Model-Based Construction and Validation of Safety-critical Interactive Systems

Philippe Palanque
LIIHS-IRIT
University of Toulouse 3, France
http://liihs.irit.fr/palanque

International Summer School on Usability-Driven Software Architecture
Tampere University of Technology, Tampere, Finland
August 22-24, 2005

Structure of the presentations
- Part I (1h)
  - Introduction to interactive systems modeling
  - Introduction to modeling processes
    - More directed towards modeling rather than architecture
- Part II (1h)
  - Interaction techniques and interactive systems modeling
  - State of the art for interactive systems modeling
  - Design rationale and modeling
- Part III (2h)
  - Cases studies
  - PetShop: a plate-forms for engineering interactive systems
  - A roadmap for notations, tools and plate-form for interactive systems

Context
- Solve the same problems as the ones raised in the introduction by Len
  - We can't change that
    - Make some changes easy to perform
  - Lower level detail than Len's proposal on SA
  - Different approach wrt Fabio's but complementary as well
  - Covers the another part of the design process (detailed design) and link with SA
  - Application domain: critical software

Safety Critical Interactive Systems
- Safety Critical Systems
  - Software Engineers
    - System centered
      - Reliability
      - Verification / Proof
      - Waterfall model / structured
  - Usability engineers
    - Usability
      - Human factors
      - Task analysis & modeling
      - Evaluation
      - Iterative process /
      - Prototyping
    - A roadmap for notations, tools and plate-form for interactive systems

Gaps
- DO NOT trust "HCI designers" (beyond innovation)
  - They will ask for "unfeasible" things
  - They will change their mind faster that one can implement
  - They will ask for trust (when no argument is available)
  - They usually do not care about safety issues
  - They are human being
- DO NOT trust "developers" (beyond hacking)
  - They don't know what to do
  - They "mostly" don't know how to do it (e.g. ATC)
  - They have a partial view of the problems
  - They usually do not care about usability issues
  - They are human being
- Control over the process and people is required

Bridges over the Gaps
- The same applies to HF
Soft. Engineering Contribution

Waterfall Process

Requirements analysis and definition
System and software design
Implementation and unit testing
Integration and system testing

Extended Waterfall Process

Architectural design
Detailed design
Coding and unit testing
Integration and testing
Operation and maintenance

Development Process

Specification
Preliminary Design
Detailed Design
Validation
Integration
Unitary Tests
Implementation
Deployment

Requirements

Evaluation

Identification and solving of risks
Define objectives
Define alternatives and constraints
Plan for next phases
Develop and verify next product

Spiral Development Process

Evaluate alternatives
Identify and solve risks
Risk analysis
Risk analysis
Plan development
Plan development
Plan integration and tests
Plan integration and tests
Planification analysis
Planification analysis
Planification of the life cycle
Planification of the life cycle

Incremental Prototyping (RAD)

Do not even think about reliability or usability

HCI Contribution

STAR Development Process

Implementation
Task analysis/functional analysis
Conceptual design/formal design

The star life cycle (adapted from Hix and Hartson, 1993).
HCI Contribution
Throw Away Prototyping

Evaluation with
Users

You can’t be right !!!

Alan Dix HCI Book

Example 1: Range Slider and
Starfield Display

Beyond Innovation
(Ahlberg & Schneiderman 94)

The Range Slider

Abstract Data Type  Range
▪ ValMin, ValMax, ValCurMin, ValCurMax
▪ Domain = \{ n ∈ \mathbb{N} \mid n \geq ValMin, n \leq ValMax \}
▪ Card(Domain) = ValMax − ValMin + 1

Usability evaluation 1:
relationship values and pixels
▪ If ValCurMax = ValCurMin
  ▪ 3 different presentations (lift)
  ▪ 2 different manipulations

Usability evaluation 2: relationship
values and pixels
▪ Various presentations for the same value
Usability evaluation 3: relationship values and pixels

- If Card(Domain (Field)) > nb pixels of Range Then
  - Impossible to select some values
  - Rendering of some values impossible (same rendering of the slider for different values)
- Example range between 100 and 105 in database Movies

Usability Evaluation 4

- The user interface is not predictable with respect to low level interactions
  - No rendering that the lift is currently used (after a down on the lift)
  - ...  
- The user interface is not predictable when resizing the windows
  - Depends on the number of pixels but only for length
  - ...

Example 2: User Interface to Data-centered software

Computer-based data-oriented system
- We have the following data-base
  - Client 1,1 \( \text{Is in Charge} \) 1,n Sales man
- Design a menu structure for managing this information

Data-oriented design

- Software engineering
  - Menus tree reflects the database structure
  - Structuring is done according to functions similarities (in terms of programme code)

Output from Task Analysis

- « I spend my day at work entering information for new clients and I must associate them with the more suitable sales man (according to various criteria) … sometimes I receive an invoice from a sales man recently recruited. In such a case, I must create both the new sales man and the client. Sometimes a sales man quit the company. Usually they leave the company for another one and the clients follow them. I then have to delete the sales man and all its related clients … »
  - Design a menu structure for these tasks
Some Maths
- Number of clients: 10000
- Number of sales man: 100
- Number of clients per sales man: 100
- Deleting clients: 100 actions
  - Remembering Sales man ID
  - Searching for the client according to the ID
  - Deleting
  - Confirming the deletion

New menu structure
- Menu structure according to task models

Things are still problematic
- There is no
  - Menu for deleting clients (without its associated sales man)
  - Menu for client modification
  - ...
- There is
  - Access to clients through sales man deleting
  - Display of clients information independently with the ones of the related sales man
  - ...

