Multi-Touch Interactions for Control and Display in Interactive Cockpits: Issues and a Proposal

Arnaud Hamon[1,2], Phil Palanque[1], Raphaël André[2], Eric Barboni[1], Martin Cronel[1], David Navarre[1]

[1]ICS-IRIT, University Toulouse 3,
118, route de Narbonne,
31062 Toulouse Cedex 9, France
{lastname}@irit.fr

[2]AIRBUS Operations
316 route de Bayonne
31060 Toulouse cedex 9
{Firstname.Lastname}@airbus.com

ABSTRACT

Over the last few years multi-touch interfaces have made their ways in most environments including mobile technologies, flight entertainment systems, consumer electronics, … Such interfaces and associated interaction techniques have demonstrated benefits partly due to the fact that the output device integrates input management thus bridging the (classical) gap between input and output in user interfaces. They have also demonstrated benefits in terms of performance for triggering commands by exploiting multi-fingers interactions thus reducing the number of un-necessary modes. Together with these benefits, multi-touch interfaces bring a set of issues that are still to be solved prior to make them "acceptable" for command and control of (safety) critical interactive systems. Such issues include usability and reliability in a way that is even more salient than in "classical" WIMP (Windows, Icons, Menus, Pointing device) interfaces that have been around for more than 30 years. This paper proposes a notation and its associated tool for describing in a complete and unambiguous way multi-touch interactions thus mainly targeting at reliability. We present from a case study (in the application domain of interactive cockpits of aircrafts) how the notation makes it possible to describe such interactions from the hardware level (input devices) to the user application level.

Keywords

Tactile interactions, model-based approaches, interactive cockpits, multi-modal, multi-touch

INTRODUCTION

Over the years, the evolution of cockpits in large civil aircrafts has taken place following two different paths:

- Small increments/evolutions targeting at solving identified problems or integrating new equipment into an existing cockpit,
- Significant steps/evolutions ending up with a complete re-design of the cockpits including control and displays. Examples of such evolutions include glass cockpit (where large display units were included in the flight deck) and more recently interactive cockpits compliant with ARINC 661 specifications [1] where interaction takes place through mouse-like input devices and keyboards.

In parallel of these evolutions in the cockpit, multi-touch interfaces have made their ways in most environments including mobile technologies, flight entertainment systems, consumer electronics, … Such interfaces have demonstrated benefits related to the fact that the output device integrates input management thus bridging the (usual) gap between input and output in user interfaces. They have also demonstrated benefits in terms of performance for triggering commands by exploiting multi-fingers interactions thus reducing the number of un-necessary modes.

Providing multi-touch interactions for the command and control of civil aircraft would consists in another significant evolutionary step as far as the evolution of cockpits is concerned. However, together with the benefits presented above, multi-touch interfaces bring a set of issues that are still to be solved prior to making them "acceptable" for command and control of (safety) critical interactive systems. These issues impact many properties such as reliability, and more globally resilience.

These set of issues are not new and already appeared when interactions in the cockpit evolved from "physical interactions" (by manipulating physical knobs and reading information on dials) to software controls mainly based on the ARINC 661 specification [1]. Often considered more reliable means to trigger commands from the cockpit equipment's, pushbuttons, rotators and safe-guarded physical buttons have populated the flight decks' space. However these devices generate significant weight load and bring modifiability (upgradability) issues. Indeed, these physical components are directly linked to the systems they command resulting in the fact that evolutions made to the aforementioned systems are likely to require modifications to the components. This requires adaptation of the cockpit itself (which is a distributed cognitive system [20]) and ultimately the adaptation of procedures and training. Finally, due to their physical nature, the more complex the aircraft systems, the more numerous physical components will appear in the cockpits. The development of the ARINC 661 specification [1], described in the related work paragraph, was mainly driven by these issues i.e. reducing cockpit size and increasing its modifiability.

While our previous work presented in [18] mostly addressed the design process for engineering multi-touch interactions in cockpits, this paper focuses on the engineering aspects of multi-touch interactions and the engineering on applications exploiting multi-touch interfaces. Therefore, the remainder of this paper is organized as follows: First it describes the diversity of multi-modal interactions by structuring the design space in multiple dimensions. Second it addresses the rationale for the use of multi-touch applications in avionic applications as well as the related issues from a specification perspective. It presents why multi-touch interactions have the potential for going beyond the numerous benefits brought by interactive cockpits compliant with ARINC 661 specification but also the new problems they raise in terms of engineering. The paper then proposes a contribution targeting at these issues and demonstrates on a case study how the approach can be applied and how it solves the identified issues. Finally, the related work section positions the contribution presented in this paper with other work in that domain.

MULTI-TOUCH DESIGN AND ENGINEERING SPACE

This section describes the multi-touch interaction design space from a design perspective via a tree-shape taxonomy (Figure 1). This taxonomy divides touch interactions into groups featuring the same number of fingers. The group composed of mono-finger interactions is divided into branch with similar properties (already defined in [19] such as time and spatial properties). For multi-touch interaction techniques, the taxonomy regroups the interactions with respect to their parallel or sequential behavior. These branches are refined to create groups where each finger involved in the interaction belongs to the same/different branch of the 1_Finger group of the taxonomy. For example, a 2 fingers' split is composed of 2 fingers (each of them being of the kind of 1_Finger_With movement_ without speed_with direction branch of the taxonomy). In addition, these 2 fingers belong to the 2_Fingers branch and as they move in opposite directions, their key properties have different values. This is why 2F_Split is the leave of the branch

2_Fingers_Parallel_SameFamilly_DifferentProperty.

