
   - Consistency can be achieved through graphical elements such as fonts, colour, shape, position being consistently same in all menus & screens, across, categories for a particular software.

2. **Enable frequent users to use shortcuts.** With increased use comes the demand for quicker methods of completing tasks. For example, both Windows and Mac provide users with keyboard shortcuts for copying and pasting, so as the user becomes more experienced, they can navigate and operate the user interface more quickly and effortlessly. Such as abbreviations, special key sequences and macros, to perform regular, familiar actions more quickly.

3. **Offer informative feedback** for every user action, at a level appropriate to the magnitude of the action.
   - Interfaces need to not just to be communicative but also need to inform the 'user' in terms of learning & feedback which tells them that they are proceeding in the right direction.
   - For every action of the user there needs to be a feedback –only then 'interaction' (in HCI) is said to take place.
   - Unless the user gets a feedback he/she cannot proceed or becomes unsure of the correctness of the action.

4. **Design dialogs to yield closure** so that the user knows when they have completed a task.
   - In an interaction -dialogue needs to have a closure which is recognized by the user as end of an action.
   - Sequence of actions need to proceed in a dialogue by engaging the user in a step by step manner.
   - Like in a mathematical expression, every enclosing bracket needs a corresponding closing bracket. So also subsequence of actions needs to be grouped with intermittent closing of each sub group followed finally by a closer action of the group.

   Ex: A message at the end of a sequence of events gives a feedback & closure of sending a SMS.



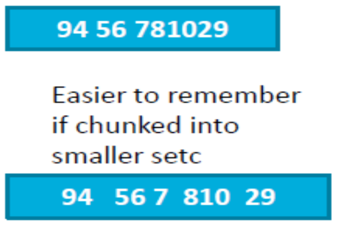
Your message has been sent. Undo

5. **Offer error prevention and simple error handling** so that, ideally, users are prevented from making mistakes and, if they do, they are offered clear and informative instructions to enable them to recover. No one likes to be told they're wrong, especially your users.
   - Systems should be designed to be as fool-proof as possible, but when unavoidable errors occur, ensure users are provided with simple, intuitive step-by-step instructions to solve the problem as quickly and painlessly as possible.
   - For example, flag the text fields where the users forgot to provide input in an online form.

6. **Permit easy reversal of actions** in order to relieve anxiety and encourage exploration, since the user knows that he can always return to the previous state.
   - Designers should aim to offer users obvious ways to reverse their actions. These reversals should be permitted at various points whether it occurs after a single action, a data entry or a whole sequence of actions.
7. **Support internal locus of control** so that the user is in control of the system, which responds to his actions.
   - Allow your users to be the initiators of actions.
   - Give users the sense that they are in full control of events occurring in the digital space. Earn their trust as you design the system to behave as they expect.
   - Give users the power to choose whether to continue running the program or exit from it.

8. **Reduce short-term memory load** by keeping displays simple, consolidating multiple page displays and providing time for learning action sequences.
   - Human attention is limited and we are only capable of maintaining around five items in our short-term memory at one time.
   - Therefore, interfaces should be as simple as possible with proper information hierarchy, and choosing recognition over recall. Recognizing something is always easier than recall because recognition involves perceiving cues that help us reach into our vast memory and allowing relevant information to surface.
   - For example, we often find the format of multiple choice questions easier than short answer questions on a test because it only requires us to recognize the answer rather than recall it from our memory.

Millers 7 chunks of information is often prescribed as a solution to limit short term memory. In psychological experiments it has been found that the short term memory can hold 7 +-2 bits called chunks of information. Long sequential actions requiring more than 7 chunks need to be broken down into smaller chunks.



## Norman's Seven Principles for Transforming Difficult Tasks into Simple Ones

1. **Use both knowledge in the world and knowledge in the head:**

   People work better when the knowledge they need to do a task is available externally – either explicitly or through the constraints imposed by the environment. But experts also need to be able to internalize regular tasks to increase their efficiency. So systems should provide the necessary knowledge within the environment and their operation should be transparent to support the user in building an appropriate mental model of what is going on.

2. **Simplify the structure of tasks:**

   Tasks need to be simple in order to avoid complex problem solving and excessive memory load. Take into account psychology, STM and LTM limitations. There are a number of ways to simplify the structure of tasks.

   - Provide mental aids (keep task the same)
   - Use technology to make visible what would otherwise be invisible, thus providing feedback and the ability to keep control.
   - Automate (but keep task the same). Don't take away control!
   - Change the nature of the task. Example: Hook-and-Loop fastener

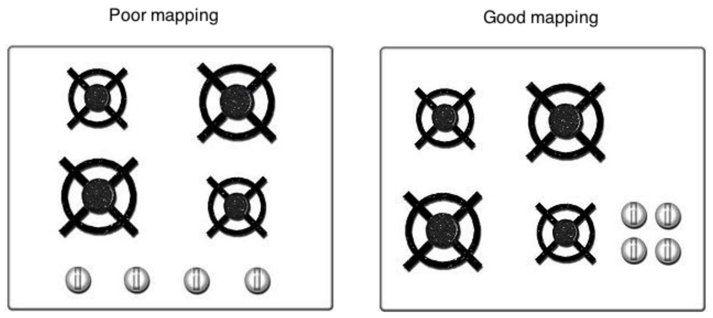
3. **Make things visible:**
   - Make the outcome of an action obvious. The interface should make clear what the system can do and how this is achieved, and should enable the user to see clearly the effect of their actions on the system.
   - The system should provide actions that match intentions. It should provide indications of system state that are readily perceivable and interpretable and that match intentions and expectations.
   - The system state should be visible and readily interpretable.
4. **Get the mappings right.**

Exploit natural mappings. Make sure the user can determine the relationships:

- Between intentions and possible actions
- Between actions and their effect on the system
- Between actual system state and what is perceivable by sight, sound or feel
- Between the perceived system state and the needs, intentions, and expectations of the user

User intentions should map clearly onto system controls. User actions should map clearly onto system events. So it should be clear what does what and by how much. Controls, sliders and dials should reflect the task.

Ex. Gas stove burner



5. **Exploit the power of constraints, both natural and artificial.**
   - Constraints are things in the world that make it impossible to do anything but the correct action in the correct way. Example is a jigsaw puzzle, where the pieces only fit together in one way.
   - This is about how to ensure that the user knows what to do next when there are more than one possibility or more than one given option.
   - In other words how a designer needs to embed constraints in the sequence of operations in an interface such that the user is guided to the right sequence choice by reducing the chance of error in choosing the wrong option

**As a principle Interfaces need to have any of the three type of constraints:**

- **Physical constraints:** Based on shape, size, area (example mouse over area demarcation)
- **Cultural constraints:** Culturally & semantically practiced rituals, symbols, color codes. Ex: Always start from Right & stop on Left lower end in a word document.
- **Technological constraints:** Example: Closing a file without saving –user needs to be warned every time this is likely to be operated by the user.
  Program it such that it is mandatory to press the save button before close button.

6. **Design for error.**

To err is human! Always design with that in mind, so anticipate the errors the user could make and design recovery into the system.

- Understand the causes of error and design to minimize those causes.
- Make it possible to reverse actions (undo) or make it harder to do what cannot be reversed.
- Make it easier to discover the errors that do occur, and make them easier to correct.
- Change the attitude towards errors: Don't think of the user as making errors, think of the actions as approximations of what is desired.
- Errors are not taken as human faults in users in HCI, this means Errors by users cannot be blamed on Users. Users are not the cause for errors.
- Often errors are 'slips' intend to do one thing but end up doing another accidentally.
- Errors happen when there is a mismatch between User's mental model, designers' understanding of User's mental model; system limitations.

**Errors can be classified as:**

- **Description Errors:** Two objects physically alike are described / taken mistakenly for each other. One solution employed is 'highlighting' the object which is in line of next action so that 'attention' is drawn to that right object from amongst similar looking group of objects.

- **Data Errors:** Could be perception errors or selection errors. A solution could be reversal of action without penalty and 'affordance' by the user to correct the error by retracing action steps.

- **Associative Action Errors:** Associative Errors are those that involve activating one sequence in place of another and realizing it when the wrong /unexpected response results. Associative Errors also happen when short term memory is overloaded or long term memory fails. Forgetting to do something as prescribed or reversing the sequence pressing the second button first instead of the first button etc.

7. **When all else fails, standardize.**
If there are no natural mappings then arbitrary mappings should be standardized so that users only have to learn them once. It is this standardization principle that enables drivers to get into a new car and drive it with very little difficulty, key controls are standardized. Occasionally one might switch on the indicator lights instead of the windscreen wipers, but the critical controls (accelerator, brake, clutch, steering) are always the same.

## Norman's model of interaction

**Interaction involves at least two participants: the user and the system**. Both are complex, as we have seen, and are very different from each other in the way that they communicate and view the domain and the task. The interface must therefore effectively translate between them to allow the interaction to be successful. This translation can fail at a number of points and for a number of reasons. **The use of models of interaction can help us to understand exactly what is going on in the interaction and identify the likely root of difficulties.** They also provide us with a framework to compare different interaction styles and to consider interaction problems.

### The terms of Interaction

The purpose of an interactive system is to aid a user in accomplishing *goals* from some application *domain*.

**Domain:** defines an area of expertise and knowledge in some real-world activity. Some examples of domains are graphic design, authoring and process control in a factory. A domain consists of concepts that highlight its important aspects. In a graphic design domain, some of the important concepts are geometric shapes, a drawing surface and a drawing utensil.

**Tasks:** are operations to manipulate the concepts of a domain.

**Goal:** is the desired output from a performed task. For example, one task within the graphic design domain is the construction of a specific geometric shape with particular attributes on the drawing surface. A related goal would be to produce a solid red triangle centered on the canvas.

**An intention:** is a specific action required to meet the goal.

**Task analysis:** involves the identification of the problem space for the user of an interactive system in terms of the domain, goals, intentions and tasks.

The concepts used in the design of the system and the description of the user are separate, and so we can refer to them as distinct components, called the *System* and the *User*, respectively.

The *System* and *User* are each described by means of a language that can express concepts relevant in the domain of the application.
The *System*'s language we will refer to as the *core language* and the *User*'s language we will refer to as the *task language*. The core language describes computational attributes of the domain relevant to the *System* state, whereas the task language describes psychological attributes of the domain relevant to the *User* state.
In HCI interaction models are translations between user and system.

## NORMAN'S MODEL:

Norman proposed a simple model of interaction to explain why users have problems with interfaces. In his model, interaction consists of seven stages all of which are performed by the user:

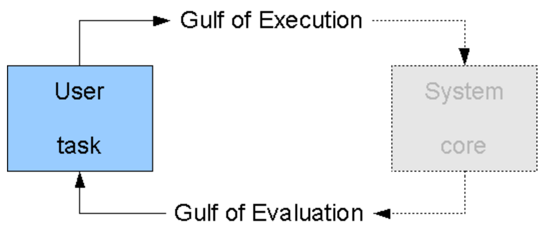
- **execution**
  - establish the goal
  - form the intention
  - specify the action sequence
  - execute the action
- **evaluation**
  - perceive the system state
  - interpret the system state
  - evaluate the system state with respect to the goals and intentions

Norman classifies the problems that arise in his model in terms of two distinct gulfs:

1. the gulf of execution - difference between the user's formulation of an action and the action allowed by the system
2. the gulf of evaluation - difference between the physical representation and the expectations of the user

For Norman, the purpose of an interface is to reduce the gulf of execution.
The gulf of evaluation measures the effectiveness of the interaction. The more effort required to interpret the presentation, the less effective is the interaction.