Example 3: Gaming

Interlude: A simple game
- The numbers from 1 to 9 are initially available to any of the two players.
- The players play in turn.
- At each turn, the player chooses one of the remaining numbers (thus making it unavailable). If the player possesses three numbers that add up to exactly 15, he wins the game.
- At some point a player can have more than three tokens

Task analysis: Goals (1)
- Both players share a common goal: "win the game". Sub-goal "if I cannot win, don't let the other player win".
- You MUST play once the game with you neighbour

\[ U = \frac{1}{F} \]
Task analysis: activity (2)

- Choose a number among the remaining numbers (strategy for selection, check whether other player is about to win).
- Evaluate if win the game. If win then end.
- Background activity
  - Remember the numbers I have already taken,
  - Remember the remaining numbers (or alternatively remember the numbers taken by my opponent),
  - Remember who is next.

From Tasks to System: your solution

![Diagram of Basic System]

Task Modelling

![Diagram of Task Modelling]

From Tasks to System (2) Dialog

![Diagram of From Tasks to System (2) Dialog]

Improved System Model

![Diagram of Improved System Model]

So What?

- Designing for large number of people
- Presentation is only one part of the problem
Aims

- User interfaces for safety critical applications
  - User Centered Design, Usability Engineering
    - Involve users at every stage of the design
  - Prototype and validate with them in iterative cycles
- Use of Formal Models
  - The user interface requires the same quality as the rest of the software
    - Completeness (model the entire UI)
    - Reliability, verification, proof
- Reconcile Formal Methods with Prototyping
- Reconcile Prototyping with Formal Methods

Formal Modeling of Interactive Systems

Some vocabulary

- Formal models for Interactive Systems
- Basic Principles of ICOs
- Short Introduction to Petri nets modeling

Interactive Systems

"Classical" Design Process for Software Systems

What is a "Model"?

- Concepts relationships between concepts
- Example: Entity Relationship Model
  - Concepts: Entity Types, Relationship Types, Attributes, Domain of Values, Identifiers, ...
  - Relationships between concepts: "A Relationship type is a sub set of the Cartesian product of two Entity Type"
  - An entity e1 cannot have more than one relationship with another entity e2 through the same relationship type

What can we expect from a Model?

- Completeness (we can express all the information we need to)
- Consistency (Contradictory elements cannot be expressed)
- Generality (Independent from a given application or application domain)
What is a Formalism?

- A set of conventions for representing the concepts of a Model
  - Lexical elements (graphical or textual)
  - Concrete Syntax (separators, terminators, ...)
- A formal definition of these conventions (otherwise just a notation)

What can we expect from a Formalism?

- Expressiveness
- Conciseness
- Closeness of the representation to the application domain
- Completeness wrt the Model (counter example: Graphical formalism of the Entity Relationship Model) no ad hoc adjunctions
- Usability

"Things must be as simple as possible BUT not more simple" (Einstein)

"ADT is a formal description technique dedicated to the specification of abstract data types" (Lamport)

"The only difficult problem the author solved was understanding his own notation" (Lamport)

Bug: With a "Formalism" we build "models" – Modelling

- model: ideal representation of a given situation form the real world restrained to the concepts covered by the Model
- Goal: analyze the model to draw conclusions about the actual state of the real world
- Meta-model: model representing the concepts of the Model using one of the formalisms of the Model (done for E/R and UML using class diagrams)

What and why Modeling Systems?

- An abstract description of the system
  - independent from the implementation
  - that do not deal too early with details
- Describe what are the outputs of the system according to the inputs
- To allow discussions between the various actors (at a time, along the design process)
- to store results of discussions

What is a System Model for Interactive Systems?

- Represents the system data and actions
- Represents behavior of the system
  - what actions are offered by the system
  - when an action is available (according to the state of the system)
  - what is the effect of an action on the state of the system
- Represents both how the system is presented to the user and how the user interacts with it

Why Modeling Interactive Systems Formally?

- To cope with the complexity
- To avoid a human observer (checking models)
- To avoid a human translator (writing code)
- To reason (verification, validation)
- To meet three basic requirements
  - Reliability: generic and specific properties
  - Efficiency: performance of the system, the user (workload, ...) and the couple (user, system) tasks
  - Usability
Fundamental Elements for Formal Methods for Interactive Systems

- Describe both state and events
- Describe both data structure and control structure
- Provide structuring mechanisms (80% of the code is dedicated to UI [Myers 90])
- Take into account concurrency (multimodal systems)
- Deals with temporal aspects (temporal windows)
- Do it formally (it is easier to prove than to test)

Overview of the Interactive Cooperative Objects

- Petri nets for behavior OO approach for structuring
- Set of cooperating classes
- For each class
  - Behavior (using OO Petri net)
  - Software Services (availability)
  - States (distribution AND value of tokens=objects references)
  - Presentation
    - User services
    - Rendering function
    - Activation function

Basic Example

- One traffic light
- Three different colors
- Behavior is the following:

```
Behaviors
```

Basic Example

- One traffic light
- Three different colors
- Behavior is the following:

```
Behaviors
```

Basic Example

- One traffic light
- Three different colors
- Behavior is the following:

```
Behaviors
```

Basic Example

- One traffic light
- Three different colors
- Behavior is the following:

```
Behaviors
```
Basic Example

- Two traffic lights

Basic Elements

- System described as:
  - State variables (places)
  - State change operators (transitions)
Basic Elements

- System described as:
  - State variables (places)
  - State change operators (transitions)
  - Pre conditions to state changing (arcs)

Tokens

- 1 token in place P1
- Transition T1 is fireable
- Transition T2 is not fireable

- If T1 is fired.....

