Multimodal Interaction

Lesson 3
Models for Multimodal Interaction

Maria De Marsico
demarsico@di.uniroma1.it

Model to design

- Identify system borders
- Identify possible system states
- Identify possible responses to external actions
- Anticipate possible evolutions

- Fo r multimodal interaction
  - Each channel identifies its own borders
  - Possible system states are a combination of the states for each channel → How many of them are truly significant?
  - Actions may come from different channels, and responses are not due through the same channels
  - Possible evolutions are more difficult to anticipate and address

Model is to ask questions
Design is to try responses

Why it is important to model software

- Need to identify project components
- Need to evaluate compliance with requisites
- Need to maintain software along time

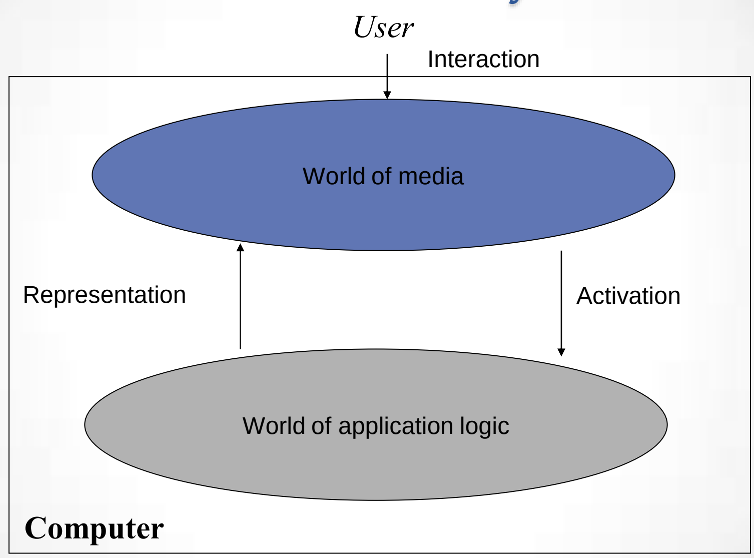
- For multimodal interfaces:
  - Each channel requires a separate component, but communication and syncrhonization among different input modalities must also be supported
  - Requisites may be more difficult to state, and therefore to check
  - Maintainenace may call for sharp separation in order to exploit technological advances at different pace

Why it is important to model interactive software

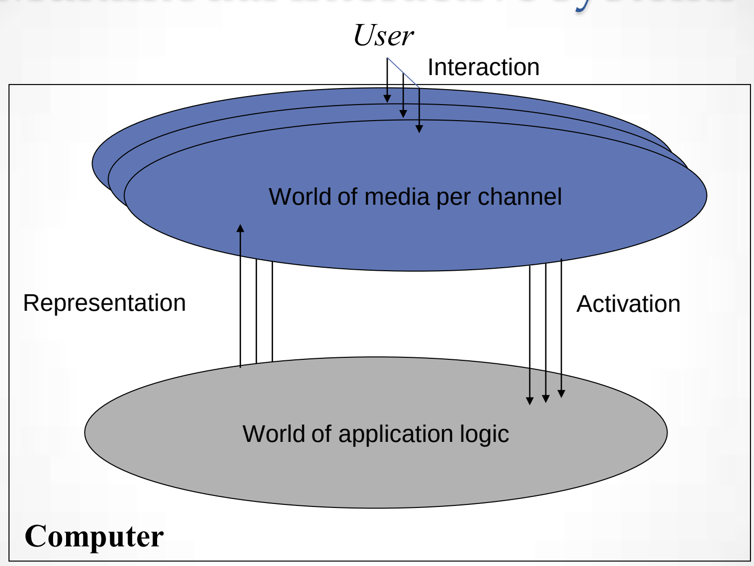
- Different levels of definition must be maintained consistent
- Identify user languages and evaluate the correspondence with interaction languages
- Evaluate the correspondence between interaction tools and application logic

- For multimodal interactive systems:
  - Consistency must also address the use of different channels
  - User languages span different modes and modalities that must be coordinated
  - Application logic can exploit the advantages of multimodality

Visual interactive systems



Multimodal interactive systems



Problems

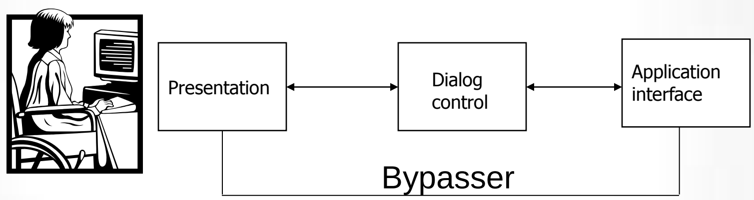
- Relation between representation medium (→media) and interaction medium (→media)
- Correspondence between application logic and interaction logic
- Need to handle representation and activation processes