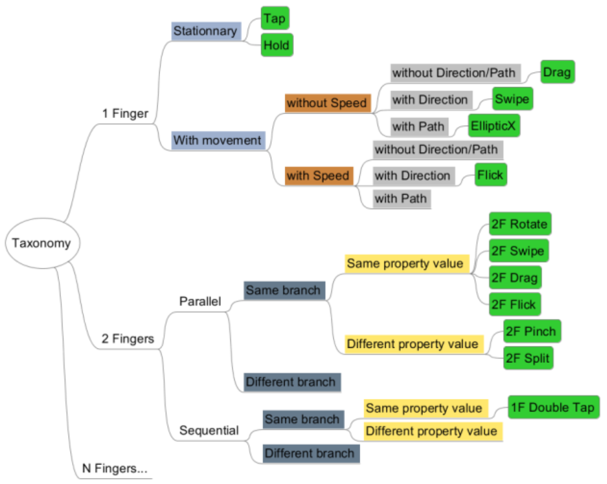

Figure 1- A taxonomy of multi-touch interactions from a design point of view

A multi-touch interaction design space

Due to space constraints, this section only provides an overview of the design space (from a design perspective) of multi-touch interactions via tree-shape taxonomy (Figure 1). This taxonomy divides touch interactions into groups featuring similar properties (already identified in [19]) such as time/spatial properties, parallel/sequential behavior...

The leaves of the taxonomy represent touch interaction techniques while the branches represent its characteristics thus clustering interaction techniques according to their intrinsic properties. In classical touch interfaces, multi-modality is only possible when several fingers are involved in the interaction. However, in a cockpit, multi-modal interactions can occur through single finger interaction performed by multiple users. These multi-modal and multi-user aspects are captured by the dimensions of Table 1 which complements Figure 1. For instance, the 2 fingers' swipe ("2F Swipe" leaf of the tree of Figure 1) is a synergistic interaction which corresponds to "g" cell in Table 1. Touch interaction techniques proposed in [17] can be positioned with respect to this taxonomy. For instance, an interaction technique involving one finger which is required to describe an elliptic movement on the screen without any constraint on speed, would be positioned at the end of the branches "1 Finger" With movement" "without speed" (currently represented as "Elliptic X" leaf).

This taxonomy is very useful for assessing the coverage of the engineering contributions presented later on in the paper, but design issues remain out of the scope of this paper. The contribution to engineering presented in [19] covers parts of these dimensions but the remaining dimensions are detailed in the following paragraph.

Engineering issues of multi-modal interaction

Engineering issues of multi-modal input interactions for a single user have been studied and classified in [10] and a taxonomy based on this classification has been proposed in [27]. Table 1 proposes a classification of fusion engines incorporating multiple modalities and multi-user interactions. Some interaction techniques produce in one single input multiple information (e.g. the vocal command "destroy the blue circle"). In such cases, this input has to be split into several ones before a command can be triggered (e.g. command "destroy" with parameter "blue circle"). This process is called input fission and is represented in the last rows of Table 1.

		Use of Input modalities[1]			
		Sequential		**Parallel**	
		Single-User	Multi-User	Single-User	Multi-User
Fusion	Independent	a	b	c	d
	Combined	e	f	g	h
Fission	Independent	i	j	k	l
	Combined	m			

[1] In the case of multiple users using the same mode of interaction on different devices, fusion engines can handle that as if it was multiple modalities used by a single user

Table 1- Classification of fusion and fission engines for input modalities

Beyond the input aspect of multi-modal interactions, multi-modality takes place at output level too. For instance, the cockpit is a place of convergence for the multitude of information produced by the many aircraft systems. In the early days of aviation, each piece of information was displayed on a specific display device (usually a dial) in order to be used by the flight crew to carry out operations. To avoid multiplication of display and to support crew member (which has in such cases to perceive multiple information to make sense out of them) several information can be blended into one single display which requires the definition and engineering of fusion output engines. Table 2 proposes a classification of fusion and fission engines for output modalities.

		Use of Output modalities	
		Sequential	**Parallel**
Fusion	Independent	n	o
	Combined	p: non relevant	q
Fission	Independent	r	s
	Combined	t: non relevant	

Table 2 - Classification of fusion and fission engines for output modalities

The following paragraphs provide examples of the elements appearing in Table 1 and Table 2.

- a and b correspond to exclusive user actions. In such cases, modalities cannot be fused and have to be used in sequence. This corresponds to standard interactions where no multimodality occurs.
- c and d correspond to concurrent user actions. Modalities are not fused but can be used for two different actions at the same moment, which is typically the multi-user scheme in a cockpit.
- e and f correspond to the cases where modalities can be fused but have to be used in sequence e.g. specific actions and modalities assigned to users.
- g and h correspond to synergistic actions. In such cases modalities can be used freely, it's possible to use them in parallel and to fuse them by one single user or by several users simultaneously.
- i corresponds to the speech command described above and requires a fission engine to make sense of the command.
- j corresponds to the multi user aspect of item “i”. Use of modalities is done in a turn by turn manner.
- k corresponds to the use of several modalities at the same time by a single user and requires a fission engine to decompose each input.
- l is similar to item “k” but authorizes several user at the same time for different actions.
- m corresponds to the case when a fusion has occurred (sometimes at the hardware level) which needs to be undone. An example could be a tactile surface with pressure which create a single input combining position and pressure).

For the outputs, we present a classification system-oriented where information from the system is the input of engine and the users' task which manipulate this information are not considered

- n corresponds to the case where information is not fused and must be presented sequentially to the user. Such sequential information can be fused in the head of the user.
- o corresponds to the case where information is not fused but can be presented in parallel (e.g. a voice message and a bip). It is left to the user to fuse or not this information.
- q corresponds to synergistic fusion of information using parallel output modalities. For instance the display of the value 100 in green (meaning that it has validated by the system).
- r corresponds to the splitting of an information on several modalities in a sequential way.
- s corresponds to the same splitting of information as in item “r” but produced in parallel. For instance the simultaneous production of an animation and of an alert sound.

RATIONALE FOR INVESTIGATING MULTI-TOUCH INTERACTIONS IN COCKPITS

The evolutions brought by user interfaces compliant with ARINC 661 specification [1] have not been able to solve all the issues raised by interactions in the cockpit:

- Despite the potential use of DUs (Display Units i.e. computer screens) combined with KCCUs (Keyboard and Cursor Control Units), the cockpit space remains confined and a bottleneck for the ever growing complexity of aircrafts;
- Input and output are still disjoints (e.g. KCCUs and DUs) which introduces articulatory distance for pilots to control the avionic systems;
- Even though some cockpits offer touch interactions they remain limited to mono-touch techniques over WIMP interfaces.