Concurency

- 1 token in P1
- T1 is fireable

- If T1 is fired...
Concurency

- 1 token in P2
- 1 token in P3
- T1 produced a token
- Now, T1 and T2 are fireable
- If T2 is fired...

Concurency

- 1 token in P3
- 1 token in P4
- T3 is fireable
- Note, T4 is not fireable because there is no token in P5
- If T3 is fired...

Concurency

- 1 token in P4
- 1 token in P5
- T4 is now fireable because P4 and P5 contain a token
- If T4 is fired...

Rendez-vous, P4 must contain a token and P5 must contain a token for T4 to be fireable

Concurency

- 1 token in P6
- T4 consumed 1 token

True Concurrency

- Duration
  - t1=5s; t2=3s; t3=7s; t4=5s
- Interleaving semantics
  - Execution time t1+t2+t3+t4= 20s
- True concurrency
  - Execution time t1+Max(t2,t3)+t4= 17s

Choice/OR

- 1 token in P1
- T1 and T2 are fireable
- We can chose which to fire
Re-initialisation
- 1 token in P1
- T1 and T2 are fireable
- If T1 -> then T5 is fireable, then we return to 1 token in P1

More than 1 token in place
- 3 tokens in P1
- T1 is fireable
- If T1 is fired…

More than 1 token in place
- 2 tokens in P1
- T1 is still fireable
- 1 token in P2
- 1 token in P3
- T2 and T3 fireable
- If T1 is fired…

Inhibitor Arcs
- For T1 to be fireable, P1 must contain no tokens & P2 must contain at least 1 token

Inhibitor Arcs
- T1 still not fireable
- If T2 is fired again…
Inhibitor Arcs

- T1 is now fireable

Test Arcs

- T1 & T2 are fireable
- If T1 is fired...

Test Arcs

- 2 tokens remain in P1

Time and Petri nets

- Time in places
- Time with transitions
- Time on arcs
- Stochastic time
- Time related with tokens value
  - ... no time to talk about time

Time in PetShop

- Modeling
  - A duration is associated to some transitions
- Behavior
  - As soon as a token is set in a place P a Timer is armed and associated to the token
  - When the timer expires, if the token is still in P then the timed transition is fired
- Allows for modeling WatchDogs
- Semantics of Generalized Stochastic Petri nets different from classical Timed nets

Marking & Locality of Transitions

- Marking
  - Distribution of tokens in the Petri net
  - Defines the state of the Petri net
  - Each different distribution represents a different state
- Locality of transitions
  - The firing of a transition impacts only the input places (token(s) removed) and the output places (token(s) produced)
Interactive Exercise

Fuel Pump

ON OFF

Motor on = fuel flows

Motor off = no fuel flow

No Fuel Flow

Fuel Flows

Interactive Exercise Hints

- Identify the states
  - one state ~ one place

- Identify the actions
  - one action ~ one transition

- Initial state: Motor off & no flow

Fuel Pump Solution

Extending State Space: High-Level Petri-nets

- Tokens hold values
- Places are typed
- Pre-conditions on transitions
- Variables on arcs (main point is the flow of tokens, instead of actions)
- Unification on arcs

Tokens with value (1-uptet)

Tokens with value (2-uptets)
The Impact of High-Level Petri nets

Philosopher (1/5)

- One philosopher
- Eating or thinking
  - Starts eating by
    - First taking right chopstick ($c_1$)
    - Then he takes left chopstick ($c_2$)
  - Eats
  - Replaces right chopstick ($c_1$)
  - Replaces left chopstick ($c_2$)
  - Thinks

Philosopher (2/5)

- Starts eating by
  - First taking right chopstick ($c_1$)
  - Then he takes left chopstick ($c_2$)
  - Eats
  - Replaces right chopstick ($c_1$)
  - Replaces left chopstick ($c_2$)
  - Thinks

Philosophers (3/5)

- Same behaviour for four philosophers
- They share the four chopsticks

Philosophers (4/5)

- Same behaviour for four philosophers
- They share the four chopsticks

Overview of Cooperative Objects

- Beyond High-level Petri nets: Object-Oriented dialect of Petri nets
  - Object-Oriented concepts to deal with static / structural aspects of the system, and to cope with complexity
  - Petri nets to deal with dynamic / behavioral aspects
- Two strategies of integration
  - Petri nets inside objects
  - Objects inside Petri nets
- Tool support (PetShop)
  - Editing, Execution, Analysis, Prototyping
Objects inside Petri nets
- The net models the system as a whole
- Tokens in the net are objects (instances of classes)
- The transitions create / delete or call methods on objects

Petri nets inside objects
- The system is modeled as a set of cooperating objects
- The behavior of each object is modeled as a Petri net

Cooperative Objects
- Integrate these two approaches
  - Behavior of a class described as a high-level Petri net (the Object Control Structure, ObCS)
  - Tokens in the net are references to other Cooperative objects in the system
- Support dynamic instantiation, inheritance, polymorphism …
  - All these constructs have a denotational semantics in terms of high-level Petri nets

Dynamic Instanciation

Extensions to ICOs
- New asynchronous multicast communication mechanism
  - events production (trigger)
  - events consumption (catch)
- Quantitative temporal information (temporal window) reuse of previous work in Petri nets theory
- Methodological decomposition
  - Component-based modeling
  - Client-server communication

The Model-View-Controller Design Pattern
- The user interface triggers transitions in the net
- The net’s state is rendered on the UI
Exemple the 4 seasons application