UIMS

- User Interface Management System: controls the relation between presentation and functionality
- *Defines separation between application semantics (business logic) and presentation (GUI); in this way it improves :*
  - portability – ability to be used on different devices and systems
  - re-usability – reuse of components cuts expenses
  - multiple interfaces – to access the same functionality
  - customizability – for designer and user
- Multimodal interfaces:
  - portability - very complex
  - multiple interfaces - multiple channels
  - customizability – for special needs

Separable Interface

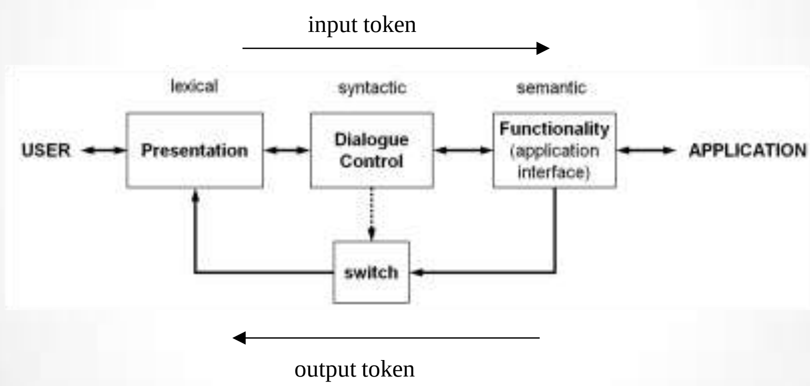
Seeheim Model



Seeheim model

- Developed in early '80 during the Workshop on User Interface Management Systems in Seeheim by a group composed by Jan Derksen, Ernest Edmonds, Mark Green, Dan Olsen, and Robert Spence.
- Not possible at that time to fully evaluate it but …
- … it was a first very important step (as any first step!)
- "… because of Seeheim …
  … we think differently!" (Alan Dix)
- Follows the division among Lexicon, Syntax, Semantics
- Limits:
  - Dialog control is monolithic
  - Bypasser makes formal description difficult

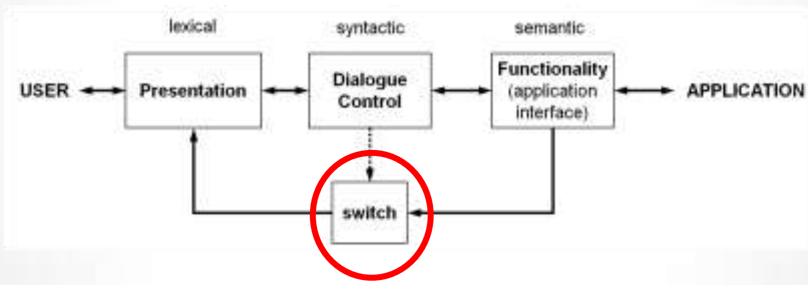
Seeheim model



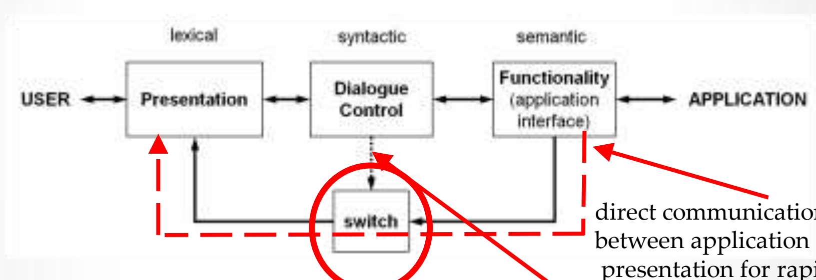
Seeheim model (continued)

- Different kinds of feedback:
  - lexical – mouse movement (the code to express that the system got user's intent to move the focus of action)
  - syntactic – a chosen menu item is highlighted (a syntactic rule says that a selected item must be highlighted)
  - semantic – the sum of numbers on a calculator is changed (application rule)
- Semantic feedback is slower of course
- In many cases, a quicker semantic feedback is needed, e.g. in hand drawing, or in highlighting the waste basket on the desktop when a document is moved nearby, or when large volumes of output data may skip the dialog layer.

Seeheim model (continued)



Seeheim model (continued)



direct communication between application and presentation for rapid semantic feedback

The model confuses logic architecture and implementation

switch o bypass

but regulated by dialog controller

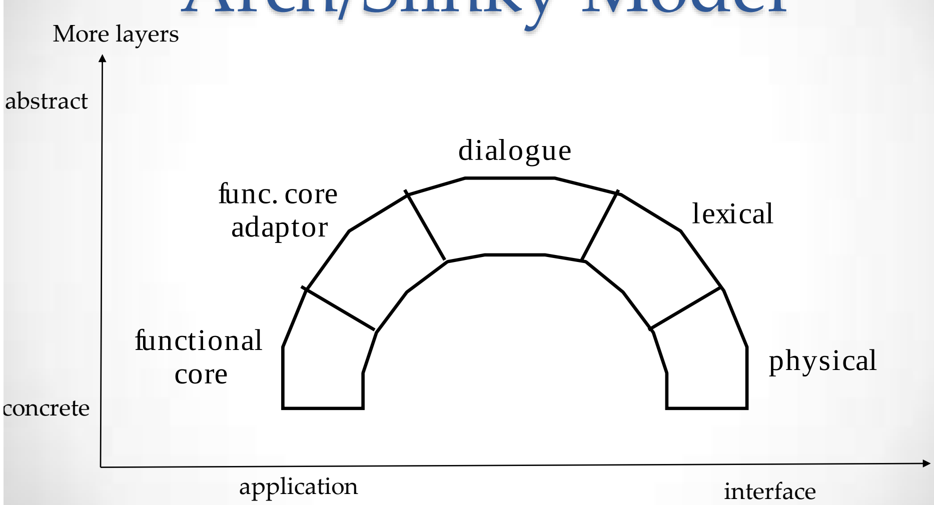
Seeheim model (continued)