Since the observations made in [18] several studies and research projects have studied the potential benefits of multi-touch interactions in cockpits. For instance, the FAA has appointed Honeywell to study the human factors aspects of pilots' activity supported by touchscreens [31]. From a manufacturer point, the Airbus R&T cockpit division [4] studied the introduction of multi-touch displays in cockpits (cell “g” in Table 1), following a human centered approach where a classification of multi-touch tasks according to their ergonomic criteria is proposed.

That study demonstrates the benefits of using multi-touch interfaces in cockpits and the large impact the context of use has on the interaction.

Multi-user interactions extend the interaction space compared to mono-user ones. Having two users implies, for instance, that there are now four hands which can interact with the system at the same time and/or location (cell “h” in Table 1). Following this policy it is possible to design specific interactions for a critical operation which would require actions to be performed by two operators simultaneously. For instance, a double check action with four hands in a delimited range of time [11] where both members of the crew have to touch the screen with their thumbs. The interaction techniques could also be different for the two users, e.g. the captain has to touch with both thumbs while the first officer has to make two circles at the same moment to avoid spurious and human error.

This section has presented the potential benefits for introducing multi-touch interfaces in the cockpit. Next section lists the issues raised by such an introduction in terms of engineering.

ENGINEERING ISSUES RAISED BY MULTI-TOUCH INTERACTIONS

This section introduces the main issues for adding multi-touch interactions in cockpits from an engineering point of view. Since multi-touch interactions are multi-modal, next paragraph addresses the generic issues for engineering multi-modal interactions while the second one details specific multi-touch issues. Last paragraph addresses the issues related to engineering such interaction techniques in the specific context of interactive cockpits.

Multi-modal interactions

As the engineering of multimodal interfaces has been studied for many years, we focus here on issues related to having more than one user involved in the interaction. We have identified two main issues related to multi-users interactions. First, for the same input device and the same input action, two users can have different interaction techniques e.g. swipe versus simple press on a touch-screen. Second, the devices used by the various users may be of different types, not co-located and installed on different computers running different operating systems and window managements. All these issues require the careful definition of multi-modal multi-user interactions making explicit the user, the input device and the interaction technique used.

Currently, our work focuses on identical interaction techniques per type of device, each person of the crew having his/her own touch-screen and a shared one (as this proposes a good combination of the issues above).

Multi-touch interactions

Multi-touch interaction techniques are, by definition, multi-modals, and therefore inherit the issues we presented in the previous paragraphs. However, they present additional specific issues and the following paragraph describes them with an emphasis on multi-touch interaction specification: interactivity continuity, dynamic instantiation, complex time relationships and finger clustering. Most of them were mentioned in [19] but interaction continuity and anthropomorphic characteristics have been recently identified.

Interaction continuity (issue 1)

One major evolution from the use of ARINC 661 WIMP interactions consists in describing continuous interactions (as identified in direct manipulation [37] or post-WIMP interactions [41]) the users’ interactions with the system are no longer discrete (one event triggers one command) and become continuous (a flow of event is produced before triggering a command) and continuous feedback has to be produced. Consequently, the notations to describe these techniques shall support the description of event streams as well as the description of the related immediate feedbacks. Commands such as pinch to zoom belong to this category.

Dynamic instantiation of input devices (issue 2)

When all input devices are not known when the user interface is initialized, the input devices detected during the interaction need to be dynamically instantiated in order to be registered and listened to (i.e. their events are possibly used for triggering commands). This is typically the case while addressing multi-touch interaction techniques since users’ fingers are only detected as they touch (or approach) the tactile surfaces. This is very different from WIMP interfaces where the set of mice and keyboard is defined in a static way.

Temporal representations and their fine tuning (issues 3, 4)

As illustrated in [19], specifying multi-touch interaction techniques requires both qualitative and quantitative time descriptions. Such timing aspects (which have to be finely tuned) are required to be explicitly and thoroughly defined for describing multi-touch and multi-modal interactions as presented in [36]. For example, to specify a double tap, designers need a fine tuning on two nonconsecutive events: first finger’s touch and last finger’s release.

Finger clustering and anthropomorphic characteristics (issues 5, 6)

In order to fully incorporate the expressive power of multi-touch interactions there is a need to be able to detect (at interaction time) group of fingers evolving jointly. This might be the case for fingers belonging to the same hand (often called finger clustering) but this usually requires specific algorithms as proposed in [8] that must be incorporable in the user interface description language. More fine grain interactions might be addressed such as analyzing the shape of contact of the fingertip (to determine orientation of the finger on the touch device as proposed in [12]. Lastly, it is important to be able to describe interaction techniques taking into account the anthropomorphic characteristics of the hand [29]. For instance, an interaction featuring two fingers from the same hand will not have the same spatial and behavioral properties as an interaction with fingers belonging different hands.

Multi-touch interactions in cockpits

The avionic software environment in cockpits of civil aircrafts requires applying dedicated design processes to ensure the reliability, safety and usability of these critical embedded applications. Therefore, specifying multi-touch interaction techniques for cockpits in a complete and unambiguous way is a mandatory requirement.

Interaction accuracy versus usability (issue 7)

Due to the un-stability of the environment, it is important to be able to avoid spurious interactions (i.e. a pilot triggering inadvertently a command through a non-desired touch sequence). This can be achieved by changing the design of touch interactions by using more pressure gestures from pilots which might have as a result a reduction of the usability. Other engineering solutions might involve context information detection and fusing this information with operators' actions.

Interaction reliability and dependability (issue 8)

Finally, as for other avionic applications there is a need to apply verification and validation (V&V) techniques to the entire cockpit display system (from the input devices to the interactive multi-touch applications). This requires the engineering techniques used for the interactive multi-touch cockpits to be amenable to formal verification.