- One user only
- Critical application
- ☺

Scope of Interactive Cooperative Objects

- MVC pattern
- Communicating Petri nets

Fonction d’activation

- Plane Manager
- Menu

Fonction de rendu

- Plane Manager
- Plane
- Menu
Interaction Techniques and the Impact on Models

- Specification of WIMP user interfaces
  - Non-interactive part
  - Dialogue part
  - Presentation part
    - Activation
    - Rendering embedded in the widgets
  - Static widgets (their number and nature is predefined)

- Rendering for post-WIMP user interfaces
- Description of dynamic behavior in UI
  - Dynamic instantiation of interactors
  - Dynamic behavior of interactors
    - subscription to events
    - external presentation
- Done thanks to the case studies
  - Methodological aspects of specification
  - Modifiability and reusability of specifications

Multimodal Interaction in ATM System

- Rafale Fighter (future versions)
  - 2 joysticks
    - 33 buttons
    - Up to four positions for some buttons
  - Voice recognition
  - Tactile screen
  - Head-Up Display (head position tracker)
  - 3D sound
  - Multimodal interaction already available

Related Work MB-UI notations and tools
Dealing With Multimodal Interfaces

- Only one example of interaction technique
- Proposed in the early 80's (Bolt's system)
- Still not available
  - In toolkits
  - In the Software Engineering community (see Graph Grammar paper at ICSE 2001)
- BUT
  - highly demo-able
  - going into safety critical systems
- Interestingly it is still the same for post-WIMP interfaces

Engineering Multimodal Safety Critical Systems

- Specificities
  - Relating input devices to other software components
  - Concurrency
  - Quantitative temporal aspects
  - Fusion of several input devices
  - Fission mechanisms
- Propositions
  - Extending architectures
  - Extending formal description techniques
  - Going from specification to implementation

Proposal: Definition and Use of Formal Notation

- The user interface requires the same dependability as the rest of the software
- Completeness (model the entire UI)
  - the complex parts must be dealt with too
  - the more complex the UI the more likely the notation is not be able to deal with it
- Verification techniques

Two Models One Interface (Fabio)

- Low-level interaction
  - "Continuous"
  - Information Flow
- Higher-level interaction (dialogue)
  - Event-based
  - State based

Relation to ARCH (Bass et al 91)

ICon: Input Configurator a model (ICom) and a tool (ICon)

ICon Tool: editor and simulator

Some examples

Getting rid of ICon
- Model low level interaction using ICOs
- Link the models to input devices drivers
- Model events transformation (transducers)

A demonstration?

Interactive Example
- Model the input device (the mouse for instance)
- Model the transducer that deals with
  - Clics
  - Double clics
  - Drag and drop
- Go from one mouse to multimodal mice

A Small Example
Modeling Click behavior

A Small Example

A Small Example

Adding Time

A Small Example

Taking Movements into account

A Small Example

Combining Click behavior

Compatibility of models

- Is the mouse able to perform everything that is said in the Double-Click models?
- Is the Double-Click model exploiting all the characteristics of the mouse model?
- Compatibility can be proven by verification on accepted languages inclusion
How to go to 2 mice ???

- Reusability: Can the one-mouse-model be reused
- Conciseness: Shouldn't be much more complex than the one-mouse-model
- Completeness: everything that has to be expressed can be expressed

Two Mice

Multimodal use of 2 mice

Relationship with ARCH

PetShop Fundamental Principles

- Highly interactive tool support for the ICO notation
  - Formal methods can be usable (and useful beyond n! calculation)
- "Shortening the Path from Specification to Prototype" ... to practically nothing
- Model-Based
  - The specification model is embedded and interpreted at run-time
  - WYMIWYR (what you model is what you run) to reduce gaps in interpretation (Norman's model)
  - No "Modes" : no "automation surprises"
  - The model is editable and interpretable at any time
  - Allows for interactive prototyping of new interaction scenarios
Basic idea: Pilots use PCs at home so we have to provide them with the same kind of interaction.

Yes but ...

Why?
- Incompetents?
- Budget shortage?
- ...

Next users? Software engineers
Need for a Standard for Interactive Cockpits: ARINC 661
- Written by the main operators in aeronautics
  - Thales, Collins, Honeywell, Airbus, Smith, BAE, Barco ...
- Trying to reach several objectives (next slide)

ARINC 661 Objectives
(from Paul J. Prisaznuk AEEC General Session AEEC General Session October 15, 2003)
1. Standardize Graphical User Interface (GUI)
2. Standardize interface between User Applications (UA) and Cockpit Display System (CDS)
   - Define table-driven interface definition
   - Bi-directional exchange
3. Expand flight deck crew and display interactivity
   - Extension of FMS interface
   - Growth for CNS/ATM
4. Reduce total cost of avionics ownership
   - Ability to modify cockpit display functionality
   - Ability to modify User Application software

Current State of ARINC 661
- AEEC PP661 adopted October 2001/published April 2002
  - Met Airbus critical need requirement
- Supplement 1
  - Vertical map display capability
  - Eight new widgets added
  - Airbus A380 CDS versus needs for future CDSs
  - ARINC 661-1 published June 26, 2003
- Supplement 2
  - Draft 1 published 1st September 2004
  - Changes to ARINC 661 necessary for the Airbus A380 (NextFocusedWidget) and Boeing 7E7 cockpit display system development
  - Seven new widgets (57 widgets in total)
  - Addition of state diagrams for interactive objects (p209)

ARINC 661 Principles
- Client-server
- Very similar to work in HCI
  - IBM CUA 89, OSF/MOTIF, ...
  - Xwindow

Basic Functioning of applications compliant with ARINC 661 specification
- Communication principle
  - Crew members -> Actions -> Monitor system 
  - Monitor system 
  - User Application

ARINC 661 Widgets
- CUA-like widgets “buttons, check box, edit boxes, labels, scroll lists, …”
- Specific to cockpits: Map management, masks, AESS bitmap …
Proposal: Definition and Use of Formal Notation

- The user interface requires the same dependability as the rest of the software
- Completeness (model the entire UI)
  - The complex parts must be dealt with too
  - The more complex the UI the more likely the notation is not be able to deal with it
- Verification, validation, certification, ...