- The two main difficulties of Seeheim model :
  – when we change a presentation component, the dialog must be rewritten to adapt it to its features
  – dialog tends to be based on presentation, and presentation must be changed everytime the dialog is changed

BUT

- If we handle each block as a whole,
  - we may provide the same outer layer to different applications (but we have to change the dialog)
  - we may apply the same look and feel to a text editor, a spreadsheet, and so on, as in Microsoft products
  - in this way the user does not have to learn different dialog languages for different applications
  - conversely, we may provide a single application to be implemented behind several different outer layers, so as to allow different companies to adopt the same application with their own corporate interface style

Arch/Slinky Model

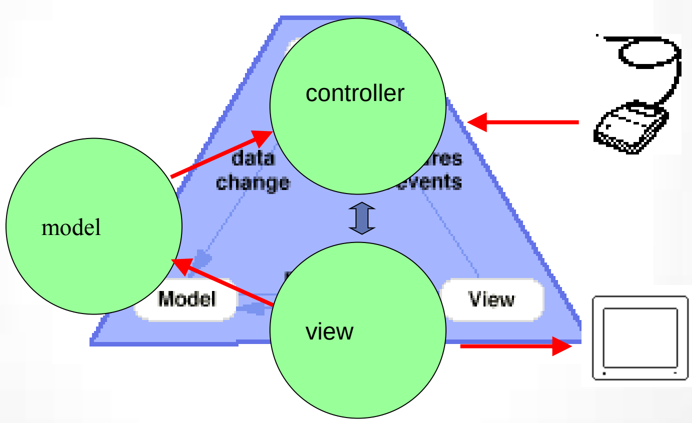


Arch/Slinky Model (cont.)

- More layers! – lexical/physical layers are distinguished
- As in a 'slinky' spring, different layers may be larger (more important) than others in different systems …
- … or in different components
- Limit - is not better than Seeheim in providing indications on the precise content of components or on their design process

Separable interface

MVC (Model View Controller)



Problems in the definition of an interactive system

	Single components	Components coordination
Representation	Identification of the kind of needed component	Layout design
Behaviour	Reaction to user actions and state change	Event propagation and view synchronization

- In multimodal interfaces:
  - the kind of needed component may refer to different channels
  - user actions may happen through different channels
  - layout is not 2D visual but multimodal
  - events may refer to different modalities

Identification of the kind of component

- Domain represenation
  - Information (data) to represent
  - Processes to activate
- Interaction management
  - Generic interaction elements
  - Interface navigation elements
  - Support to specific behaviours

Behaviours to support:

- Process activation
- Information retrieval
- Information feeding

Coordination Policies (for behaviour)

- Policies for enable/disable
- Activation propagatione
- Concurrent activation
- Multimodal interfaces:
  - Intra-channel
  - Inter-channel

Coordination Policies (for presentation)

- Overall interface "Layout"
- Mutual constraints between elements
  - coordinated changes
  - admissible relations among elements
- Consistency with other types of interfaces (which ones???)

Readings

- Roope Raisamo. Architectures for User Interfaces - SoSE tutorial lecture course http://www.cs.tut.fi/~sose/phdcourse08/ArchitecturesForUserInterfaces.pdf
- L. Schomaker, J. Nijtmans, A. Camurri, F. Lavagetto, P. Morasso, C. Benoît, T. Guiard-Marigny, B. Le Goff, J. Robert-Ribes, A. Adjoudani, I. Defée, S. Münch, K. Hartung, and J. Blauert, A Taxonomy of Multimodal Interaction in the Human Information Processing System. A Report of the Esprit Basic Research Action 8579 MIAMI. February, 1995. http://www.ai.rug.nl/~lambert/papers/TaxonomyMultimodalInteraction-RepEsprit-Project8579-MIAMI.pdf
- Laurence Nigay and Joëlle Coutaz, A design space for multimodal systems: concurrent processing and data fusion. Human Factors in Computing Systems, INTERCHI '93 Conference Proceedings, ACM Press, 1993, 172-178.
- Sharon Oviatt, Mutual disambiguation of recognition errors in a multimodal architecture. Human Factors in Computing Systems, CHI '99 Conference Proceedings, ACM Press, 1999, 576-583.