RELATED WORK

Cockpits' interactivity

Current (or soon to be released) interactive cockpits essentially feature both physical and WIMP interactions for triggering avionic functions. Indeed, when critical commands are considered, dependable interactions using physical interactors are preferred to software ones. Multi-touch applications on EFBs (Electronic Flight Bags) are already available as demonstrated by the Airbus application on the appStore [15]. Finally mono-touch interactions have already appeared in light aircrafts' cockpits and fighters [40] (cell "a" in Table 1); but multi-touch applications for avionic application still remain at the level of industrial research projects such as [39] (cell "c","g" in Table 1).

Notations for engineering multi-touch interactions

Many research works have contributed to allow developers handling multi-modal interfaces design more easily including [28] and [32]. The domain specific language presented in [25], for instance, addresses some dimensions we described in [19] as dynamic instantiation of input devices but lack parallel behavior description and interaction analysis. We observed that most notations (identified in [19]) lack expressive power to encompass all the dimensions for multi-touch interactions and are not able to specify in a complete and non-ambiguous way multi-touch (and more generally multi-modal) interactions. Proton++ [23] does not explain how to support multi-users touch interactions with several fingers. In conclusion, except the ICO notation (that is further described in next section), currently available notations for engineering multi-modal interactions, are not able to address all the engineering issues presented above.

A CONTRIBUTION TO THE ENGINEERING OF MULTI-TOUCH INTERACTIONS

This section demonstrates how the approach proposed in this paper addresses all the issues presented above. Our contribution is composed of two main elements: a software architecture enabling the use of multi-modal interactions in the context of interactive cockpits and a notation capable of describing in a complete an unambiguous way such interactions. This notation is supported by a tool suite which is presented in the last paragraph of this section.

A layered architecture

This section proposes a layered architecture (presented Figure 2) that ensures the availability of "good" properties of software systems mentioned in [25] (flexibility, separation of concerns, extensibility and hardware independence) and supports the handling of interactions with multiple fingers which corresponds to the dynamic instantiation of input devices (issue 2).

Figure 2 describes an example of this architecture where the system is composed of two multi-touch sub-systems, one for each pilot. The physical interaction layers are designed to provide the system access to the users' events: fingers' touch on the display. These events are sent to hardware independent finger models which are registered to the interaction models. The fusion engine ensures the creation and forwarding of coherent interactions' events towards the applications (used by the pilots) as well as the communication between the two sub-systems. To determine the coherence of the interactive events send to the applications, the fusion engine needs to know the characteristics of the widget on which the interaction takes place.

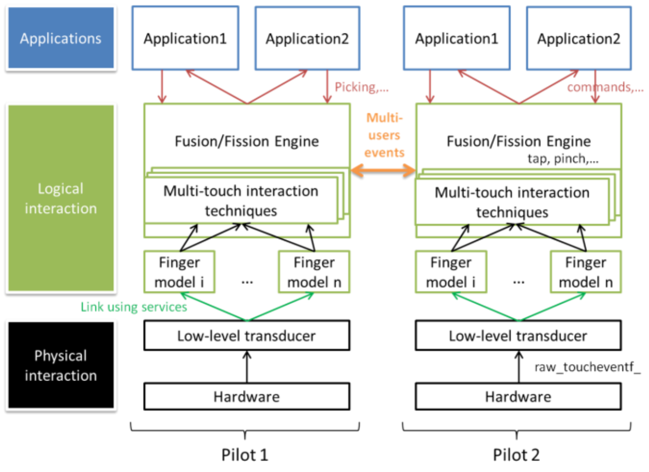

Figure 2 - Proposed layered architecture for multi-modal interaction support in input

This architecture is based on the work previously published in [19] which was an extension of another layered approach proposed in [13] but only dealing with multimodal interactions. We improved that architecture by introducing an intermediate level between the low-level transducer and the interaction models: a model to describe each finger behavior. These dynamically instantiated models are independent from the lower layers and therefore ensure a higher flexibility and independence from the upper layers. In addition, we enriched the fusion engine with the capability to communicate with other systems. This communication capability via the Ivy bus [9] will be described more precisely in the case study section. The complete description of the other layers properties and communications have been introduced in [19] and are not repeated here due to space constraints.

A notation for engineering multi-touch interactions

ICO: Informal definition

The ICO notation (Interactive Cooperative Objects) is a formal description technique devoted to specify interactive systems. Using high-level Petri nets [16] for dynamic behavior description, the notation also relies on object-oriented approach (dynamic instantiation, classification, encapsulation, inheritance and client/server relationships) to describe the structural or static aspects of systems.

ICO notation objects are composed of four components: a cooperative object for the behavior description, a presentation part (i.e. Graphical Interface), and two functions (activation and rendering) describing the links between the cooperative object and the presentation part.

ICO addresses all the issues presented in the previous sections including dynamic instantiation of input devices which is not addressed elsewhere in a modelling technique. ICO uses the capabilities of Petri nets to describe tokens' creations which are used in models to represent input devices such as fingers on a touchscreen.

Lastly, it is important to note that ICOs have been used for various types of multi-modal interfaces [26] and in particular for multi-touch [19]. This notation is also currently applied to formally specify interactive systems in the fields of Air Traffic Management [30], satellite ground systems [34] and cockpits of military [6] and civil [5] aircrafts.

Generalization of ICO event handling capabilities

The models we present in this paper feature an extension of ICO that was required to address the issues of multi-touch interactions. In order to do so we propose a new transition allowing one cooperative object to receive events. In models describing multi-touch interaction techniques, tokens may represent pilots' fingers "currently" on the device. These transitions can be used to specifically retrieve the input that emitted the event and to specify the event listened to as well as its source. It also exposes the properties of the event and allows specifying a condition on the event received in order to fire the transition. As this extension supersedes the previous event handler transitions in ICOs, the former transitions are not used anymore.

Figure 3 represents[2] how to use this new mechanism: the model's extract presented is composed of a place **OperationalInputDevices** that contains the input devices of the system and their activation status. Such transition would belong to the Low-level transducer layer of Figure 2. The disableInputDevices transition listens to "*error*" events emitted by the input devices. When the place **OperationalInputDevices** contains at least one active device, disableInputDevices is enabled. If so, when it receives an "error" event, disableInputDevices is fired if the criticality field of the "*error*" event is high (this is defined by the eventcondition level.isHigh()). When disableInputDevices is fired, the action (in Figure 3 "device.idle()") is executed (setting the device in its idle mode) and the source device is moved to place **FailedInputDevices**.

