Interaction Styles

The interaction styles currently in use can be categorised as follows:

- Command-language
- Form Fill-in
- Menu-selection
- Direct Manipulation
- Anthropomorphic / Natural User Interface (NUI)

Command-language (e.g., Unix, DOS)

This was the first interaction style to be adopted widely, and it is still in use, mainly in Unix/Linux systems.

Commands are typically very short in order to minimise key-strokes, and the user must learn both the commands and their parameterised options.

Advantages and disadvantages:

- Powerful and flexible.
- Steep learning curve
- Relies heavily on recall
- Generally suited to complex systems used by trained staff.
- Easier to script than other styles
- Error-rates tend to be high.

Form Fill-in

Form Fill-in interfaces present the user with a series of fields into which data can be entered.

The fields are labelled, so interaction relies on recognition, not recall.

Where appropriate, the user can move between fields using the Tab key rather than a mouse or other pointing devices.

In early systems which used form fill-in, it was often used exclusively throughout the interface. Nowadays form fill-in is normally used in conjunction with other interaction styles.

Advantages and disadvantages:

- Easy-to-use
- Interactivity is limited.
- Generally suited to any application which involves gathering relatively large amounts of data, whether for novice/occasional or expert users.
- Being able to see all the fields at once is an advantage in many applications.

Menu-selection

Menu selection interfaces present a set of options from which the user selects one (or sometimes more).

Selecting an option changes the state of the interface.

The options are labelled, so interaction relies on recognition, not recall.

If the labels are well chosen and the options are appropriately grouped, the amount of learning required is minimal.

Advantages and disadvantages:

- Presents the options available to the user at each stage of interaction.
- Easy to use since it relies on recognition, not recall.
- Not as flexible as command-language or direct-manipulation.
- Generally suited to relatively simple tasks performed by occasional and/or untrained users.
- Can be annoying for experienced users unless 'shortcuts' are provided.

Direct Manipulation

Shneiderman coined the term Direct Manipulation to describe the style of interaction in which objects of interest are represented as visible entities which can be manipulated directly by the user.

Its defining features are:

- Visibility of objects
- Replacement of complex command-languages with actions to manipulate visible objects directly
- Incremental action at the interface with rapid feedback
- Reversibility of actions
- Syntactic correctness of actions (every action is legal)
- Few/no changes of mode

Advantages and disadvantages:

- Powerful and flexible
- Relies on recognition, not recall, so is easy to use.
- Suited to moderately complex and complex systems.
- The amount of training required varies with the application but is usually less than for equivalent command-language systems.
- More difficult to script than command-language systems.

Anthropomorphic

Anthropomorphic is a general term used to describe interfaces that allow natural forms of interaction, e.g.:

- natural language (typed/speech)
- gesture.

Advantages and disadvantages:

- Currently limited in power and flexibility but improving rapidly.
- Work best where the range of communication is constrained, e.g., surgery, air-traffic control, emergency vehicle dispatching, etc..
- Currently suited to the same kind of tasks as menu-selection interfaces (e.g., in voice-mail systems), but may eventually offer the same potential as command-language and direct-manipulation systems.

Interaction Styles in Use

Many interfaces incorporate more than one type of interaction. For example:

- A modern graphical user-interface may use direct-manipulation for most tasks, but...
- some tasks (e.g., changing system settings) may be carried out using a command-language or form fill-in approach.

Web applications have traditionally used form fill-in or menu-selection rather than direct-manipulation.

This is reflected in the set of interface widgets supported by web-browsers.

Many direct-manipulation actions - for example, drag-and-drop - rely on rapid feedback, and this is not possible using server-side techniques.

Until recently, such interactions could only be supported using:

- Java Applets
- Proprietary client-side technologies, such as Flash and Silverlight
- Dynamic HTML (the DOM, CSS and JavaScript), and libraries using these (e.g., JQuery)

In recent years, web browsers have come to be seen as the logical choice of front-end for cloud-based and software-as-a-service applications.

This requires that web browsers support a much wider range of interactivity than in the past - ideally, the full range of interactions used in desktop computing.

HTML5 represents a significant move towards this goal, with a greatly extended set of interface widgets and native support for drag-and-drop.