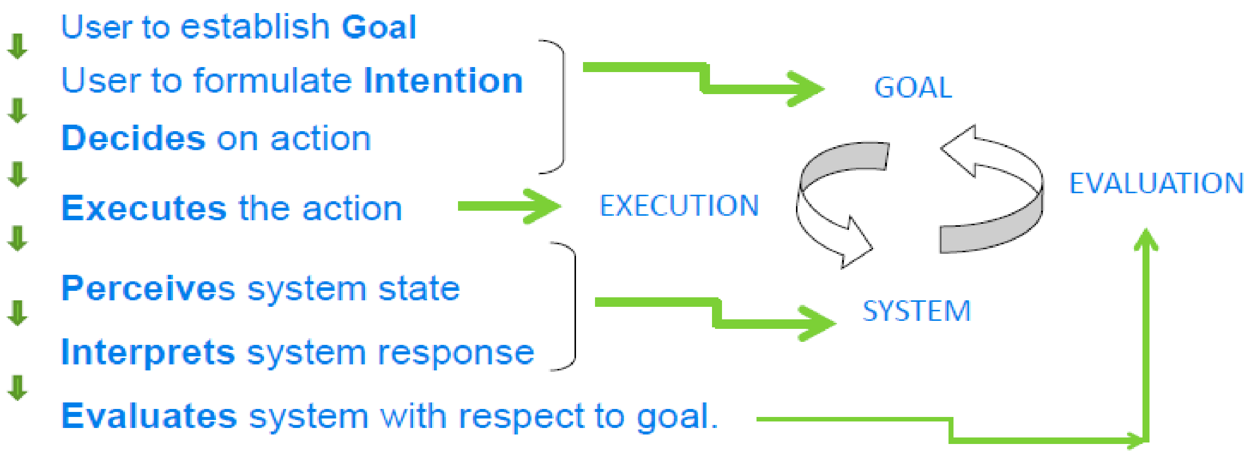
**The execution–evaluation cycle (Norman's model of interaction)**



**Norman's Model**

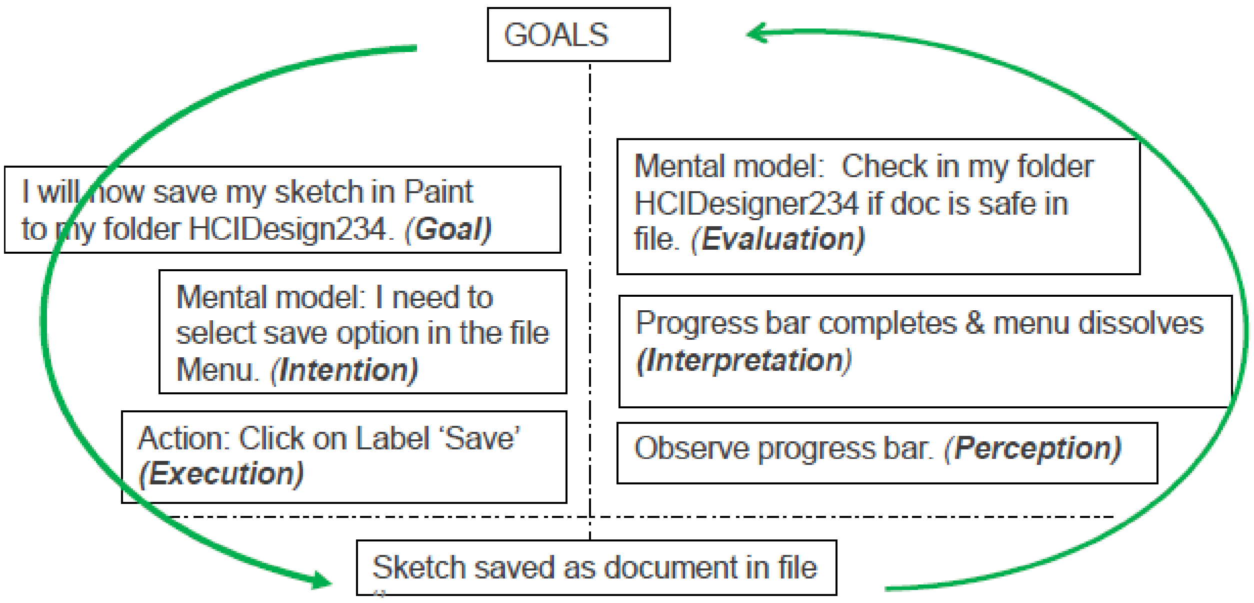
**Norman's Model of Interaction consists of seven stages as follows:**

Donald Norman's Interaction model concentrates on the Users thought processes and accompanying actions.



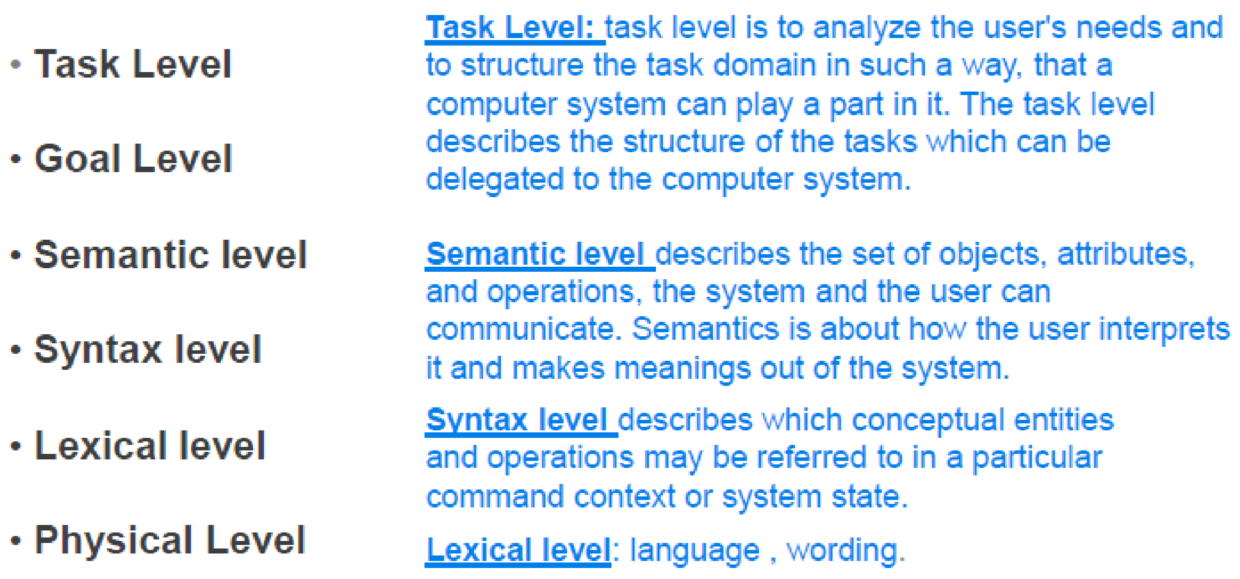
Given a need a user sets about achieving the goal of fulfilling the needs. A series of actions are performed –one leading to another –till the result expected is obtained.

**Understanding Normans Model with Example:**
**Need:** Documenting work done
**Task:** Save My Sketch
**Goal:** Safely store the sketch in a place which I can fetch it from


GOALS
I will now save my sketch in Paint to my folder HCIDesign234. (Goal)
Mental model: I need to select save option in the file Menu. (Intention)
Action: Click on Label 'Save' (Execution)
Mental model: Check in my folder HCIDesigner234 if doc is safe in file. (Evaluation)
Progress bar completes & menu dissolves (Interpretation)
Observe progress bar. (Perception)
Sketch saved as document in file

There are different Interaction Styles (nature of the dialogue) and there are different Interaction Contexts **(Social, Organizational, Educational, Commercial etc.)**

As a basis for his Interaction Model Norman proposed the following levels of abstraction of knowledge of the user:



- Task Level
- Goal Level
- Semantic level
- Syntax level
- Lexical level
- Physical Level

**Task Level:** task level is to analyze the user's needs and to structure the task domain in such a way, that a computer system can play a part in it. The task level describes the structure of the tasks which can be delegated to the computer system.

**Semantic level** describes the set of objects, attributes, and operations, the system and the user can communicate. Semantics is about how the user interprets it and makes meanings out of the system.

**Syntax level** describes which conceptual entities and operations may be referred to in a particular command context or system state.

**Lexical level**: language , wording.

**Norman's HCI model consists of three types:**

1. User's Mental Model
2. System Image Model
3. Conceptual Model.

**The User's Mental Model:** is the model of a machine's working that a user creates when learning and using a computer. It is not technically accurate. It may also be not stable over time. User's mental models keep changing, evolving as learning continues. In a way Mental Models are models people have of themselves, others and environment. The mental model of a device is formed by interpreting its perceived actions and its visible structure.

**The System image Model:** is the visible physical part of the computing system / device.

**The Conceptual Model:** This is the technically accurate model of the computer / device / system created by designers / teachers/researchers for their specific internal technical use. Users too have a Conceptual model but it is their mental model unless the user is a technically qualified as the evaluator.

In a way as far as the user is concerned mental models and conceptual models are inherent to each other. Designer's too have mental models of the system. So a Conceptual model of the system needs to be as close as possible to the System's Image Model. The User model (what the user develops in the self to explain the operation of the system) and the system image (the system's appearance, operation way it responds) is usually a blend of the users mental model and conceptual model all rolled into one.

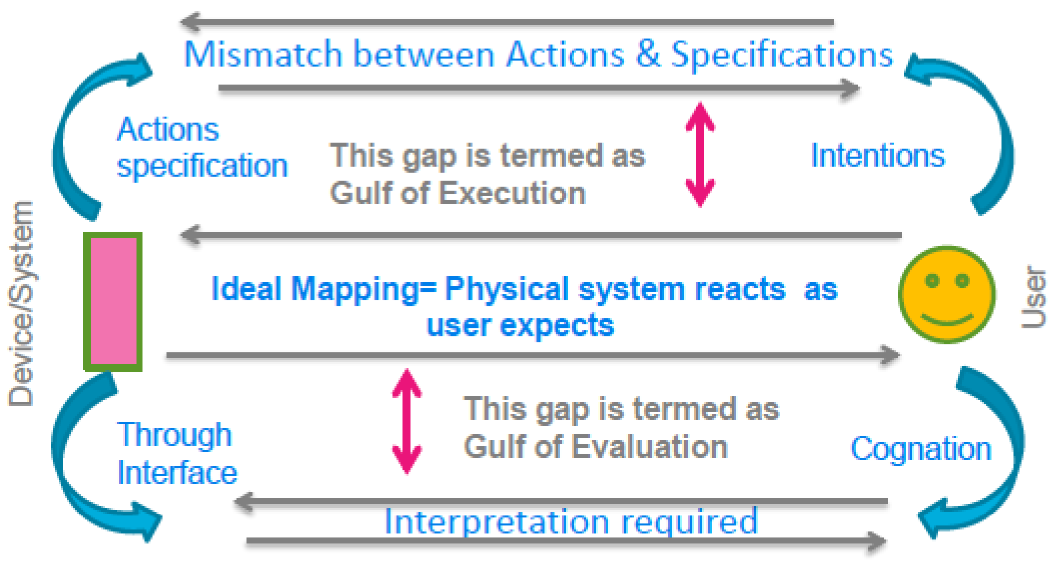
**Interaction Model and device /system Design**
A good device / system will emerge when the starting point of the design process is the user-his/her mental model' (in turn derived through user research-task analysis, walk thoughts Contextual inquiry etc.) being the basis of the system image and its conceptual model
The Conceptualization of the Designer had in his/her mind is called the design model. Ideally, the design model and user model have to be as close as possible for the systems acceptance. The designer must ensure that the system image is consistent with and operates according to the proper conceptual model.
Norman applies the Model to explain why some interfaces cause problems to the users.

**Norman's model (also sometimes called as Gulf Model)** is useful in understanding the reasons of interface failures from the user's point of view. The Seven stages of action model is an elaboration of the Gulf model.
**Gulf of Execution represents the difference between user's formulation of the action to reach their goals and the actions allowed by the system.**

User's formulation of action $\neq$ Actions allowed by the system.

**The Gulf of Evaluation is the difference between physical presentation of system state and the expectations of the user.**

User's Expectation $\neq$ system's presentation.