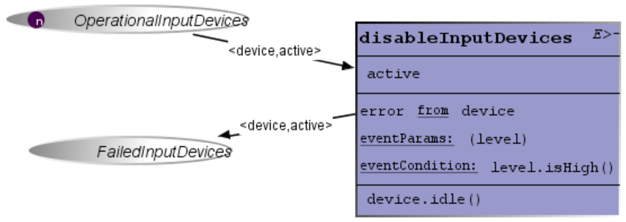

Figure 3- new ICO event transition's example

This extension to ICO formalism enhances multi-modal interactions specification by offering the possibility to explicitly describe the events properties and their relations to dynamically instantiated input devices.

A tool to support the notation

In order to edit and execute ICO models, the notation is supported by the CASE tool PetShop [35,18]. PetShop framework also provides tools to support user evaluations [33] and formal analysis of models [38] thus addressing issue 8.

In addition PetShop describes and simulates OS independent multi-touch interfaces (by connecting dedicated APIs to its kernel), which is not the case of most of the frameworks presented in [21].

CASE STUDY

The case study presented in this section focuses on demonstrating the expressive power of the ICO notation for the engineering of multi-modal and multi-touch applications. This case study corresponds to a subset of a weather radar system from a civil aircraft cockpit which provides atmospheric data to the cockpit crew and is controlled by different systems:

- The input system is composed of a pushbutton on the glare shield panel: one on the CPT (Captain) FCU (Flight Control Unit) and one on the FO (First Officer) FCU and interactors on the pylon (duplicated for each pilot);
- The outputs of the weather radar are provided through the NDs (Navigation Displays);
- The back-up system, supported by the FCU-backup application is not part of the study. It provides pilots

[2] To enhance the readability of this paper, font attributes are modified for the places, transitions and "events" when ICO models are described.

additional access to the weather radar controls based on this ARINC 661 UA (User Application).

The scenarios we present are representative of the potential use of multi-touch (modalities, users' interactions…) for such an application. However, we do not intend to address design aspects or to demonstrate their operational validity as it is out of scope. Nevertheless, we intend to demonstrate that our notation provides a detailed and exhaustive mean to specify multi-modal critical systems and covers all the issues presented above.

Figure 4- A350 cockpit overview

For the purpose of this study, we consider a cockpit configuration in "T-shape" as of the Airbus A350 (Figure 4). For the case study we consider a set of three displays: one screen in front of each pilot and one on the pylon, between them. These displays support multi-touch inputs (at least 5 touch inputs recognizable at the same time). The other technical features of the hardware are not considered here (resolution, latency…).

To illustrate the main dimensions of multi-modality, multi-touch and multi-user interactions, we propose three pilot tasks touching different dimensions (see Table 3):

Tasks	Dimensions
Basic weather-radar manipulations: activate, change layer opacity	Single user, multi-touch
Cloud evasive route,	Multi-users, synergistic
Modification of a "unique" weather parameter by both pilots jointly	Multi-touch, multi-user, Exclusive

Table 3- Pilots' tasks and their dimensions

To support these three tasks we have setup a platform composed of three PCs running PetShop and linked using the Ivy Bus [9], one ND application on each pilot's computer and a validation application for both of them to interact on. Such architecture is compliant with the work done in [14] where each PetShop is running over an ARINC 653 operating system to address dependability issues (not addressed here).

Figure 5 presents the multi-touch interface displayed on each pilot's ND. The interface in composed of a multi-touch map and various custom widgets emulating physical buttons and interactors that command the display modes and the weather radar controls.

Task 1: Advanced multi-touch manipulation

For this task, a pilot (either the CPT or the FO) interacts with the weather radar through the multi-touch Navigation Display to (de)activate its visualization or modify its intensity (opacity of the layer displaying the weather information on top of the ND's map). In addition, the pilot is able to interact with the map itself: modifying the range (zoom) level and translate the map.

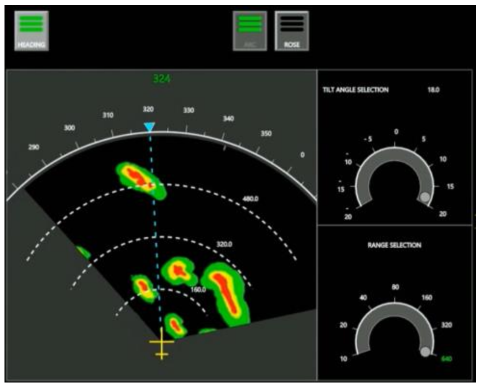

Figure 5 – Navigation Display's interface for this case study

These interactions have been designed and proposed in various research papers as well as public SDKs (Software Development kits) of multi-touch platforms. Table 4 summarizes theses interactions and their references. They cover issues 1, 2, 3 and 4 described above and are covered by models belonging to the logical interaction layer of Figure 2.

Interaction technique	Command description	Interaction properties targeted	Reference
1 finger drag	Change Heading	Direct Manipulation	Most SDKs
1 finger lasso	Select object on map	Selection	[25]
2 fingers' flick	Trigger weather layer visibility	Two fingers with same unitary interaction, velocity and acceleration	Most SDKs
Precise Tilt/Z-transition	Tune weather layer intensity	Timing aspects, multi-fingers (>2)	[24,17]
2 fingers' rotate	Map rotation	Direct Manipulation (continuous)	Most SDKs
2 fingers' scale	Map range	Direct Manipulation (continuous)	Most SDKs

Table 4 – A sample of multi-touch interactions for weather radar manipulation

We modelled each of Table 4's interaction techniques using the ICO notation and implemented them in our application according to the architecture presented Figure 2. In this paragraph, we emphasize ICO's expressive power by describing an excerpt of the precise tilt interaction technique (also referred as "*Z-Translation*" in [24]), and more precisely the behavior part regarding fingers movements on the touchscreen. [24] describes this interaction technique as follows: "*z translation – a conjoined touch on the object, together with a one-touch drag up and down*".

