Formal Description of Multi-Touch Interactions

Arnaud Hamon¹,², Philippe Palanque ²,
¹ AIRBUS Operations, ² ICS-IRIT, University of Toulouse
Context of this work

- The world is changing (in terms of interaction with computing device)
  - In our daily life
  - In the aeronautics area
  - In all the other areas
  - Step forward ? Step backward ? – FAA certification of iPAD for EFB
The Last 10 Years - J. Gray (Dep. In the internet area - High Dep Comp. Conf 2005)

- Things got better, then things got a lot worse!
Context of this work - the times they were a-changin’
Context description - rationale for changes (already from ARINC 661 - 2000)

- More systems to manage,
- More complex systems,
  - Need for more command and control functionalities
  - Need for more monitoring capabilities
  - Need integration (concentration) of information

- Available space limited cockpits
  - Manage the command and control surfaces
- Evolvability of the systems
  - Need for more upgradable equipment's

Increase Pilots-CDS interactions bandwidth
Evolution of the HCI concepts
Bringing Tactile Interactions in the Cockpit

• Rationale
  ▫ Bandwidth improvement between pilots and systems
  ▫ Wider integration of input and output
  ▫ Space management (input integrated in output)

• Reduction of constraints
  ▫ Technology getting mature (is it enough for cockpit integration?) for studies
  ▫ Widespread use and basic knowledge in the user population – reduced training / differential training
Current Situation

- Low hanging fruits already been collected
- Foundations identified many years ago
- Refinement and deeper understanding over the years
Requitements for notations - State of the art UIDL

**Dynamic instantiation**
- Fingers are detected as users touch the screen
- \(\text{\textgreater} \) Need to instantiate and remove their behavior, link all possible associated events dynamically
- Their value need to be connected to the models evolution

**Fingers’ clustering**
- Being able to group, ungroup and regroup fingers that are belonging to the same interaction technique

**Multimodal interactions**
- Full concurrency
- Quantitative time

**Reconfigurability (in case of failures)**

**Scalability**

**Verify-ability**
<table>
<thead>
<tr>
<th>Data Description</th>
<th>State Representation</th>
<th>Event Representation</th>
<th>Time</th>
<th>Multimodality: fusion of several modalities</th>
<th>Concurrent Behavior</th>
<th>Analysis</th>
<th>Dynamic finger clustering</th>
<th>Capability to deal with multi-touch interactions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Qualitative between two consecutive model elements</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Implicit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Quantitative between two consecutive model elements</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Explicit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Quantitative over non consecutive elements</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Concurrent Behavior</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Output: One duo to rule them all, and in the design bind them...
The proposed approach

- A formal notation
  - State-based and event-based
  - Concurrent
  - Able to handling dynamic instantiation
- A dedicated tool for the notation
  - Models editing
  - Models simulations
  - Models verifications
- An architecture
  - To structure models
  - To go from hardware to software
The choice of notation and its associated framework

- ICO

Token (interactive object)

And services available:

which corresponds

Timed transition

- PetShop environment edition, simulation, performance evaluation, testing and verification
The PetShop Environment
Multi-touch interaction system architecture
The need for an models’ architecture

- Build on proven software architecture software that match our targeted properties
- To address the different event abstraction levels and a clean separation between dialog and presentation: ARCH

The low level transducer receives the raw data events, and forwards them then with the proper structure.
The low level transducer receives the raw data events, and forwards them with the proper structure.
Relationship with Hardware platform

- Low level transducer

The low level transducer receives the raw data events, and forwards then with the proper structure.

Dynamically registration to the event produced by the LowLevelTouchProvider.
Relationship with Hardware platform

- Elementary interaction techniques

Theses models specify elementary multi-touch interactions:
- Tap
- Tap and hold,
- ... and zoom, pinch ...

The models are independent and trigger the corresponding events which are listened by the Touch interaction manager.
Formal Modeling of Interaction Techniques - Tap
Formal Modeling of Interaction Techniques - Tap and hold
Formal Modeling of Interaction Techniques - Tap and hold
The presented model is currently in draft. It creates elementary interaction models and acts as a fusion/fission multimodal engine.

- **Touch interaction manager**

Elementary model creation

Link with low level transducer
Conclusion

- A formal description technique for the description of touch-based interaction techniques
- Reasoning about models and properties (e.g. conservative and repetitive components)
- Software tool to support
  - description and execution of these interaction techniques
  - Tuning of interaction techniques
  - Integration within interactive applications
- A notation within a framework and within a process to model multi-touch interactions
Future Work

• Close work with people in charge of the definition of touch interactions suitable with the context of use (vibration, light variation, stress level, ...)
• Integrate precisely with rendering aspects (animations, ...)
• **Address more precisely gesture aspects** (directions, speed, shapes, ...)
• Experimental results and user studies
• Connection with work on fault-tolerance and dependability
Thank you very much for your attention