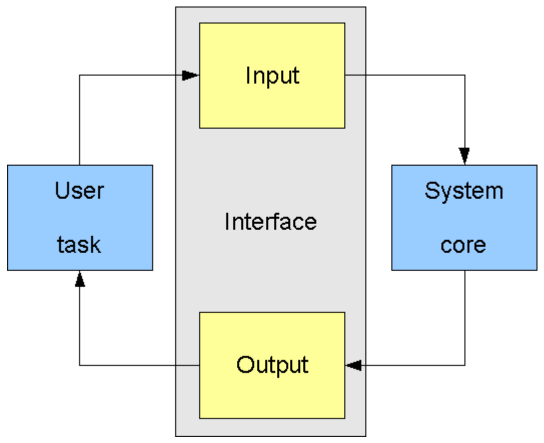

Mismatch between Actions & Specifications
Actions specification
This gap is termed as Gulf of Execution
Intentions
Device/System
Ideal Mapping= Physical system reacts as user expects
User
Through Interface
This gap is termed as Gulf of Evaluation
Cognation
Interpretation required

## INTERACTION FRAMEWORK

Norman's model does not consider the system's role in the communication. To remedy this, **Abowd and Beale** (1991) extended Norman's work by proposing the interaction framework. We use the interaction framework to judge the usability of the interface.

The interaction framework consists of 4 components

- user
- input
- system
- output



**Interaction Framework - Components**

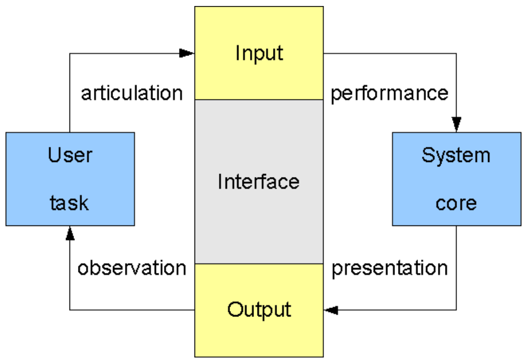
The framework introduces languages for input and output in addition to the core and task languages. By concentrating on the language translations, the interaction framework allows us to determine if the concepts are being communicated correctly.

**Interaction Steps**
The interaction framework identifies four steps in the interaction cycle:

1. the user formulates the goal and a task to achieve that goal
2. the interface translates the input language into the core language
3. the system transforms itself into a new state
4. the system renders the new state in the output language and sends it to the user

**Translations**
The four translations involved in the interaction framework are

1. articulation - the user articulates the task in the input language
2. performance - the interface translates the input language into stimuli for the system
3. presentation - the system presents the results in the output language
4. observation - the user translates the output language into personal understanding



**Interaction Framework - Translations**

**Articulation:** Consider a user who wants to turn on the lights at the far end of a room.

- **Poor Articulation:** The user sees a bank of unlabelled switches. The user has no idea which switch controls the lights at the far end of the room. The problem here is that the language provided (the unlabelled switches) does not allow the user to articulate the goal (turn on the lights at the far end of the room).
- **Good Articulation:** The switches are clearly labelled. The user articulates their task of pressing the switch that is labelled "the far end of the room". The language provided here (the labelled switches) allows the user to articulate their task without difficulty.

**Performance:** This translation is determined by the designer or programmer (not the user).

- **Poor Performance:** Consider a remote control for a television without a button for turning off the television. The user must go directly to the device and turn it off on the control panel.

**Presentation:** This translation is determined by the designer or programmer.

- **Poor Presentation:** Consider writing an essay using a word processor. You need to see the effects of your editing as a whole. However, the word processor only displays the immediate paragraph without the surrounding text or other pages. The surrounding text and other pages may have changed as well during the editing. In effect, all of the state changes cannot be displayed in the output language.

**Observation:** This translation is done by the user.

**EXAMPLE**
Consider some deficiencies that might arise in configuring a game:

- Articulatory Problem - the user is not sure which options to set in order to configure the game properly
- Performance Problem - the controller does not have the ability to select the option
- Presentation Problem - the display does not show that the option has been set
- Observational Problem - the user does not interpret the display correctly

Anyone of these deficiencies would give rise to an interaction problem.

## Heuristic Evaluation (Nielsen's ten heuristics)

A heuristic is a guideline or general principle or rule of thumb that can guide a design decision or be used to critique a decision that has already been made. *Heuristic evaluation*, developed by Jakob Nielsen and Rolf Molich, is a method for structuring the critique (Analysis) of a system using a set of relatively simple and general heuristics. Heuristic evaluation can be performed on a design specification so it is useful for evaluating early design. But it can also be used on prototypes, storyboards and fully functioning systems. It is therefore a flexible, relatively cheap approach. Hence it is often considered a *discount usability* technique.

The general idea behind heuristic evaluation is that several evaluators independently critique a system to come up with potential usability problems. It is important that there be several of these evaluators and that the evaluations be done independently. Nielsen's experience indicates that between three and five evaluators is sufficient, with five usually resulting in about 75% of the overall usability problems being discovered.

To aid the evaluators in discovering usability problems, a set of 10 heuristics are provided. The heuristics are related to *principles* and *guidelines*. Nielsen recommends the use of these 10 as providing the most effective coverage of the most common usability problems.

Each evaluator assesses the system and notes violations of any of these heuristics that would indicate a potential usability problem. The evaluator also assesses the severity of each usability problem, based on four factors:

1. how common is the problem,
2. how easy is it for the user to overcome,
3. will it be a one-off problem or a persistent one, and
4. How seriously will the problem be perceived?

**These can be combined into an overall severity rating on a scale of 0–4:**

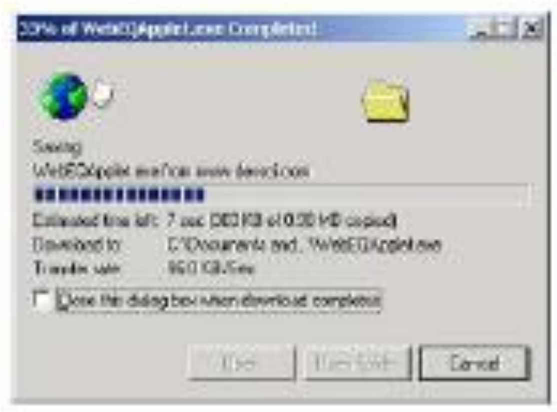
**0** I don't agree that this is a usability problem at all
**1** Cosmetic problem only: need not be fixed unless extra time is available on project
**2** Minor usability problem: fixing this should be given low priority
**3** Major usability problem: important to fix, so should be given high priority
**4** Usability catastrophe: imperative to fix this before product can be released (Nielsen)

## Nielsen’s Ten Heuristics are:

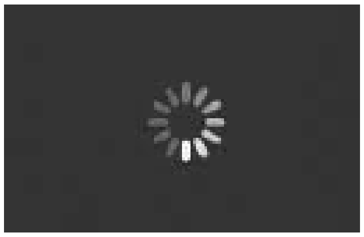
1. **Visibility of system status:** Always keep users informed about what is going on, through appropriate feedback within reasonable time. For example, if a system operation will take some time, give an indication of how long and how much is complete.

Elaboration: This means the user needs to be constantly made aware of his/her interaction with the interface while interacting. The control response ratio (input –output time) need to be as small as possible. Any interface needs to communicate that it is in a ready state to be operated upon –at the start of an interaction cycle.

**Example:** A glowing LED / flashing element indicating that the interface is live.
Most important to users is to know "Where am I?" and 'Where can I go next?" Internal reference is a must to feel in control.



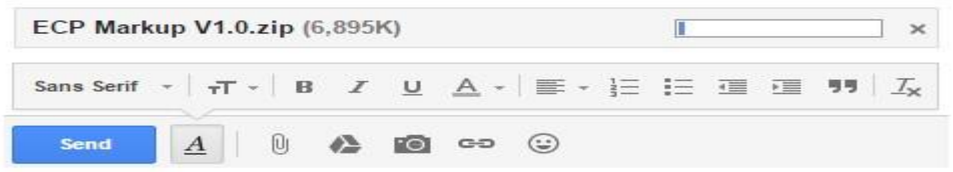
An animated symbol that states that ‘saving’ act is going on.....



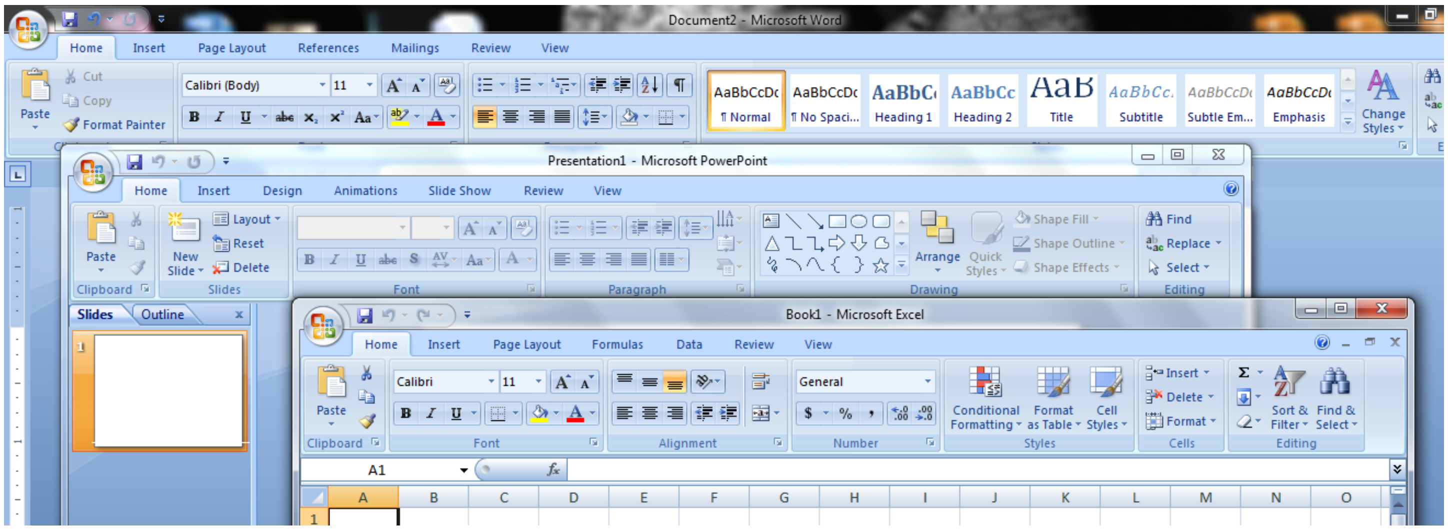
2. **Match between system and the real world:** The system should speak the user’s language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in natural and logical order. It’s also important for the application to speak the language of the target user base.

**Example**: Recycle bin icon is similar to a real bin, and icon itself shows weather it has files in it or not.

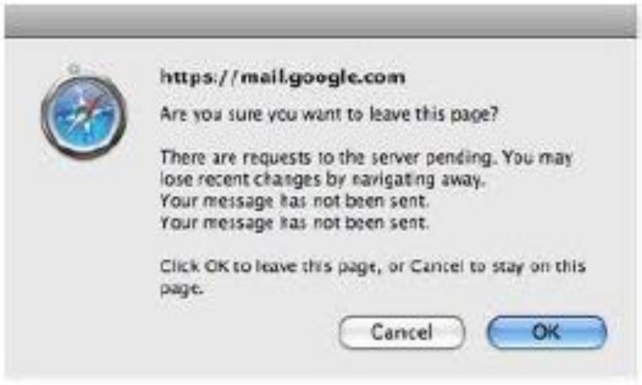
3. **User control and freedom:** Users often choose system functions by mistake and need a clearly marked ‘emergency exit’ to leave the unwanted state without having to go through an extended dialog. This principle talks about giving the user the freedom to navigate and perform actions. The freedom to undo any accidental actions**.** Support undo and redo.
**Example:** If you attached a large file in gmail by mistakenly, you can cancel it before its fully uploaded.