Overview of Interactive Cooperative Objects

- Set of cooperating classes
- For each class
  - Behavior (Petri nets)
  - Services (availability)
  - State (distribution and value of tokens)
  - Presentation
    - Activation (how users’ actions on the input devices trigger systems methods)
    - Rendering (how state changes are presented to the users)

Extensions to ICOs

- New asynchronous multicast communication mechanism
  - Events production (trigger)
  - Events consumption (catch)
- Quantitative temporal information (temporal window) reuse of previous work in Petri nets theory
- Methodological decomposition
  - Component-based modeling
  - Client-server communication

Outline

- Introduction
- ARINC 661 specification
- The basics of the ICO formalism
- Modeling ARINC 661 components
- Modeling ARINC 661 application
  - User Applications
  - Server
  - A case study
    - The MPIA
    - A demo
- Conclusion and perspective

Basic Functioning of applications compliant with ARINC 661 specification

- Communication principle
  - Actions
  - Events
  - Modules
- Where do we apply formal description
  - (Server not yet finished)
  - Widgets
  - User Application
Formal Description of a "simple" widget:
ARINC 661 PushButton p.98-101

- Informal presentation
- Formal Description of the PushButton
  - CORBA-IDL Interface
  - Behavior
  - Activation and Rendering functions
- Thales CDS Look & Feel

PushButton : Description in ARINC 661

- Parameters table, Creation structure table
  - No impact on behavior
- Event structure table

<table>
<thead>
<tr>
<th>Event structure</th>
<th>Size</th>
<th>Value/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EventId</td>
<td>16</td>
<td>A661_EVT_SELECTION</td>
</tr>
</tbody>
</table>

- Run-time modifiable parameter table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Size</th>
<th>Parameter Idn</th>
<th>Type of structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>Uchar</td>
<td>8</td>
<td>A661_ENABLE</td>
<td></td>
</tr>
<tr>
<td>Visible</td>
<td>Uchar</td>
<td>8</td>
<td>A661_VISIBLE</td>
<td></td>
</tr>
<tr>
<td>LabelString</td>
<td>String</td>
<td>{32}+</td>
<td>A661_STRING</td>
<td></td>
</tr>
<tr>
<td>StyleSet</td>
<td>Ushort</td>
<td>16</td>
<td>A661_STYLE_SET</td>
<td></td>
</tr>
</tbody>
</table>

PushButton : Interface CORBA-IDL

- For each attribute ATT modified at run-time, there is a method setATT
- For PushButton
  interface A661_PUSH_BUTTON {
    void setEnable(in char A661_ENABLE);
    void setVisible(in char A661_VISIBLE);
    void setStyleSet(in short A661_STYLE_SET);
    void setLabelString(in string A661_STRING);
  }
- Each of these method is translated into a part of the formal description

PushButton : Activation and Rendering Functions

- Rendering function

<table>
<thead>
<tr>
<th>Object</th>
<th>Event</th>
<th>Rendering method</th>
</tr>
</thead>
<tbody>
<tr>
<td>LabelString</td>
<td>Enter/Leave</td>
<td>Default/NC</td>
</tr>
<tr>
<td>Enabled</td>
<td>Enter/Leave</td>
<td>Default/Show/Disabled</td>
</tr>
</tbody>
</table>

- Activation function
  - No activation function per se
  - Triggering of methods is done by method calls from the server (not events from the user interface)

Example of behavior of the Combo-Box2

Einstein: "Things should be as simple as possible but not more simple"
Behavior Highlighted from the Formal Specification

A Case Study: Multi-Purpose Interactive Application
- A simple application highly interactive
- From Topjet project based on Look & Feel of Thales CDS

Description of the User Application
- Formal Description in four parts
  - Presentation
  - Behavior
  - Activation Function
  - Rendering Function
- Coupling with Thales KIDS (CDS) Server
  - JAVA/JNI interface with existing server
- Process and models tested on the MPIA
  - 4 pages: MAIN, WXR, GCAS et COND

Informal Description of the MPIA application: WXR Page (weather radar)
- The working mode
- The tilt selection mode: AUTO or MANUAL (default)
The CTRL push-button allows to swap between the two modes
- The stabilization mode: ON or OFF
  - The CTRL push-button allows to swap between the two modes
  - The access to the button is forbidden when in AUTO tilt selection mode
- The tilt angle: a numeric edit box permits to select its value into range [-15°; 15°]
  - Modifications are forbidden when in AUTO tilt selection mode

Formal Description of the MPIA application: Object MAIN

Formal Description of the MPIA application: WXR Page (weather radar)
Formal Description of the MPIA application: WXR Page

- Presentation

- Activation Function

<table>
<thead>
<tr>
<th>Widget</th>
<th>Event</th>
<th>Rendering Method</th>
<th>Test Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode Selection</td>
<td>MODE_SELECTION</td>
<td>enters showModeSelection(ms)</td>
<td></td>
</tr>
<tr>
<td>Tilt Angle</td>
<td>TILT_ANGLE</td>
<td>enters showAngle(angle)</td>
<td></td>
</tr>
</tbody>
</table>