This designer-like description remains at a very high level and when developers have wish to implement such interaction techniques, several cases need further details as, for instance, the description of the impact of each finger movement on the screen.

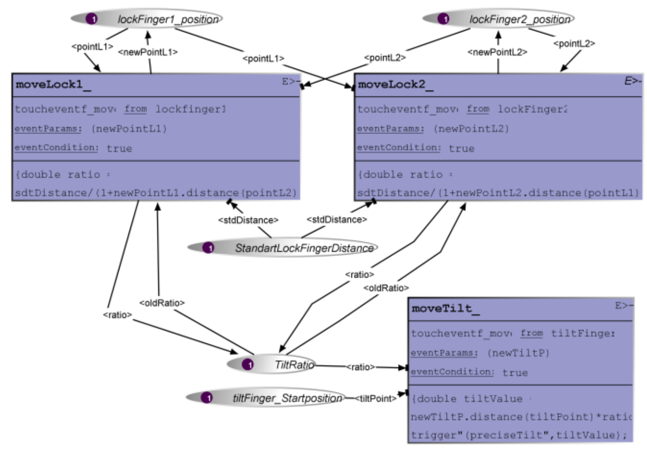

Figure 6- Excerpt of the model of the behavior of "tilt"

Figure 6 describes a simple case where the third finger (controlling the tilt value) is moving. When is does so, the transition moveTilt_ is fired and executes its code, triggering a "*preciseTilt"* event. The tilt value only takes into account the distance between the initial touch of the tilt finger and its current position. As a refinement, designers may need to modify this behavior by weighting this value with the distance between the two fingers serving as anchors. The model presented in Figure 6 illustrates how events' handling mechanisms in ICO can describe that feature with a rather simple and localized modification of the original model.

Task 2: Cooperative cloud-avoiding manoeuvers

This paragraph details a synergistic task between both pilots consisting in avoiding clouds during flight. While the captain is responsible for flying the aircraft from a short term perspective using the weather radar at close range, the first officer is studying the weather further away along the predetermined flight path to avoid potentially uncomfortable zones for the passengers. The first officer then proposes a new route via his ND. Both pilots must then validate the flight plan modification on the confirmation display, each one having to press his validation button at the same time.

Figure 7 describes the behavior of the validation application which is symmetrical for both pilots. When the FO proposed a new flight plan, the newFlightPlanFromFO transition is fired and a token containing a reference towards the FO's ND and the proposed flight plan is put in place **NeedOfValidation**. When the FO (resp. CPT) presses his validation button, the transition FOValidate (resp. CPTValidate) is fired and a new token is put in **FO_interacts** (resp. **CPT_interacts**). The CPT (resp. FO) has to validate within 2.5s otherwise the validation is not allowed. If the FO (resp. CPT) released the button before the CPT (resp. FO) validates, he has to start over the validation. If the proposal is not validated withing ten seconds, it is discarded, discardValidationRequest_ is fired and the FO's ND is notified accordingly.

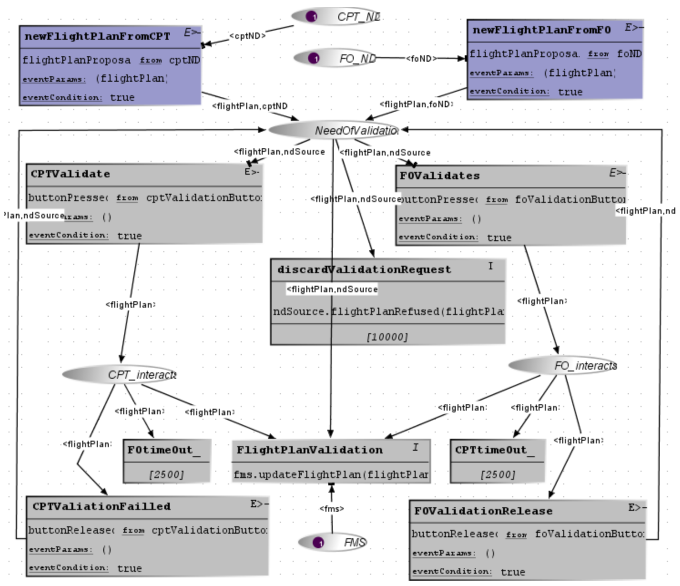

Figure 7 - Flight Plan cooperative validation model

Task 3: Concurrent edition of a singleton parameter

This third task illustrates the interaction conflict that may occur as both pilots modify simultaneously the same system parameter. For this task, both pilots try to modify the tilt angle of the weather radar at the same time.

Such conflicts are likely to appear as more and more system parameters are controlled by User Applications which are easily replicated on multiple displays. In the latest Airbus cockpits, when a pilot is moving his KCCU's cursor over a system page, conflicts are avoided by denying the other pilot's cursor access to that DU area. This approach is fundamentally opposed to the one proposed in [22] where conflicts are likely to be solved by human rather than locking mechanisms. Therefore they propose an optimistic algorithm for conflict resolution favoring interactions as much as possible and providing undo mechanisms when a seldom conflict occurs.

Our approach preserve the main principle supported by [22]: "*User commands must be handled immediately without waiting for either authorizations or acknowledgments*".

Let us consider the interactor commanding the tilt angle of the weather radar on the captain side. Because denying both pilots interacting over the same system page is very restrictive, one might want to only prevent them from interacting on the same widget. Figure 8 presents an excerpt of the behavioral model of such an interface. This model is identical for both pilots. Since all DUs and their UAs are connected, the place **Peer** contains a reference of the FO's ND applications. When a pilot starts interacting on the widget controlling the weather radar's tilt angle, this widget w (of which a reference is stored in place

TiltWidget) triggers an "*interactionStarting"* event. If the tilt is not locked for this pilot, the event is received via the canReserve transition. This transition locks the tilt on the other pilot's application using the service lockTilt() preventing the other pilot to interact with his own tilt widget. At the end of the interaction, the model updates the tilt on the other application and allows the other pilot to interact with it.