4. **Consistency and standards:** Users should not have to wonder whether words, situations or actions mean the same thing in different contexts. Follow platform conventions and accepted standards.
Within an interface if multiple words or actions are used to mean the same thing, it only leads to confusion in the user due to perceived lack of consistency. Interaction pattern gets disrupted. When pattern becomes complex, user's cognitive load increases.
Consistency in dialogue as well as in visual elements is achieved by specifying and adhering to a dictionary of words / labels/ symbols/ colors which together form a 'standard' –a prescribed set – compulsorily to be followed.
**Example:** Microsoft Word, Excel, and PowerPoint all use the same style toolbar with the same primary menu options: Home, Insert, Page Layout… Consistency results in efficiency and perceived intuitiveness


Document2 - Microsoft Word
Home Insert Page Layout References Mailings Review View
Paste Cut Copy Format Painter
Calibri (Body) 11
Normal No Spaci... Heading 1 Heading 2 Title Subtitle Subtle Em... Emphasis
Change Styles
Presentation1 - Microsoft PowerPoint
Home Insert Design Animations Slide Show Review View
Paste New Slide Layout Reset Delete
Clipboard Slides Font Paragraph Drawing Editing
Arrange Quick Styles Shape Fill Shape Outline Shape Effects
Find Replace Select
Slides Outline
Book1 - Microsoft Excel
Home Insert Page Layout Formulas Data Review View
Calibri 11 General
Conditional Formatting Format as Table Cell Styles
Insert Delete Format
Sort & Filter Find & Select
Clipboard Font Alignment Number Styles Cells Editing
A1 fx
A B C D E F G H I J K L M N O

5. **Error prevention:** Make it difficult to make errors. Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
To err is human. Errors can happen regardless the level of expertise of the user or familiarity of the interface. A good principle of design is to seek out error prone interactions, build in error prevention within the dialogue. Forewarning, restricting, prompting, retracing or recovery routes, etc. are means of addressing errors.
**Example:** GUI-style widgets cut down on the errors but may still have to be double checked before confirmation.


https://mail.google.com

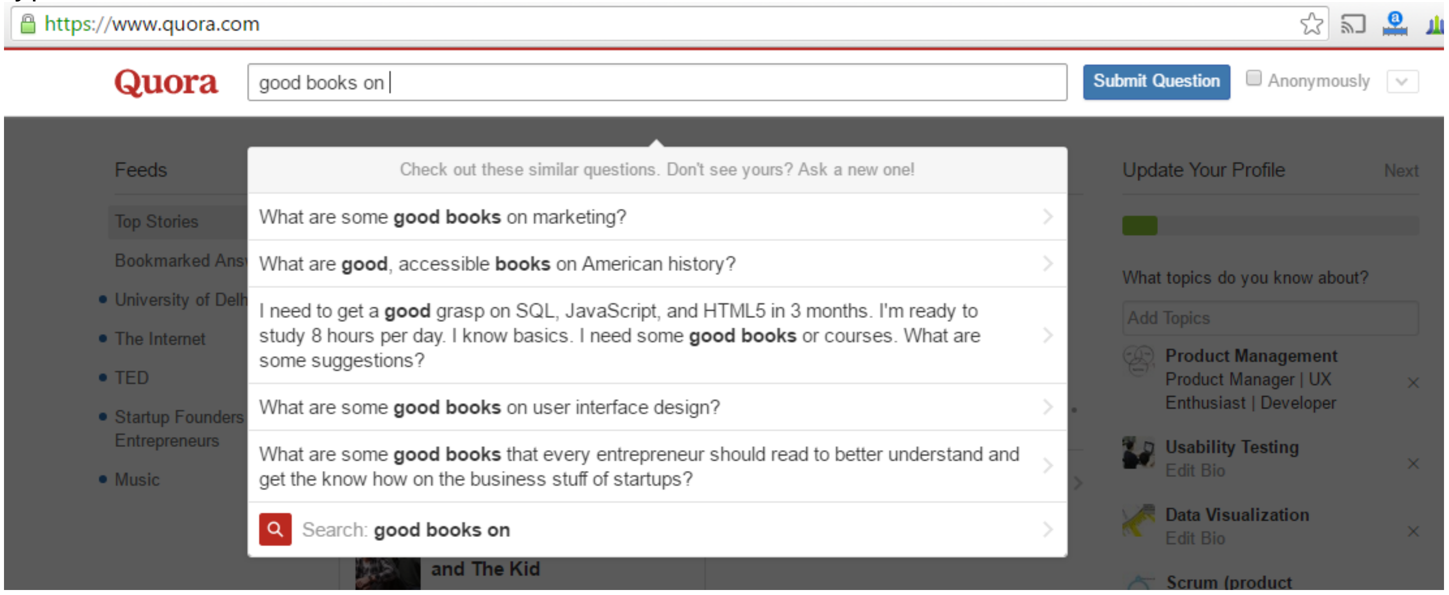
Are you sure you want to leave this page?

There are requests to the server pending. You may lose recent changes by navigating away.
Your message has not been sent.
Your message has not been sent.

Click OK to leave this page, or Cancel to stay on this page.

6. **Recognition rather than recall:** Make objects, actions and options visible. The user should not have to remember information from one part of the dialog to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Reduction on cognitive load during the interaction ensures that the user is not asked to rely on means and methods that extract human cost. If an interface requires specialized training and use of memory to operate it will be quickly abandoned by the user.
Analogy, metaphor, symbols, sounds, etc. are used as design elements in an interface to ease recall thereby eliminating the need for 'thinking while interacting' and memory loads for the user.
**Example:** Below is an example of Quora suggesting possible questions based on what I am trying to type.



Check out these similar questions. Don't see yours? Ask a new one!

What are some **good books** on marketing?

What are **good**, accessible **books** on American history?

I need to get a **good** grasp on SQL, JavaScript, and HTML5 in 3 months. I'm ready to study 8 hours per day. I know basics. I need some **good books** or courses. What are some suggestions?

What are some **good books** on user interface design?

What are some **good books** that every entrepreneur should read to better understand and get the know how on the business stuff of startups?

Search: **good books on**

**7. Flexibility and efficiency of use:** Allow users to tailor frequent actions. Accelerators – unseen by the novice user – may often speed up the interaction for the expert user to such an extent that the system can cater to both inexperienced and experienced users.

Once a user becomes adept at using an interface, he/she upgrades into a higher level user from a novice. Such users will always seek to complete the task faster. Such users seek out shortcuts. An interface need to allow this. It needs to be flexible and make it possible for the user to adopt quicker dialogues through shortcuts. The user feels efficient as well as proficient. The feeling of having mastered the software is a flexible sign of being in control thereby.
**Example:** The Interface should be flexible transforming itself between a novice user and an advanced user**.** One frequents this option while installing a new software that asks if the user wants to go ahead with default installation or custom installation. An advanced user chooses a custom installation to cut out the unnecessary services.

**8. Aesthetic and minimalist design:** Dialogs should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialog competes with the relevant units of information and diminishes their relative visibility.
Visual clutter in the interface only adds to inefficiency however impressive it is visually. Simplicity is equal to efficiency is equal to elegance is equal to beauty is the aesthetic algorithm in minimalism. Use of least number of elements (minimalism) is more 'scientific' rather than 'artistic'. Visual noise needs to be completely eliminated.

**9. Help users recognize, diagnose and recover from errors:** Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Nobody likes to be loudly informed that he/ she has erred. Error messages need to be disused as suggestions / prompts and precise instructions so as to be able to correct the error and recover. The learning component in errors so that the user recognizes the error as it is being made, or recognizes the reason why the error happened in the first place helps the user learn.

**10**. **Help and documentation:** Few systems can be used with no instructions so it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Elaboration: This again is to assist the user learn and understand the dialogue between the user and the machine or understand where what went wrong or aid recall during memory lapses due to long usage time gaps. Adequate 'Help 'support system when the user wants and at the point where the user wants it is a good principle of Interface design.

**Contextual Inquiry**

**Introduction**

Contextual inquiry is a field based data collection technique employed to capture detailed information about how users of a product interact with the product in their normal work environment or in other words-interact with the product in its context of use.

In Human Centered Designing methodology, understanding the users, their needs, the context in which these needs raise and the context in which the user attempts to fulfill needs is the first step. Specific techniques have been developed to identify and specify the context of use. **This process is called "Contextual Inquiry".**

Contextual Inquiry is a scientific way of understanding user's needs, their intentions and their practices.

**Definition:** Contextual inquiry is the systematic analysis based on observations of users performing tasks / activity in a context. Hypothesis is made linking cause effect based on these observations. The hypothesis are tested in discussion with the users. As a result of this the context itself gets understood in all the dimensions.

By 'Context' is meant the anchoring environment /situation/ reference / work activity -with respect to which a designing process (solving a problem or conceptualizing a new product) is underway.

Contextual Inquiry is predominantly a qualitative method. In some cases it is a qualitative cum quantitative method of research. The techniques used in Contextual inquiry are rooted in Ethnography, Psychology, and Ergonomics & Design.

**Results of Contextual Inquiry are used** to formulate the Users' conceptual model based on visualization of the users Mental Maps of tasks, intentions, interpretation and action.

**Advantages of the Contextual Inquiry method over other user data collection methods:**

- Marketing based data or information on the user as a 'customer' or 'consumer' is of limited use for a HCI designer as it does not give mental & psychological insights while the user is using the device.
- This method being open ended makes it valuable deep-mining of tacit knowledge from the user. Tacit knowledge is that knowledge which normally the user is not consciously aware of themselves. It helps to develop a shared understanding between the device interface creator and the user.
- Even though both qualitative as well as quantitative data is involved, this method is reliable and scientific.

**Disadvantages are few**.

Since majority of information is qualitative it is not provable statistically significant. The inquirer needs to be highly skilled in multiple disciplines such as Ethnography, Psychology, Culture, Design and HCI.

**Some field based difficulties:**

- Gaining confidence of shy and suspicious users can pose a problem.
- Users may not want to be seen as stupid and hence may exhibit extra smartness (mislead). It is well known that when observed humans do things different from the way when alone.

**Contextual Inquiry Methods:**

**In short the method involves:**

- Going to the user's environment
- Observe real work in natural conditions
- Seek clarifications and confirmations through questions.
- Conceive the field observed data into a model.

**The user is treated as an expert.**

- Interviewer observes users in real time as they perform the tasks.
- Questions on the users' actions are asked so as to understand their motivations and approach to a given set of interactions with the interface.
- Care is taken NOT to 'lead' the user by prompting while inquiring or assisting them in completing their answers.
- Interviews /observations are conducted at the user's actual work place /environment.

**Data gathering processes:**

- Inquiry alternates between observing and discussing / clarifying from the user as to what the user did and why.
- In this technique the researcher interprets and shares insights with the user during the interview / discussions.
- Often the researcher's understanding stands corrected by the user. Researcher needs to take care that the discussions do not move away from the focus of the contextual inquiry.

**Planning for a Contextual Inquiry:**

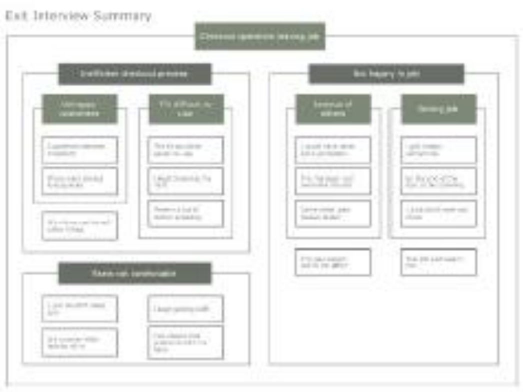
- Define the issue / problem /context for which the Inquiry is being planned.
- Plan for identifying users, their location, their numbers, and their willingness to cooperate.
- Work on the briefing that will be given to the participating users. Prepare a list of possible questions to start the dialogue with the users.
- Prepare documenting mediums such as cameras, voice recorders etc.