- Rendering Function

<table>
<thead>
<tr>
<th>Widget</th>
<th>Event</th>
<th>Rendering Method</th>
<th>Test Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF Check Button</td>
<td>A661_EVT_STATE_CHANGE</td>
<td>setEnable Off</td>
<td></td>
</tr>
<tr>
<td>STBY Check Button</td>
<td>A661_EVT_STATE_CHANGE</td>
<td>setEnable Stdby</td>
<td></td>
</tr>
<tr>
<td>TST Check Button</td>
<td>A661_EVT_STATE_CHANGE</td>
<td>setEnable Tst</td>
<td></td>
</tr>
<tr>
<td>WXO Check Button</td>
<td>A661_EVT_STATE_CHANGE</td>
<td>setEnable WXo</td>
<td></td>
</tr>
<tr>
<td>WXAC Check Button</td>
<td>A661_EVT_STATE_CHANGE</td>
<td>setEnable WXa</td>
<td></td>
</tr>
<tr>
<td>AUTOP Push Button</td>
<td>A661_EVT_SELECTION</td>
<td>setEnable switchAUTO</td>
<td></td>
</tr>
<tr>
<td>Stab Push Button</td>
<td>A661_EVT_SELECTION</td>
<td>setEnable switchSTABILIZATION</td>
<td></td>
</tr>
<tr>
<td>Angle Text Field</td>
<td>A661_EVT_STATE_CHANGE</td>
<td>setEnable changeAngle</td>
<td></td>
</tr>
</tbody>
</table>

Benefits related to modeling:
- Define entirely the behavior of components
- Easier early specification of the application
- Verification of expected properties (mainly related to behavior, accessibility, reinitialisability, liveness, ...)
- Is compatible with previous server implementation (CDS in a simulation environment)
- Makes validation of the application possible
  - At least one widget is available on the UI
  - All the widgets useful wrt to the current flight phase, task, ... are available
  - Come back to initial state in at most 2 interactions
- Makes behavioral verification of extant servers possible

Formal Description of the MPIA application: GCAS Page (Ground Anti collision System)

- Behavior
Conclusion

- A notation for the description of behavioral description of interactive cockpit applications
  - From basic interactive components
  - To layers
  - To User Applications
- Applied to a simple but real User Application MPIA
- Formal verification and performance evaluation as a first level evaluation (not presented)
- Support testing activities using accidents/incidents reports

Future Work

- Carry on with ARINC 661 formal description of components and integration with Thales CDS
  - Provide additional support for testing and verification
  - Integrate within a "real" cockpit application
- Technological trends: From OO modelling and analysis to component models
  - Cooperative Objects extended to CompoNets
  - ICO extended to ICopoNets
- Extensive use in several application domains related to new and innovative interaction techniques
  - Future versions of RAFALE fighter cockpit (Thales)
  - Ground Satellite Control room (CNES)

Case Study Military Aircraft Multimodal Cockpit

Introduction to the case study
Dialogue modelling and modification

- 2 joysticks
  - 33 buttons
  - Up to four positions for some buttons
- Voice recognition
- Tactile screen
- Head-Up Display (head position tracker)
- 3D sound
- Multimodal interaction already available

The ICO notation in the framework of INTUITION

- ICARE-based multimodal input description
- Activation
- Place Event Rendering method
  - NoMark Token_entry <coord, ger, most> most.process_event(1012);
  - Token_entry <coord, ger, most> most.process_event(1013);

PetShop

- Highly interactive tool support for the ICO notation
  - Formal methods can be usable (and useful beyond n! calculation)
  - “Shortening the Path from Specification to Prototype”... to practically nothing
- Model-Based
  - Specification model is embedded and interpreted at run-time
    - WYMIWYR (what you model is what you run) to reduce gaps in execution & interpretation
    - No "Modes", no “automation surprises”
    - The model is editable and interpretable at any time
    - Allows for interactive prototyping of new interaction scenarios
Case Study

- "Mark" command
  - Informal description
    - The pilot looks at a position on the ground and triggers the "Mark" command (multimodal input specified with ICARE)
    - The geographic position designated by the pilot’s gaze is visualized from now on (multimodal presentation specified with MOST) until the pilot triggers the "Clear mark" command

Software architecture

Formal modeling

- Provide complete and non ambiguous answers to questions such as:
  - What if the pilot designates an "impossible" position (such as looking at the sky) ?
  - What if a legal position is currently marked, and the pilot designates another illegal position ?

One possible formal model of the "Marking" dialogue

Alternative dialogue model

Designing Interactive Systems Rationally

Assessing quality attributes: slide 7. Jan session 2 Quantitative versus qualitative assessment
Issues

- Design methods
  - RUP, OViD, eXtreme Programming
  - Mefisto
- Limitations
  - Methods lead to **one** solution
  - Neither traceability of options nor choices
- ESARR 6 (Eurocontrol) and DO-178B
  - Recommend the use of DR
  - Provide no approach nor tool

Design rationale

- Design rationale supports
  - Systematic proposal of several alternatives
  - Evaluation
  - Option and choice argumentation
- Objectives are to
  - Improve quality of interactive systems
  - Support reuse of information beyond
    - Program code
    - Class hierarchies

Design rationale side effects

- Supports collaboration
  - Graphic designers, customers, human factors, programmers
  - Build consensus
  - Encourages participation
- Supports knowledge transfer
  - Explain system to new comers [Karsenty, 96]
- Supports maintenance
  - [Conklin, 1989]