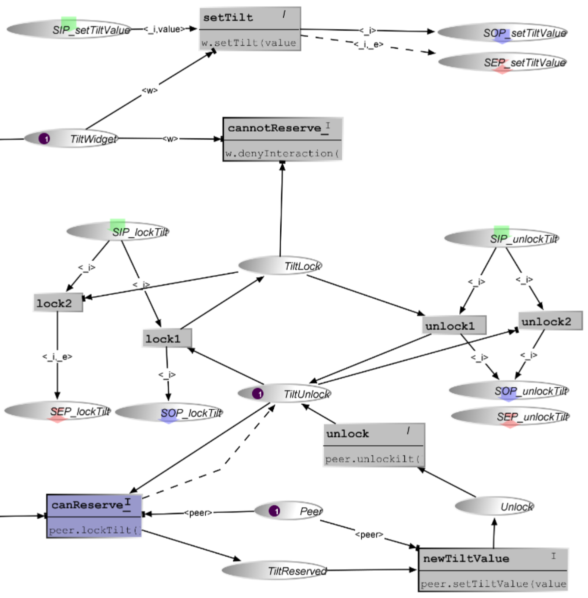

Figure 8 - Excerpt of the ND application model

The dashed arcs represent exception arcs formally described in [7]. Such an exception can be thrown by the FO's ND application if the FO is interacting with the tilt widget (this corresponds to cell "g" in Table 1). The Captain model represents how this exception can be caught making it possible to specify what behavior follows its occurrence: this could be denying captain's interactivity or notifying the conflict to him. This example illustrates how the ICO notation can address multi-users issues that were introduced in Table 1.

CONCLUSION

This paper has presented the extensions and the use of a formal description technique [19] for the detailed specification of multi-touch and multi-user interfaces. It has built upon previous work done in the area of low-level interaction modeling as proposed in [1] and [2]. It also extended the layered architecture proposed in [26] to address the specific issues of touch interactions.

The paper has provided a thorough design framework for fusion and fission engines in the area of multi-touch interactions addressing on an equal basis input management and output rendering. Due to space constraints only input has been addressed with details in the paper but the proposed combination of the architecture and the notation makes it possible to address output in a similar way as what was presented with input.

Through a set of representative tasks, we have demonstrated on a case study how the ICO notation can tackle multi-modal and multi-users issues providing a detailed and legible notation for interactive critical system specification. All the models presented in the paper can be run in PetShop environment and (provided the paper is accepted) the presentation will demonstrate their execution on a multi-touch platform.

This work is part of a more ambitious work trying to identify the modifications that would have to be added to actual A661 specification standard to address multi-touch and multi-users interactions.

ACKNOWLEDGMENTS

This work is mainly funded by Airbus under the contract CIFRE PBO D08028747-788/2008 and CORAC COCKPIT DGA project.

REFERENCES

1. Accot J., Chatty S., Palanque P. A formal description of low level interaction and its application to multimodal interactive systems. EUROGRAPHICS workshop on "design, specification and verification of Int. syst.", Springer Verlag, (1996), 156-175.
2. Accot, J., Chatty, S., Maury, S., Palanque, P.: Formal Transducers: Models of Devices and Building Bricks for Highly Interactive Systems. EUROGRAPHICS workshop on design, Granada, Spain.. Springer, Heidelberg (1997), 23-37.
3. ARINC 661-5, Prepared by Airlines Electronic Engineering Committee. Cockpit Display System Interfaces to User Systems. ARINC Specification 661-5; (2013)
4. Barbé J., Wolff M. & Mollard R.. 2013. Human centered design approach to integrate touch screen in future aircraft cockpits. In Proceedings of the 15th international conference on Human-Computer Interaction: interaction modalities and techniques - Volume Part IV (HCI'13), Masaaki Kurosu (Ed.), Vol. Part IV. Springer-Verlag, Berlin, Heidelberg, 429-438
5. Barboni E., Conversy S., Navarre D. & Palanque P. Model-Based Engineering of Widgets, User Applications and Servers Compliant with ARINC 661 Specification. 13th conf. on Design Specification and Verification of Interactive Systems (DSVIS 2006), LNCS Springer Verlag. p25-38
6. Bastide R., Navarre D., Palanque P., Schyn A. & Dragicevic P. A Model-Based Approach for Real-Time Embedded Multimodal Systems in Military Aircrafts. Sixth International Conference on Multimodal Interfaces (ICMI'04) October 14-15, 2004, USA, ACM Press
7. Bastide R., Sy O. & Palanque P. A formal notation and tool for the engineering of CORBA systems. Concurrency - Practice and Experience 12(14): 1379-1403 (2000)

8.Bojan Blazica, D. V. & Mladenic V.D. HDCMD: A Clustering Algorithm to Support Hand Detection on Multitouch Displays. SouthCHI 2013: 803-814

9.Buisson M, Bustico A, Chatty S., Colin F-R., Jestin Y., Maury S., Mertz C. & Truillet P. Ivy: un bus logiciel au service du développement de prototypes de systèmes interactifs. Proc. of Conférence Francophone sur l'Interaction Homme-Machine (IHM '02). ACM DL, 223-226.

10.Caelen J., Coutaz J. 1991. Interaction Homme-Machine Multimodale : Problèmes Généraux, IHM'91, Dourdan.

11.Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J., & Young, R. 1995. Four Easy Pieces for Assessing the Usability of Multimodal in Interaction: the CARE Properties. Human Computer Interaction, Interact' 95. Nordby, K., Helmenrsen, P., Gilmore, D., Arnesen, S. Chapman & Hall (IFIP); 95:pp. 115-120.

12. Dang C-T., Straub M., & André E. Hand distinction for multi-touch tabletop interaction. Proc. of the ACM International Conference on Interactive Tabletops and Surfaces (ITS '09).