**Tools / Instruments used in Contextual Inquiry:**

- Open ended questioning based on observations
- Pre-prepared Questionnaire (User Survey)
- Ethnographic observation dairy with notes (These notes are converted into Affinity diagrams)
- Focus group interviews
- Structured discussions
- Photo / video documentation.
- Hierarchy diagrams
- Story boards
- Mind maps

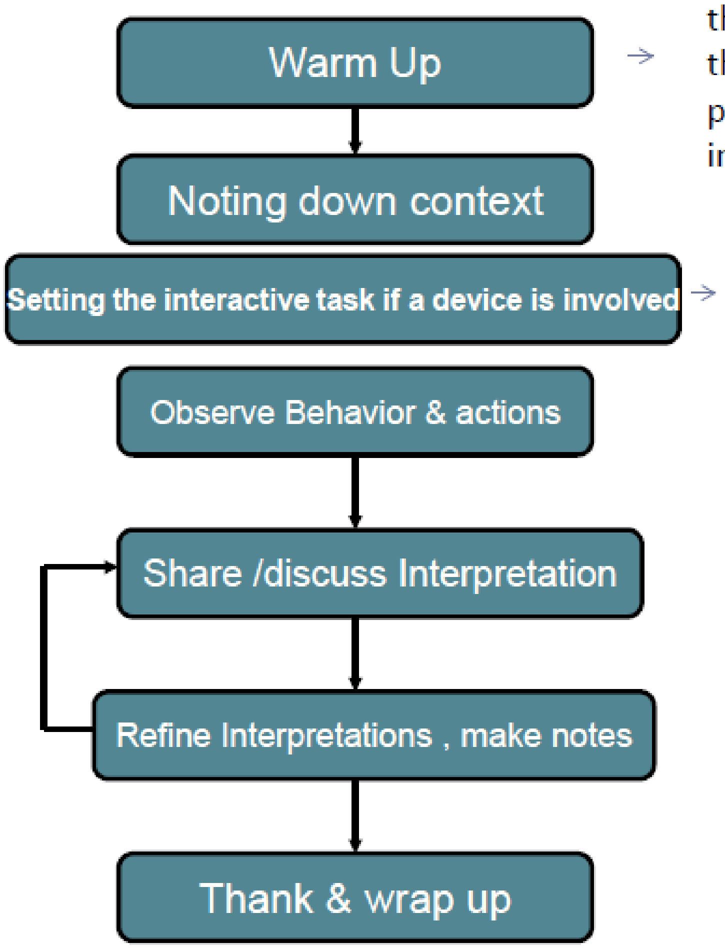
**Affinity diagrams:** Data that have affinity to each other based on are grouped together to form a category.

Cards on which labels / words describing a characteristic are moved around on a board to ultimately result in 'groups'. Groups are given labels.



**Stages of a Contextual Interview:**

Stages of a Contextual Interview


Warm Up
Noting down context
Setting the interactive task if a device is involved
Observe Behavior & actions
Share /discuss Interpretation
Refine Interpretations , make notes
Thank & wrap up

Greetings. Explaining the purpose. Rights of the subjects for protection of personal information.

Prejudging the user. Gathering usage data & experience data of the user.

**Analyzing the data collected in Contextual inquiry:**

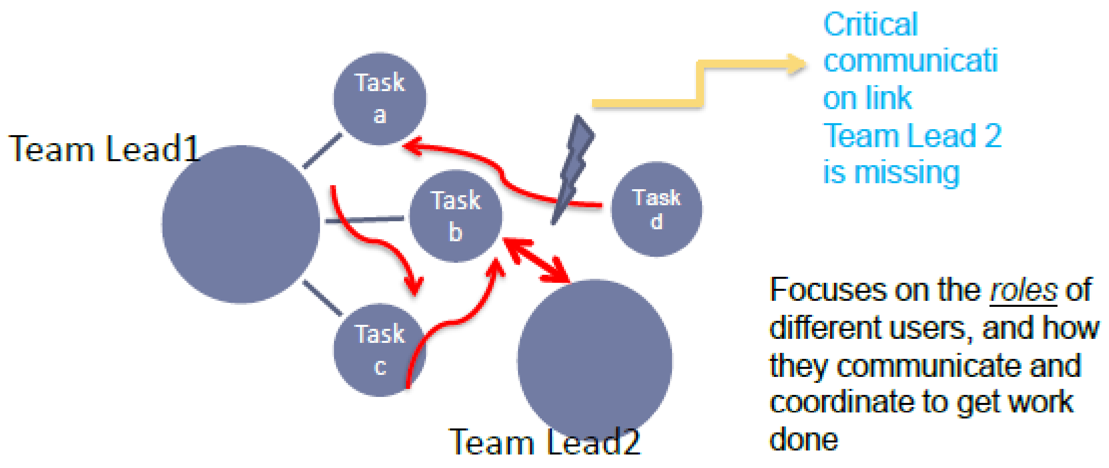
Data collected from contextual inquiry is analyzed, interpreted and finally visualized and represented by the researcher using one or all the following models which are part and parcel of the HCD process.

- Flow Model
- Sequence Model
- Cultural Model
- Artifact Model
- Physical Model

**Descriptions of the Models:**

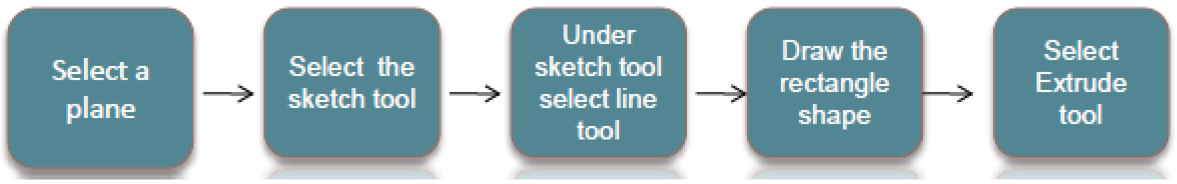
**Flow Model:**

Flow model represents the coordination, communication, interaction, roles, and responsibilities of the people in a certain work practice. It is based on the logic of flow of information between different entities making up the system within the context. This model is mainly use to depict the logic behind the flow of information.


Team Lead1
Task a
Task b
Task c
Task d
Team Lead2
Critical communication link Team Lead 2 is missing
Focuses on the *roles* of different users, and how they communicate and coordinate to get work done

**Sequence model**:

Sequence represents the steps users go through -to accomplish an activity. Sequence models are liner and sequential in nature. Sequence models of a number of smaller tasks when integrated represent the interconnected sequence within a larger system as shown in figure:



Low-level, step-by-step information on how work is actually done"

Includes the intent behind the action, the trigger that led the user to this action, and breakdowns that create problems.

**Cultural model:**

Cultural model represents the norms, influences, and practices that are present in the work environment and which are specific to a particular region or are traditionally followed as local norms. Often culture specific comments or differences are mentioned using either flow diagrams or sequence diagrams or both.

Language for example is a Culture model variable.

In Indian culture parents are the ones who establish the first contact and collects information of a prospective son in law /daughter in law. This flow model is not evident for a person from another culture.

**Artifact model:**

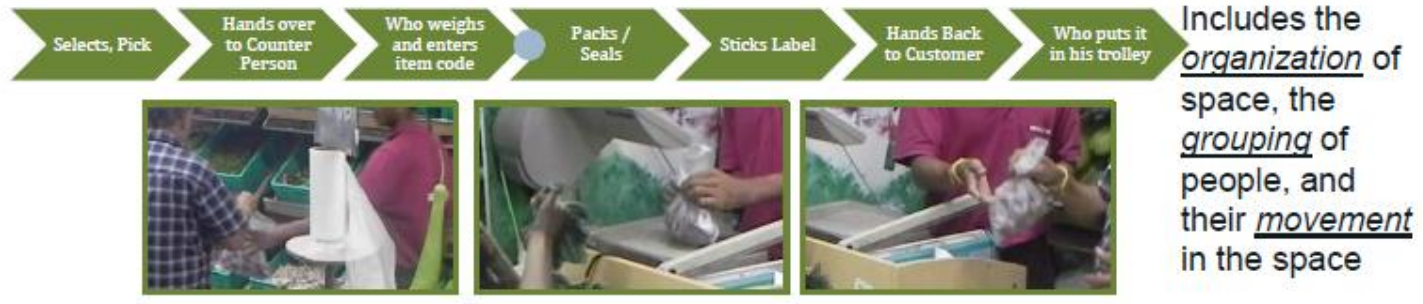
Artifact model represents the documents or other physical things that are part of the work / task execution. These artifacts are aids to the tasks created while working or are used to support the work. Example would be a Paper based voucher simultaneously filled up in a particular step of a sequential task flow.

Interviewers should inquire into the structure, content, presentation and usage of the artifact.

**Physical model:**

Physical model represents the physical layout of the environment where the work tasks are accomplished. Simple examples would be office layout, network topology, or the layout of icons on a computer display environment.

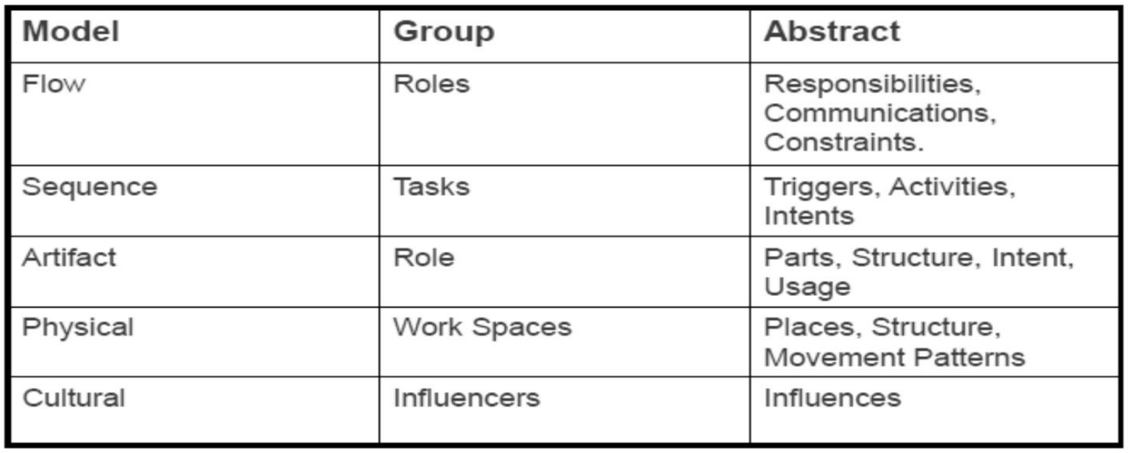
The flow of work as it moves in the physical environment is represented as a:


Selects, Pick
Hands over to Counter Person
Who weighs and enters item code
Packs / Seals
Sticks Label
Hands Back to Customer
Who puts it in his trolley
Includes the *organization* of space, the *grouping* of people, and their *movement* in the space

**Consolidating Work Models:**

All flow models are taken for each user and their interconnectedness to form a whole system is attempted. The groups are formed based on roles played by individual users. Extracts from the flow models represent abstracts of communications, responsibilities and constraints. The same thing is done with other models.

A report on the contextual inquiry is generated for designing team.



| Model | Group | Abstract |
|---|---|---|
| Flow | Roles | Responsibilities, Communications, Constraints. |
| Sequence | Tasks | Triggers, Activities, Intents |
| Artifact | Role | Parts, Structure, Intent, Usage |
| Physical | Work Spaces | Places, Structure, Movement Patterns |
| Cultural | Influencers | Influences |

**COGNITIVE WALKTHROUGH**

Cognitive walkthrough is a usability method that focuses on evaluating a design (existing or proposed) for ease of learning particularly through explorations by the User.

**Cognitive Walkthroughs [CWT] are:**

- Usability Inspection Methods [UIM]
- Focus on Evaluating a Design's navigation.
- Basis is 'Ease of Learning' by self-exploration by the user

**Background:**

Cognitive Walkthrough has the same basic structure and rationale found in software walkthroughs (Yourdon 1989).

In the software code walkthrough, the sequence represents a segment of the program code that is ' walked through' by the reviewers to check certain characteristics (e.g., that coding style is adhered to, conventions for spelling variables versus procedure calls, and to check that system wide invariants are not violated).

- In cognitive walkthrough [CWT], the sequence of actions refers to the steps that a user will require to perform on the interface so as to accomplish a task.
- The evaluators then 'walk through' that action sequence to check it for potential usability problems.
- Focus of the cognitive walkthrough is to establish how easy a system is to learn by operating it. The focus is on learning through exploration.

**CWT Questions:**

1. Will the user try to achieve the right effect?

2. Will the user notice that the correct action is available?

3. Will the user associate the correct action with the effect that the user is trying to achieve?

4. If the correct action is performed will the user see that progress is being made towards solution of the task?

**Significance: Walkthroughs help answer interfaces design questions like:**

•How will the user approach a task?