State of the art

- IBIS (Issued-Based Information System) [Kunz & Rittel, 1970]
  - Not well structured
- PHI (Procedural Hierarchy Language) [McCall, 1991]
  - Based on IBIS
  - Difficult to extract relevant information
- DRL (Decision Representation Language) [Lee, 1991]
  - Vocabulary too precise
- QOC (Questions, Options, Criteria) [MacLean, 1991]
  - Simple / comparison
  - Potential inconsistency
- DQN (Design Question Notation) [Bramwell, 1995]
  - Formal notation (require scientific background)
  - Based on QOC

Summary

<table>
<thead>
<tr>
<th></th>
<th>IBIS</th>
<th>PHI</th>
<th>DRL</th>
<th>QOC</th>
<th>DQN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notation</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
<td></td>
<td>ok</td>
</tr>
<tr>
<td>Evaluation degree</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Comparative evaluation of alternatives</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>ok</td>
<td>ok</td>
</tr>
<tr>
<td>Hierarchical</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
</tr>
<tr>
<td>Easy to write</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
</tr>
<tr>
<td>Easy to read</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
<td>ok</td>
</tr>
<tr>
<td>Specific to interactive systems</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Our hypothesis: Why is design rationale not used?

- Inadequate coupling of notation / tool
- Unrelated approaches / methods
- Not linked with interactive systems engineering
- Not linked with software architecture
- Ratio costs / benefits
  - Additional cost (short term)
  - Hard to see/perceive benefits
Objectives of the work

- Integrate design rationale into a design method
  - Define how to use design rationale
- Provide a notation heavily connected to HCI knowledge
  - Task modelling, scenarios, user models, ...
  - Interactive software architecture
- Provide tool support
  - Archive, exploit and visualise information

Presentation outline

- Introduction
- Related work (QOC notation)
- Contributions
  - TEAM notation
  - DREAM tool
- Case studies
- Conclusion & perspectives

QOC – Interaction technique

Question

Options

Criteria

Click on the flight

Move over the flight

Select the flight from the list

Performance

Error Management

Cognitive load

How (which interaction technique) to select a flight?

QOC – Meta model

TEAM

- TEAM
  - Traceability, Exploration and Analysis Model
- Extension of QOC
- Special emphasis for interactive systems
  - Software architecture
  - System models
  - Task models
  - Scenarios,

Presentation outline

- Introduction
- QOC notation
  - TEAM notation
- DREAM tool
- Case studies
- Conclusion & perspectives
TEAM

- Tackles some known usability problems with QOC [Shum, 1991]
- Type of links: 5 possible values
- Criterion weighting: 5 possible values
- Arguments related to all entities
- Directional links questions / questions

Question 1 → Question 4
Question 2 → Question 4
Question 2 → Question 3

TEAM – ARCH

- Classification of questions

TEAM – Task models

Which currency converter to use?

TEAM – User models

<table>
<thead>
<tr>
<th>Task</th>
<th>Type of task</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Know value</td>
<td>Cognitive</td>
<td>25 ms</td>
<td>170 ms</td>
</tr>
<tr>
<td>Delete</td>
<td>Motor</td>
<td>70 ms</td>
<td>360 ms</td>
</tr>
<tr>
<td>Enter 1</td>
<td>Motor</td>
<td>70 ms</td>
<td>360 ms</td>
</tr>
<tr>
<td>Enter 1</td>
<td>Motor</td>
<td>70 ms</td>
<td>360 ms</td>
</tr>
<tr>
<td>Enter 6</td>
<td>Motor</td>
<td>70 ms</td>
<td>360 ms</td>
</tr>
<tr>
<td>Choose conversion</td>
<td>Cognitive</td>
<td>25 ms</td>
<td>170 ms</td>
</tr>
<tr>
<td>Conversion</td>
<td>Motor</td>
<td>70 ms</td>
<td>360 ms</td>
</tr>
<tr>
<td>Read value</td>
<td>Perceptive</td>
<td>50 ms</td>
<td>200 ms</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>450 ms</td>
<td>2346 ms</td>
</tr>
</tbody>
</table>
TEAM – Task models

Which currency converter use?

Change rate

Convert a value

Give change back

TEAM – Factors

- [Farenc & Palanque, 1999]
- Definition [McCall, 1977]
  - Quality factors
    - Requirements expressed by the clients and / or users
  - Quality criteria
    - Characteristics of the product (technical point of view)
  - Metrics
    - Allow the actual valuation of a criterion
  - Weighting
    - 5 possible weightings

TEAM – Example

How to select a flight?

- Click on the flight
- Flying over the flight
- Select the flight from the list

Performance (1)

Error Management (3)

Cognitive load (3)

Presentation outline

- Introduction
- QOC notation
- Mefisto method
- TEAM notation
- DREAM tool
- ATC case study
- Conclusion & perspectives

DREAM

- Design Rationale Environment for Argumentation and Modelling
- Models edition
  - QOC notation
  - TEAM notation
- Reuse of models
- Three visualisations of the model
- Manages duplication of elements
- Detects inconsistency

DREAM – Example
DREAM

- Extends expressive power of TEAM
- Supports hard-to-manage-by-hand tasks
  - Additional information for elements (more than a label)
    - Textual descriptions
    - Attach web pages
    - Attach electronic files
- Session
  - Authors
  - Model
  - Date

DREAM overview

Task Models

- id
- label
- information
- attached file

Question

- id
- label
- information
- attached files

Scenario

- id
- label
- information
- weight

Criterion

- id
- label
- information
- attached file

Facteur

- id
- label
- information
- attached file

Argument

- id
- label
- information
- attached file

Option

- id
- label
- information
- selected

Propose

- evaluation

Connected

- Arg_1
evaluation
- Arg_2
evaluation
- Arg_3
evaluation
- Arg_4
evaluation
- Arg_5
evaluation
- Arg_6
evaluation
- Arg_7
evaluation

