SNS COLLEGE OF TECHNOLOGY

(An Autonomous Institution)

COIMBATORE-35

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

19CST201 AGILE SOFTWARE ENGINEERING

WebApp Interface Design

- **Introduction :** WebApp interface design should be perfect so it answers three primary questions for the end user.

- Where am I? The interface should
- provide an indication of the WebApp that has been accessed
- inform the user of her location in the content hierarchy.

- What can I do now? The interface should always help the user understand his current options
- what functions are available?
- what links are live?
- what content is relevant?

- Where have I been, where am I going? The interface must facilitate navigation. Provide a "map".
- An effective WebApp interface must provide answers for each of these questions as the end user navigates through content and functionality.

Interface Design Principles and Guidelines

Anticipation (Expectation)

- A WebApp should be designed so that it anticipates the user's next move.
- For example, consider a customer support WebApp developed by a manufacturer of computer printers.
- The designer of the WebApp should anticipate that the user might request a download of the driver and should provide navigation facilities that allow to download latest driver without searching.

Communication

- The interface should communicate the status of any activity initiated by the user.
- Communication can be obvious (e.g., a text message) or displaying a current location of page.

Consistency

- The use of navigation controls, menus, icons, and aesthetics (e.g., color, shape, layout) should be consistent throughout the WebApp.

Controlled autonomy

- The interface should facilitate user movement throughout the WebApp, but it should be in a manner that enforces navigation conventions that have been established for the application.
- For example, navigation to secure portions of the WebApp should be controlled by userID and password.

Efficiency

- The design of the WebApp and its interface should optimize the user's work efficiency, not the efficiency of the developer who designs and builds it or the client server environment that executes it.

Flexibility

- The interface should be flexible enough to enable some users to accomplish tasks directly

Focus

- The WebApp interface (and the content it presents) should stay focused on the user task(s) at hand.

Fitt's Law

- "The time to acquire a target is a function of the distance to and size of the target."

Human interface objects

- A vast library of reusable human interface objects has been developed for WebApps.

Latency reduction

- Rather than making the user wait for some internal operation to complete (e.g., downloading a complex graphical image), the WebApp should use multitasking in a way that lets the user proceed with work as if the operation has been completed.

Learnability

- A WebApp interface should be designed to minimize learning time, and once learned, to minimize relearning required when the WebApp is revisited.

Metaphors (Symbol / Image)

- An interface that uses an interaction metaphor is easier to learn and easier to use, as long as the metaphor is appropriate for the application and the user.

Maintain work product integrity

- A work product (e.g., a form completed by the user, a user specified list) must be automatically saved so that it will not be lost if an error occurs.

Readability

- All information presented through the interface should be readable by young and old.

Track state

- When appropriate, the state of the user interaction should be tracked and stored so that a user can logoff and return later to pick up where she left off.