•What is the correct action sequence for each task and how it needs to be described to the user?

•How to achieve the desired action sequence form the user with minimum human cost and maximum efficiency

•How quickly will the user learn & becomes comfortable with the interface?

**Example of where WTC is useful**

When ATMs were first introduced one of the questions on the design of operational sequence was – Should balance in account be displayed simultaneously every time the user access the ATM?

Or

Is it better to display balance after the transaction is over?

A walk through reveled both the above assumptions are out of sequence as far as the user is concerned. Seeking 'Balance' is a sub goal either before starting of a transaction or after a transaction is over. In either case it needs to be an independent Goal by itself rather than a sub goal of accessing the account. Users approach ATM more often for Withdrawal than for knowing Balance.

In hind sight –this wanting to know the 'balance', though seems to be implicit-is not necessarily so.

**Cognitive Walk Through has two phases:**

1. **Preparation phase:**

i) Building a prototype {paper; mock up; screen based} with description. It need not be perfect or complete in all request.

ii) Making a list of selected tasks you want the user to 'walk through' the interface along with you. The task should have ready well defined sequences for Goals and sub goals with written actions used to complete each individual task.

iii) A clear understanding of the user, his /her background; level of expertise in the domain; prior experience of using similar software etc.

2. **Evaluation phase:**

i) Conducting the Walkthrough session

ii) Recording the Observations

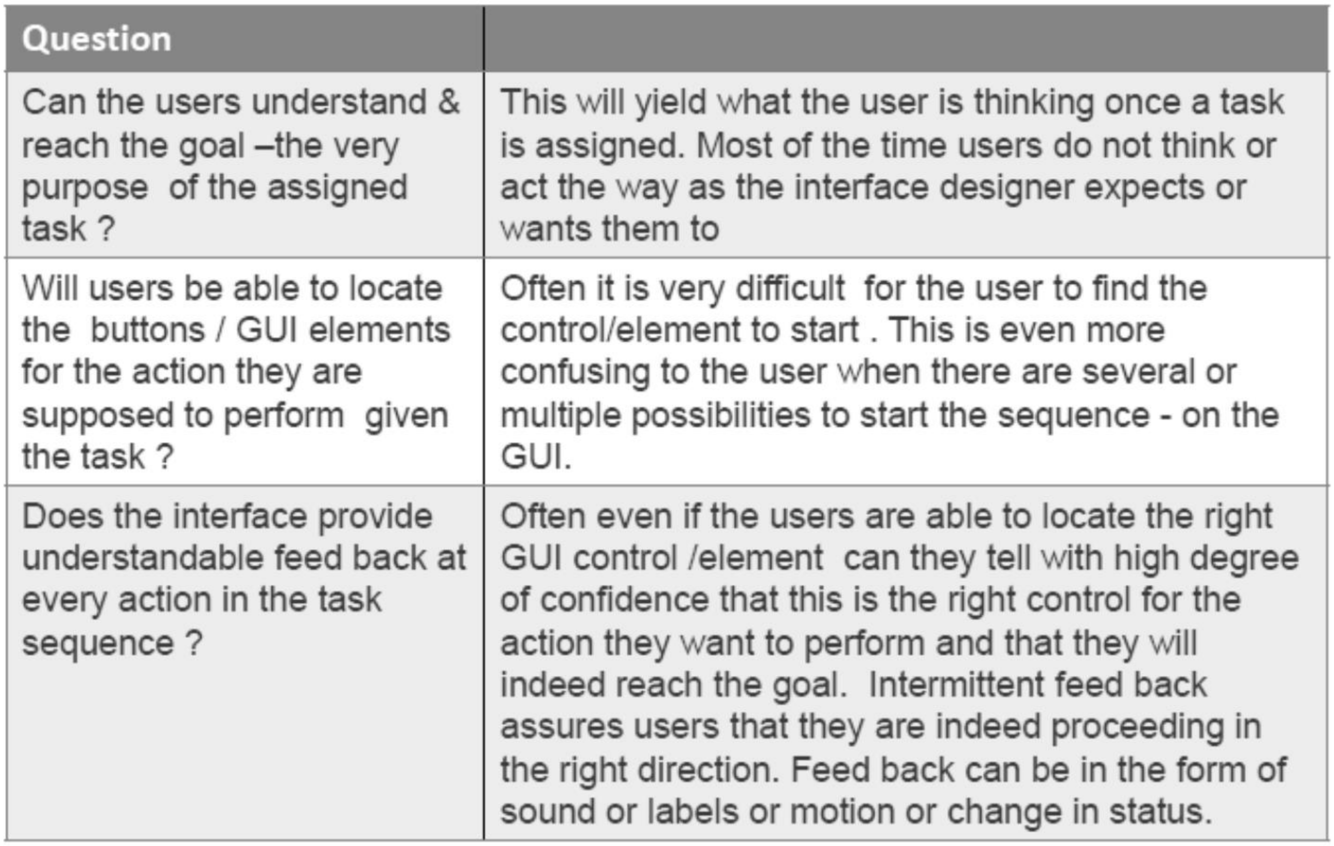
iii) Analysis

iv) Inferences

v) Recommendation to Interface Team

The evaluator should prepare to look for answers to Questions in table:

The purpose of conducting the walk through must be clear to the evaluator prior to starting the walk through.



| Question | |
|---|---|
| Can the users understand & reach the goal –the very purpose of the assigned task ? | This will yield what the user is thinking once a task is assigned. Most of the time users do not think or act the way as the interface designer expects or wants them to |
| Will users be able to locate the buttons / GUI elements for the action they are supposed to perform given the task ? | Often it is very difficult for the user to find the control/element to start . This is even more confusing to the user when there are several or multiple possibilities to start the sequence - on the GUI. |
| Does the interface provide understandable feed back at every action in the task sequence ? | Often even if the users are able to locate the right GUI control /element can they tell with high degree of confidence that this is the right control for the action they want to perform and that they will indeed reach the goal. Intermittent feed back assures users that they are indeed proceeding in the right direction. Feed back can be in the form of sound or labels or motion or change in status. |

**Overview of the actual Walkthrough Processes**

**Pre-preparation:**

1. Define Users: Who are the users. Identify them. (Categorize them as Novices, Intermittent & Experts)
2. Identify the tasks for the evaluation:
   Ex: Evaluation for "Checking out Balance on an ATM"
   Prepare notes on what the user must know prior to performing the task and what the user should be learning while performing the task.
3. Prepare action sequences for completing the Tasks

   Make a "AND THEN "list of Goals & sub glass.

   Ex: Overall Goal: Find out balance from the ATM

   Subgoal1: Activate ATM [Physical action Insert Card]

   Subgoal2: Identify self [Input pin code]

   Sub goal 3: Get balance [press action button with label]

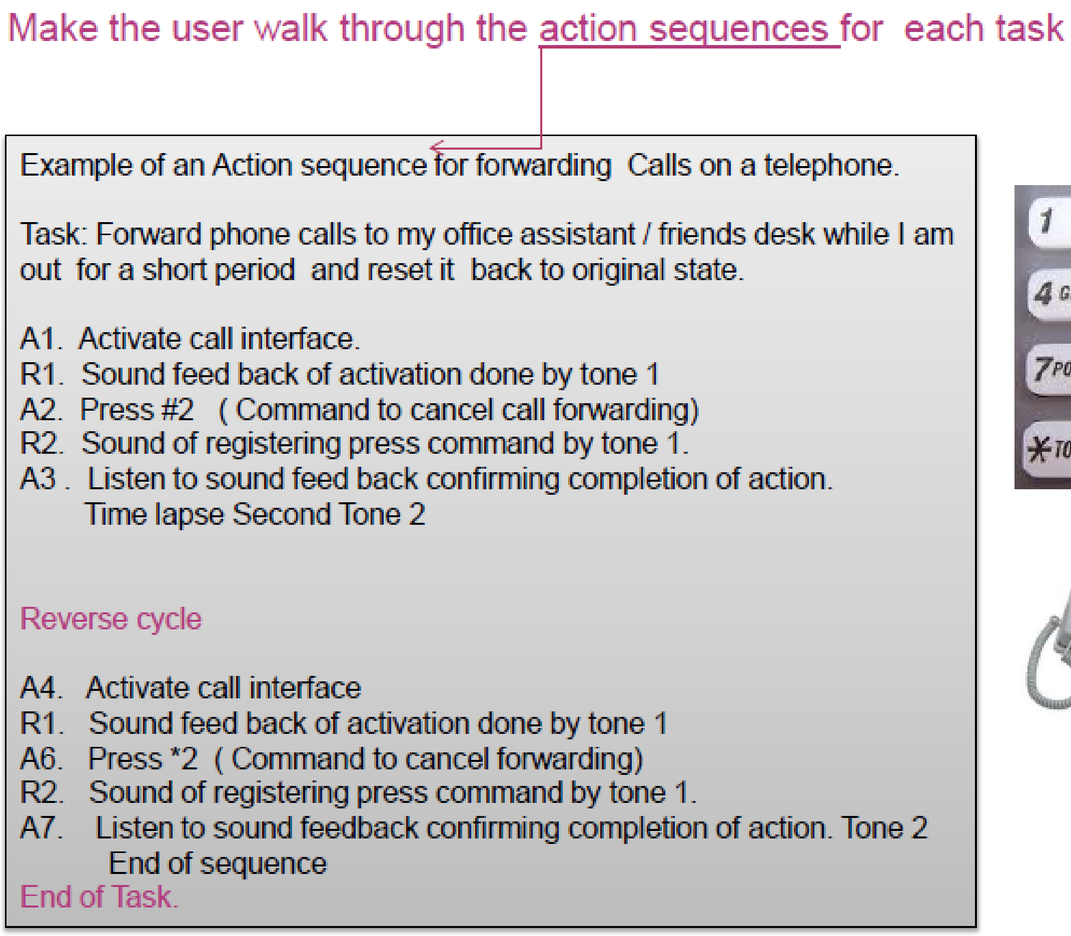
Sub goal 4: Get a printout [if required]

Sub goal5: Log out from ATM.

4. Conduct the Walk Through Session

**Conducting the Walk -through Session**

- Using the mock up prototype ask the user to perform a task. See Example of a mock up bellow
- Make the user walk through the action sequences for each task. See Example in next slide
- Make a recording of observations in a Recording Sheet.



Make the user walk through the action sequences for each task

Example of an Action sequence for forwarding Calls on a telephone.

Task: Forward phone calls to my office assistant / friends desk while I am out for a short period and reset it back to original state.

A1. Activate call interface.
R1. Sound feed back of activation done by tone 1
A2. Press #2 ( Command to cancel call forwarding)
R2. Sound of registering press command by tone 1.
A3 . Listen to sound feed back confirming completion of action.
Time lapse Second Tone 2

Reverse cycle

A4. Activate call interface
R1. Sound feed back of activation done by tone 1
A6. Press *2 ( Command to cancel forwarding)
R2. Sound of registering press command by tone 1.
A7. Listen to sound feedback confirming completion of action. Tone 2
End of sequence
End of Task.

**Observation during the Walk Trough**

- The above task is assigned to a user. The user is asked to proceed executing the task on a mock up / paper prototype / wire frame prototype.
- The user is asked to achieve a goal (of forwarding a call in his absence and informed about the sequence of actions.
- The sequence of inputs as carried out by the user are observed.
- The errors committed (deviation from the expected sequence and corresponding action) are noted.
- The difficulties are mutually discussed with the user. Why a user acted in particular way and did not act in ways that was expected is explored.