Propose_2

- evaluation

Sub question from question

- Part of (1)
- Part of (2)
- Part of (3)
- Part of (4)
- Part of (5)
- Part of (6)
- Part of (7)
- Part of (8)

Part of (2)

- Part of (3)
- Part of (4)
- Part of (5)
- Part of (6)
- Part of (7)
- Part of (8)

Presentation outline

- Introduction
- QOC notation
- Mefisto method
- TEAM notation
- DREAM tool
  - ATC case study
- Conclusion & perspectives

ATC Case Study 1/3

- Task model of AT Controller

User Interface
Case studies results

- TEAM able to model ALL relevant information
- Graphical representation supports analysis

Conclusion

- Definition of a notation (TEAM)
  - Based on a well known and used notation (i.e. QOC)
  - Improved by usability work on QOC
  - Takes into account HCI approaches
- Development of a tool (DREAM)
  - Available at http://lihs.irit.fr/dream/
  - Taught in Toulouse Master in HCI

Interactive Systems Modelling
Generic Modelling Framework

The parts we have tackled…

- Relating system model & task model to ensure coherence using scenarios with CTTe & Petshop
- Incident & accident investigation techniques to inform system modelling using safety cases & ICOs
- Task modelling taking into account human error using HERTs & task patterns

CTT-ICO cooperation
- Task model edited in CTTe
- Scenario built using CTTe
- "execution" of the scenarios on the ICO models using Petshop

An Roadmap for Interactive Systems Modelling
### Rodmap for Interactive Systems Modelling

- **Target Applications, Domains**
  - Leisure
  - Healthcare and Medical Systems
  - Web Applications
  - Business Applications
  - Automated autonomous Real-Time Systems (VAL, TCAS)

- **Notations and tools**
  - UML, E/R, ...
  - B (Atelier B), Z, ...

- **User Interface Interaction Technique**
  - Direct Manipulation
  - Augmented Reality

- **Technique**
  - Command and Control Systems
  - Multimodal Interaction

- **Technique**
  - Full concurrency
  - Dynamic instantiation

- **Technique**
  - Tool support
  - Advanced Analysis techniques

- **Technique**
  - Hardware/Software
  - Infinite number of states

- **Technique**
  - Embedded UI
  - Tangible User Interface

- **Technique**
  - Multimodal Interaction for Interactive Safety-critical and Error-tolerant Systems

- **Technique**
  - Direct Manipulation
  - WIMP
  - No Interaction Technique

- **No more programming ?
No more bugs ?
Automated autonomous Real-Time Systems (VAL, TCAS)
No Interaction Technique
WIMP
Direct Manipulation
Augmented Reality
Command and Control Systems
Multimodal Interaction for Interactive Safety-critical and Error-tolerant Systems
Direct Manipulation
WIMP
No Interaction Technique

### Application to Industry Funded Projects

- **CNES Research and Technology action**
  - Starting date: October 2003 - Duration: 2 years
  - Theme: Multimodal Interfaces and 3D visualisation for Satellite Management and Control
  - Application domain: Satellite control and management

- **DGA PEA (long term research project)**
  - Starting date: January 2003 - Duration: 3 years
  - Theme: Multimodal Interaction Embedding Innovative Technologies for RAPALE aircraft
  - Application domain: Military Aircraft Cockpits and Air Traffic Management

- **DPAC Research and Technology action**
  - Starting date: January 2004 – Duration: 2 years
  - Theme: Formal Specification, Verification and Certification of Interactive Cockpits

### Research Cooperation Projects

- **Research Training Network ADVISES**
  - Starting date: October 2002 - Duration: 4 years – Funding: EU
  - Theme: Analysis Design and Validation of Interactive Safety-critical and Error-tolerant Systems

- **COST (Co-operation in Scientific and Technical Research) action MAUSE**
  - Starting date: January 2005 – Duration: 4 years – Funding: European Science Foundation (ESF)
  - Theme: COST 294: Towards the Maturation of IT Usability Evaluation

- **Network of Excellence ResIST**
  - Starting date: January 2006 – Duration: 3 years – Funding: EU
  - Theme: Resilience for Information Science and Technology

### IFIP WG 13.5 Activities

- **http://liihs.irit.fr/IFIP_WG13.5/**
  - One meeting per year (related to a conference)
  - Recent activities of WG 13.5
    - Special Interest Group session at CHI 2004 on Safety Critical Interaction
      - Includes slides, documents, list of participants, ...
    - HESSD 2005 conference within the IFIP WCC
  - Close relationship with Research Training Network ADVISES funded by the EU
    - http://www.dcs.gla.ac.uk/advises/
    - Analysis Design and Validation of Interactive Safety-critical and Error-tolerant Systems
    - Open/Free Tutorials offered by the network from each node of the network on a regular basis
    - Various material including tutorial notes available
    - Some post-doc and pre-doc positions available in the nodes

### IFIP WG 13.5 Activities

- **http://liihs.irit.fr/IFIP_WG13.5/**
  - One meeting per year (related to a conference)
  - Recent activities of WG 13.5
    - Special Interest Group session at CHI 2004 on Safety Critical Interaction
      - Includes slides, documents, list of participants, ...
    - HESSD 2005 conference within the IFIP WCC
  - Close relationship with Research Training Network ADVISES funded by the EU
    - http://www.dcs.gla.ac.uk/advises/
    - Analysis Design and Validation of Interactive Safety-critical and Error-tolerant Systems
    - Open/Free Tutorials offered by the network from each node of the network on a regular basis
    - Various material including tutorial notes available
    - Some post-doc and pre-doc positions available in the nodes