13. Echtler F. & Klinker G.. 2008. A multitouch software architecture. In Proc. of the 5th Nordic Conf. on Hum.-Comput. Interact: building bridges (NordiCHI '08). ACM, 463-466

14. Fayollas C., Fabre J-C., Navarre D., Palanque P. & Deleris Y. Fault-Tolerant Interactive Cockpits for Critical Applications: Overall Approach. Software Engineering for Resilient Systems (SERENE 2012), Springer-Verlag, p. 32-46, 2012

15. FlySmart with Airbus EFB Manager - Retrieved: 2013-11-24. Available at: https://itunes.apple.com/us/app/flysmart-airbus-efb-manager/id605062108?mt=8

16.Genrich, H. J. 1991. Predicate/Transitions Nets. In High-Levels Petri Nets: Theory and Application. K. Jensen and G. Rozenberg, Springer Verlag (1991) pp. 3-43

17. GestureWorks – true Multitouch for Flash. Website. Retrieved: 2013-11-08. Available at: http://gestureworks.com/

18. Hamon A., Palanque P., Deleris Y., Navarre D. & Barboni E.. A Tool-supported Development Process for Bringing Touch Interactions into Interactive Cockpits for Controlling Embedded Critical Systems. Int. Conf. on Hum.-Comput. Interact. in Aeronautics (HCI'Aero 2012), ACM DL, p. 25-36, 2012

19. Hamon A., Palanque P., Silva J-L., Deleris Y., & Barboni E. Formal description of multi-touch interactions. In Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems (EICS '13). ACM, 207-216

20. Hutchins E. & Lauwsen T. Distributed Cognition in an airline cockpit. In Cognition and Communication at work (Engeström & Middleton Eds.), Cambridge University Press, 1996

21. Kammer D., Keck M., Freitag G. & Wacker M. Taxonomy and Overview of Multi-touch Frameworks: Architecture, Scope and Features. In Proc. of Workshop on Engineering Patterns for Multi-Touch Interfaces, Berlin, Germany, June 2010.

22. Karsenty A., Beaudouin-Lafon M. An Algorithm for Distributed Groupware Applications. ICDCS 1993: 195-202

23. Kenrick K., Björn Hartmann, Tony DeRose, and Maneesh Agrawala. 2012. Proton++: a customizable declarative multitouch framework. In Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12). ACM, New York, NY, USA, 477-486.

24. Kenrick K., Miller T., Bollensdorff B., DeRose T., Hartmann B. & Agrawala M.. Eden: a professional multitouch tool for constructing virtual organic environments. Proc. of conf. on Human factors in computing sys. (CHI '11). ACM, 1343-1352.

25. Khandkar S.H. & Maurer F. A domain specific language to define gestures for multi-touch applications. In Proceedings of the 10th Workshop on Domain-Specific Modeling (DSM '10). ACM, New York, NY, USA, , Article 2 , 6 pages.

26.Ladry J-F., Navarre D., Palanque P. Formal description techniques to support the design, construction and evaluation of fusion engines for sure (safe, usable, reliable and evolvable) multimodal interfaces. ICMI 2009: 185-192

27.Lalanne D., Nigay L., Palanque P., Robinson P., Vanderdonckt J. & Ladry J-F. Fusion engines for multimodal input: a survey. ACM ICMI 2009: 153-160, ACM DL.

28. Lü H. & Li Y. Gesture studio: authoring multi-touch interactions through demonstration and declaration. Proc. of SIGCHI Conf. on Human Factors in Computing Sys. (CHI '13).

29. Micire M., Desai M, Drury J-L., McCann E., Norton A., Tsui K. M. & Yanco H.A. Design and validation of two-handed multi-touch tabletop controllers for robot teleoperation. Proc. of the 16th international conference on Intelligent user interfaces (IUI '11). ACM, New York, NY, USA, 145-154.

30. Navarre D., Palanque P., Ladry J-F. & Barboni E. ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. ACM Trans. Comput.-Hum. Interact., 16(4), 18:1–18:56. 2009

31. NBAA 2013: Touchscreen Cockpit Systems and Next Generation Business Aircraft - Online Article from Aviation Today - Retrieved: 2013-12-01. Available at:

32. Ortega F.R., Hernandez F., Barreto A., Rishe N.D., Adjouadi M. & Liu S. Exploring modeling language for multi-touch systems using petri nets. Proc. of the 2013 ACM int. conference on Interactive tabletops and surfaces (ITS '13). ACM, 361-364.

33. Palanque P., Barboni E., Martinie C., Navarre D., Winckler M: A model-based approach for supporting engineering usability evaluation of interaction techniques. EICS 2011: 21-30

34. Palanque P., Bernhaupt R., Navarre D., Ould M. & Winckler M. Supporting Usability Evaluation of Multimodal Man-Machine Interfaces for Space Ground Segment Applications Using Petri net Based Formal Specification. Ninth Int. Conference on Space Operations, Italy, June 18-22, 2006

35. Palanque P., Ladry J-F, Navarre D. & Barboni E. High-Fidelity Prototyping of Interactive Systems can be Formal too 13th Int. Conf. on Hum.-Comput. Interact. (HCI International 2009) Springer Verlag, LNCS 5610

36. Quinn P.,Malacria S. & Cockburn A. Touch scrolling transfer functions. Proc. of ACM symposium on User interface software and technology (UIST '13). ACM, 61-70.

37. Shneiderman B. (1983). Direct manipulation: a step beyond programming languages, IEEE Computer 16(8), 1983, 57-69.

38. Silva J-L., Fayollas C., Palanque P. & Barboni E. Analysis of WIMP and Post WIMP Interactive Systems based on Formal Specification, Proceedings of the 5th International Workshop on Formal Methods for Interactive Systems (FMIS 2013)

39. The Cockpit of 2030 - Retrieved: 2013-11-24. Available at: http://onboard.thalesgroup.com/2010/intuitive-innovative-interactive-the-cockpit-of-2030/

40. Touch Screens Are Tested for Piloting Passenger Jets - Retrieved: 2013-11-24. Available at: http://www.nytimes.com/2013/07/06/technology/passenger-jets-testing-touch-screen-technology.html?_r=0

41. van Dam A. 1997. Post-WIMP user interfaces. Commun. ACM 40, 2 (February 1997), 63-67