**Make a recording of observations in a Recording Sheet.**

| Description of step. | Did the user try to achieve the end goal or did he give up At the start itself. | Did the user notice that the correct action choices are available. Yes – PARTLY-No | Did the user confidently know that the choice being made by him/her is the right one ? Y N | Did the user understand the feedback after every action | Did the user Complete the Task With satisfaction Yes PARTLY No | Comments / Alternative suggestions/ solutions / discussion points. |
|---|---|---|---|---|---|---|
| A1. Activate call interface. | YES | PARTLY | YES | YES | YES | |
| A2. Press #2 ( Command to cancel call forwarding) | YES | YES | YES | NO | PARTLY | Was not paying attention to sound feed back as it was not expected. |

## Analysis & Inference

Evaluators Rating Sheet

| Action in Sequence | System mismatch question | Potential Problem & Design solution | % Mismatch to ideal situation (qualitative estimation) |
|---|---|---|---|
| A1. Activate call interface. | Is it clear to the user that system has taken input | Low clarity of sound. Ambient Noise. Increase volume | 30% |
| | Can the user resume control for the next action | YES | |
| | Are the systems response visible & interpretable | No | |
| | Is the end of the system action clear | YES | |
| A2. Press #2 (Command to cancel call forwarding) | Is it clear to the user that system has taken input | PARTLY | 50% |
| | Can the user resume control for the next action | NO | |
| | Are the systems response visible & interpretable | PARTLY | |
| | Is the end of the system action clear | NO | |

Summarize the findings:

Percentage of mismatch 50%
The Interface needs improvement .
Sound Tone to be changed
Sound Volume to be increased
Additional Feed back to be incorporated in A2

End of Walk through Testing Report

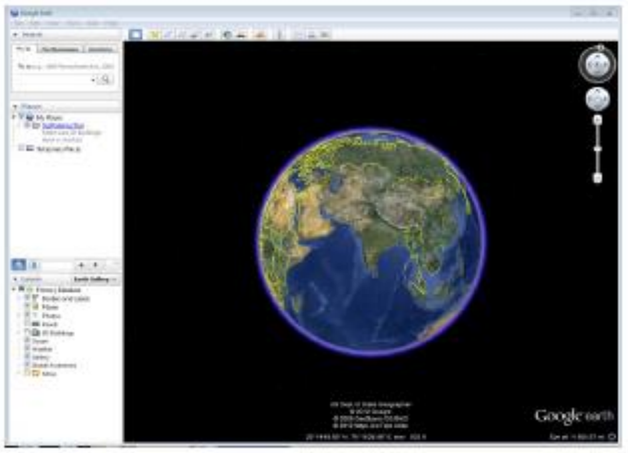
**Case Study 1: GOOGLE MAPS in Goggle Earth**

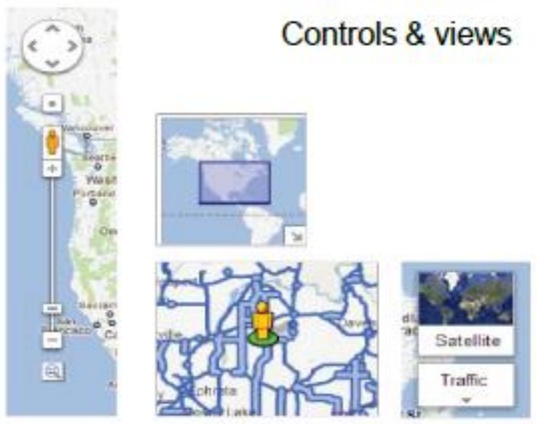
**Evaluator: 1 nos. Expert user**

**Heuristics Used: Nielsen's Ten Heuristics**

Google Maps is a well-known free service provided by Google worldwide.

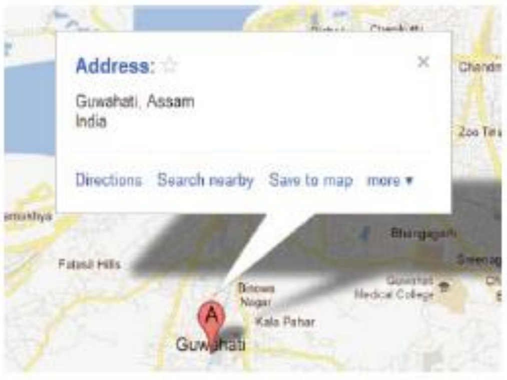
It's not just a bunch of maps, it includes multiple layers: roads, terrain, satellite, street view, traffic etc. It also integrates user ratings and pictures with locations and businesses in the area.

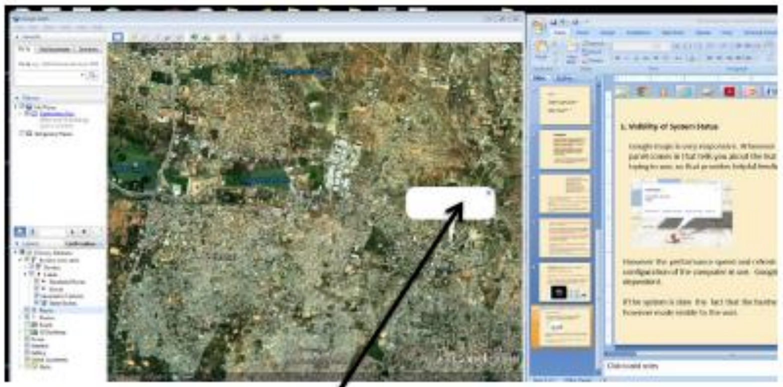




## 1. Visibility of System Status

"Google maps is very responsive. Whenever you click a button, a panel comes in that tells you about the feature that you're trying to use, so that provides helpful feedback."



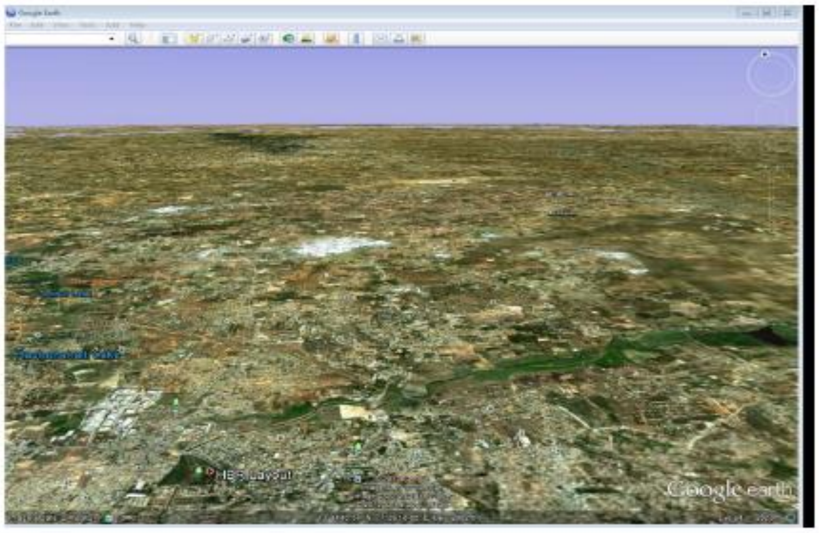


"System status if the Network connection is absent or suddenly is lost is not made visible to the user. The user sees a blank label" (second screen shot)

"However the performance speed and refresh rate depends on the hardware configuration of this computer in use. Google is very CPU intensive and RAM dependent.

If the system is slow due to hardware mismatch this fact that the hardware is not optimum -is not however made visible to the user. "

**2. Match between System and Real World**

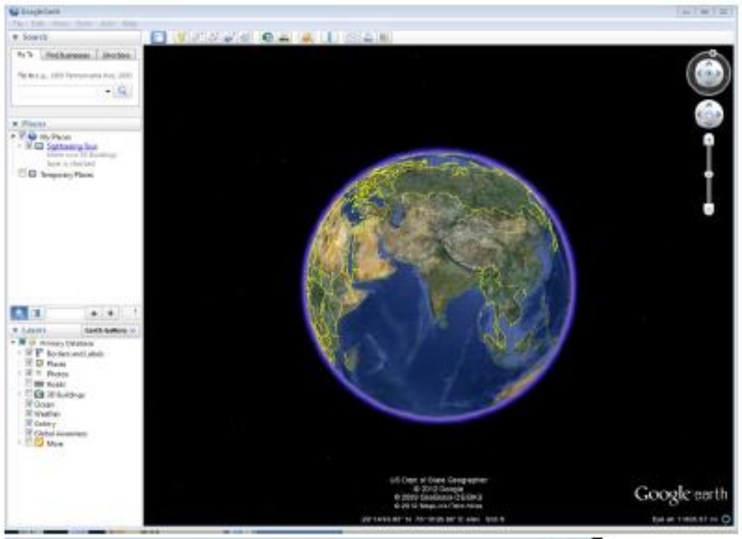


Closeness to real world is very good.

The window can be panned and horizon can be lowered giving a very Real world view.

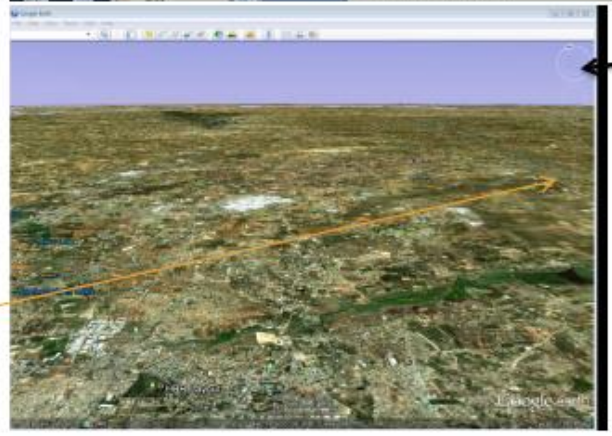
**3. User Control and Freedom**

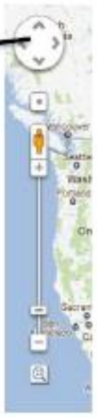
Almost all the features are available as checkboxes. When they are checked, those items are added to the map and when they're not checked they go right back off the map.



The zoom in controls are fairly intuitive. They recede into the background and come alive when mouse hovers on it. The direction ring gives full control to the user.

Freedom for the experienced user but for a novice a disappearing zoom slider bar can be confusing!





## 4. Consistency and Standards

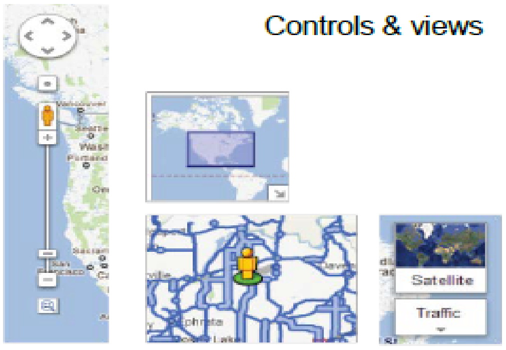
Google maps is pretty consistent with the words and phrases that they use.

the symbols are pretty clear and a user could probably figure it out without even needing the labels.

Successive screens maintain consistency in continuity in terms of how & where tools / buttons appear.

The response of various interactions is standardized across the entire application.

Controls & views

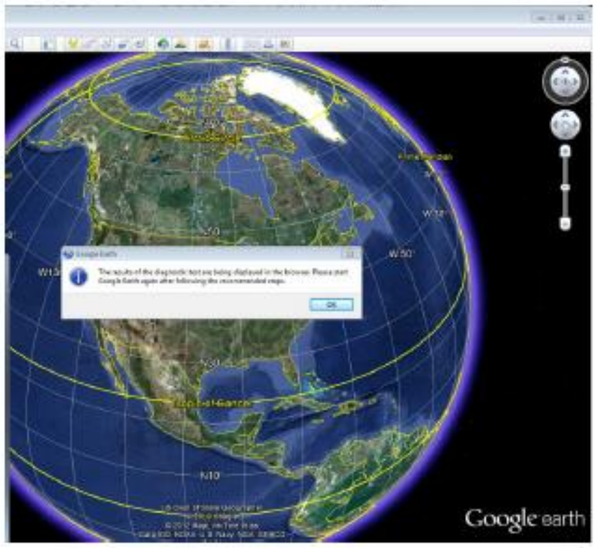


## 5. Error Prevention

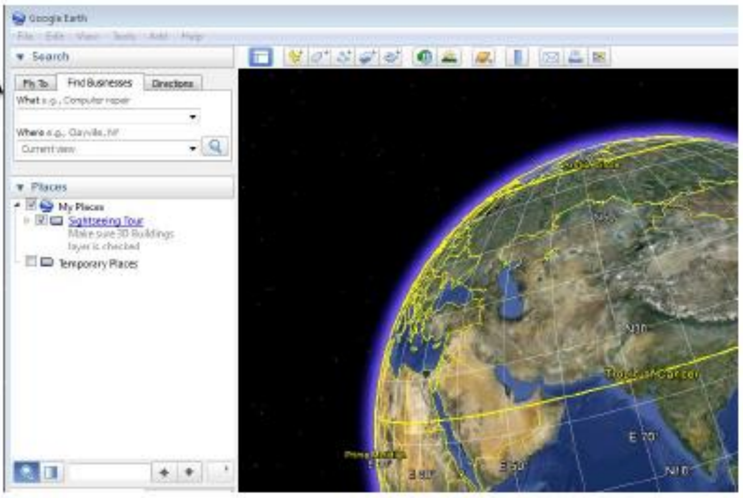
There really isn't much error involved in a mapping program.

There is a possibility that a user may enter the wrong address, but Google maps is well configured to automatically decipher what the user may have been looking for and presents them with a list of options that could be correct.

This not only prevents accidental errors but also suggest corrections for the user.



The results of the diagnostic test are being displayed in the browser. Please start Google Earth again after following the recommended steps.

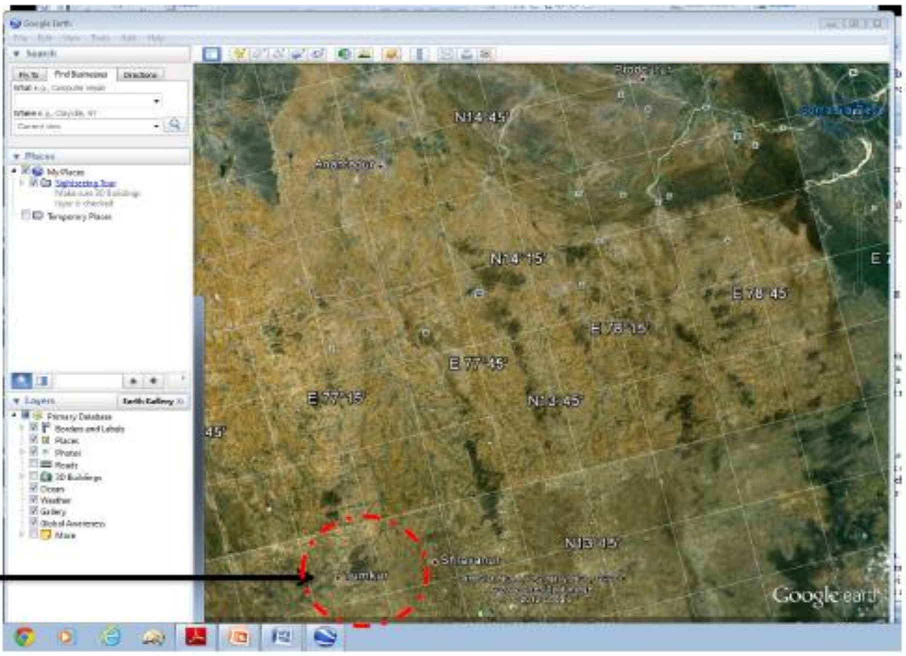


## 6. Recognition Rather than Recall

While most of the more useful functions are visible, but a user must click a button to get directions.

Navigating by panning and zooming often leads to being lost . Users often want to know which direction they need to pan.

Ex: On the right TUMKUR is visible. For a user to now go to BANGALURU & ( if the user is not sure where Bangalore is with respect to Tumkur - East, West, North, South) or is not familiar with the latitude - longitude of Bangalore – will have to recall or recognise or resort to trail and error by first zooming out. Zooming out also leads to disappearing of small towns like Tumkur -. At lower zoom levels the user will have to 'RECAL' which is not an easy thing to do on a map of an unfamiliar geography.



It is here that recall is required or in order to become aware ( recognize) where in the map one is with respect to overall map.

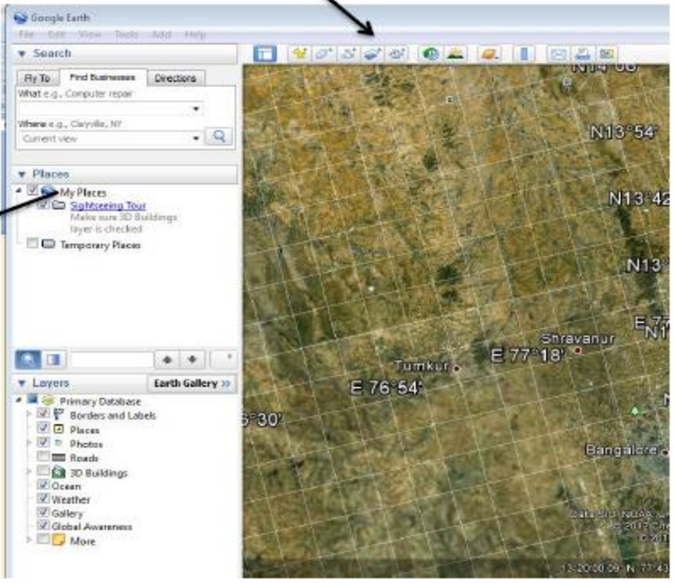
## 7. Flexibility and Efficiency of Use

The buttons on the task bar reflect the flexibility that is built in.

The Google Earth is highly flexible & efficient. Even if a GPS connection is not available

A user can set a reference location " starting point" and also key in where one wants to go –

The software map navigates itself by either panning , zooming, turning, and makes the destination visible.

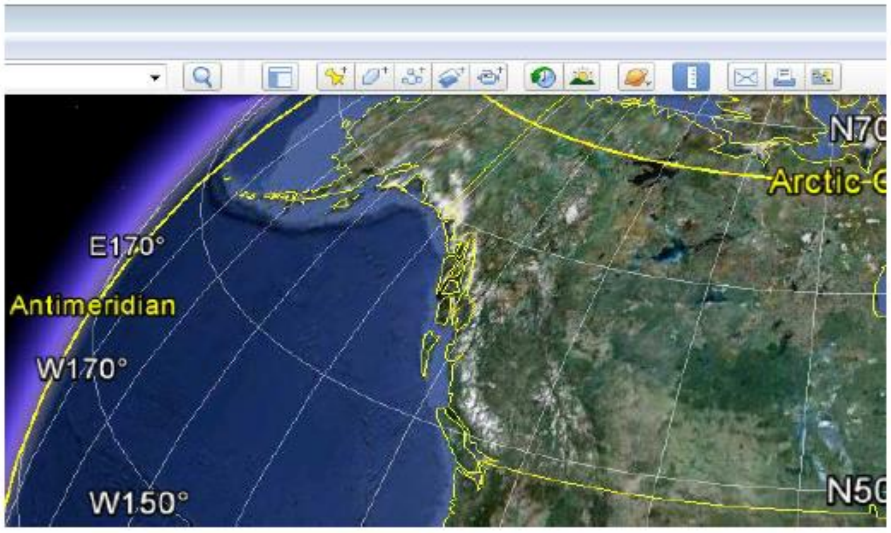


## 8. Aesthetic and Minimalist Design

The entire user experience is pleasant. The design of the interactive buttons is  subdued and minimalist.

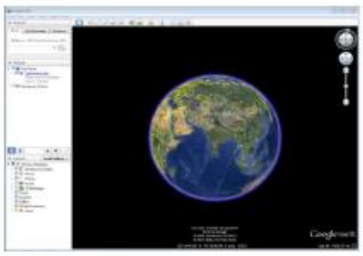
The colour tone of the labels, instructions etc that appear are not loud and brash.

The whole graphics is tuned to make the Map window important (which is the main function of the software)

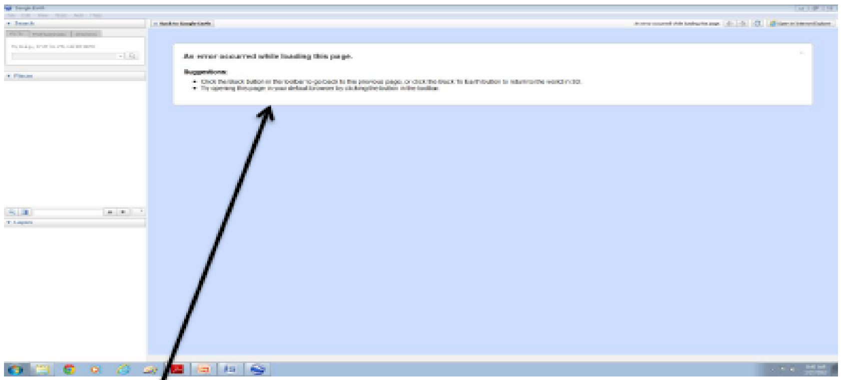


Address:

Guwahati, Assam
India



## 9. Help Users Recognize, Diagnose, and Recover from Errors


An error occurred while loading this page.

Suggestions:

- [illegible]
- [illegible]

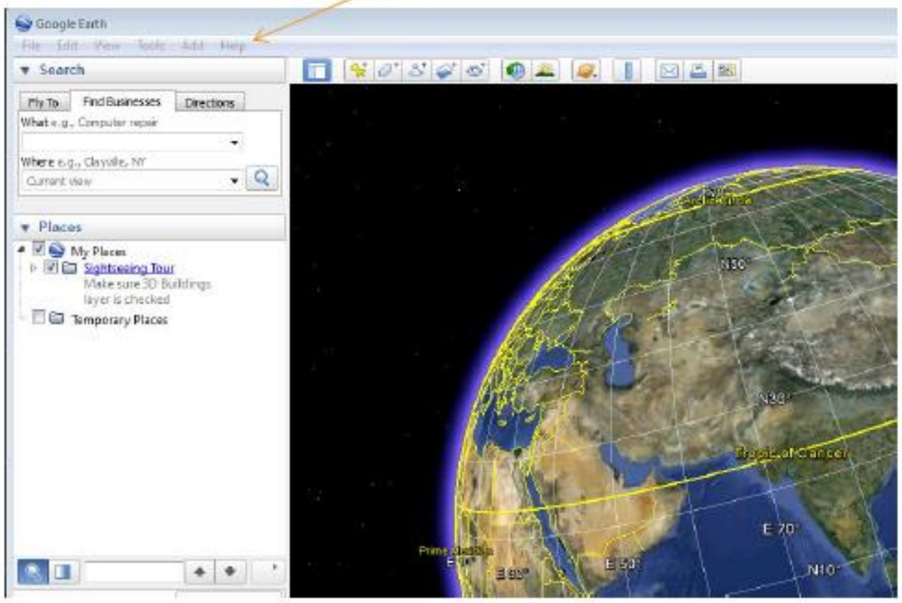
If a user enters an address that is not in the maps database, Google maps will suggest alternatives to help the user figure out the correct address they are looking for.

The software is user friendly.
It does not penalise the user for not knowing how to use the software.

Even if an error happens, the software recovers quickly without the user having to do much more than going back to a previous screen  by pressing “ Back” button.

## 10. Help and Documentation

In the Documentation under ' Help' it is easy to use and figure out where to find the information you're looking for.



## Conclusion

The Heuristic evaluation is compiled into a consolidated report by including results of other evaluators. Intensity of the problem may also be indicated in terms of severity. High severity means it is a HCI problem. Medium means the problem needs attention as it is partially resolved. Low means improvement can still be done to the existing state.

| Heuristics | Evaluator 1 | | Evaluator 2 | Evaluator 3 | Evaluator 4 |
|---|---|---|---|---|---|
| **1. Visibility of System Status** | **System status if the Network connection is lost is absent** | Severity : Medium | | | |
| **2.Match between System and Real World** | **Good. No intervention required** | NA | | | |
| **3. User Control and Freedom** | **For a novice user disappearing zoom slider bar can be confusing!** | Severity : Low | | | |
| **4. Consistency & Standards** | **Good. No intervention required** | | | | |
| 5. **5. Error Prevention** | **Good. No intervention required** | | | | |
| **6. Recognition Rather than Recall** | **Navigating by panning and zooming often leads to being lost . Direction of movement is required** | Severity : HIGH | | | |