Complexity of Design in Safety Critical Interactive Systems: Gathering, Refining, Formalizing Multi-Type and Multi-Source Information while Ensuring Consistency, Reliability, Efficiency and Error-Tolerance

Sandra Basnyat, David Navarre, Philippe Palanque
(Basnyat, Navarre, Palanque)@irit.fr
LIIHS – IRIT, University Paul Sabatier, Toulouse, 31062, France
http://liihs.irit.fr

Abstract: The design of a usable, reliable and error-tolerant interactive safety-critical system is based on a mass of data of multiple natures from multiple domains. In this paper we discuss the complexity and dangers surrounding the gathering and refinement of this mass of data. This complex and currently mostly informal process can be supported using models that allow handling data at a high level of abstraction. However, not all relevant information can be embedded in a single model. Thus, the various models ought to be consistent and coherent with one another. This paper discusses methodological issues. We present a set of issues raised by the gathering and the modeling of data and some issues raised by their consistency. These issues are addressed in a preliminary unifying framework describing the various models, the data embedded in each model and the interconnections of models.

Keywords: Design, Verification, Safety Critical Interactive Systems, Consistency, Reliability, Error-Tolerance

Introduction

Human-Computer Interaction and related disciplines have argued, since the early days, that interactive systems design requires the embedding of knowledge, practices and experience from various sources. For instance, user centered design (Norman, 1986) advocates the involvement of human factors specialists, computer scientists, psychologist, designers … in order to design useful and usable systems. While designing interactive software, the use of formal specification techniques is of great help as it provides non-ambiguous, complete and concise models. The advantages of using such formalisms are widened if they are provided by formal analysis techniques that allow checking properties about the design, thus giving an early verification to the designer before the application is actually implemented.

During design, one should try consider all stakeholders. That is, "persons or groups that have, or claim, ownership, rights, or interests in a corporation and its activities, past, present, or future. Such claimed rights or interests are the result of transactions with, or actions taken by, the corporation, and may be legal or moral, individual or collective" (Clarkson, 1995). The consideration for all stakeholders leads systems designers and analysts to look at the same system (the one to be designed) from multiple perspectives. Such perspectives come from, but are not limited to domains such as human factors, produce development, training, product management, marketing, the customers, design support, system engineers and interface designers. A number of these domains will be discussed more in detail hereafter and more precisely describing the roles they have in supporting interactive safety-critical systems design.

Due to the large number of domains involved, it is highly unlikely that the data gathered, analyzed and documented will be represented in the same way. For example, it is unlikely that the system engineers will take into account all information provided by human factors analysts (for instance about work practice and users). This is not only because of time constraints and the amount of data involved, but also and mainly, because the kind of notation they are used to employ cannot record that information efficiently. This can have serious effects on the reliability, efficiency and error-tolerance of a system. For example, if a task is represented in a task model by a human factors expert and if that information is not represented (in one way or another) in the system model by a systems engineer there is no means to ensure and check that the system will support this task.

It is clear that there is a need for formalizing not only the process of gathering this mass of data, but also for refining and modeling it when necessary in order to provide valuable input to the system design.

The paper is structured as follows. The next section deals with the issues raised by information gathering per se. Section “Sharing and Embedding Information” discusses the feeding and embedding of information from one phase to another within the design process. Section “Formalizing Information” deals with the need for formalization of information and data. The following sections discuss multi-type and multi-source data respectively. This data has to be gathered throughout the development process in order to allow designers to reach the ultimate goals discussed in section “Ultimate Goals”. The last section (section “Consistency”) presents the consistency problem that has arisen from advocating the use of multiple models.

Gathering Information

The phase of gathering information for the design of a new system is crucial for the success of the end product. If performed incompletely, inaccurately or indeed ignored, gaps are left in understanding the scope, concept and function of the new system.

The process of experts gathering data from various domains for input into the system design has been studied as part of the Mefisto Method. ‘The process cycle’ (Palanque et al., 2000) describes a path that has to be followed to build both usable and reliable interactive systems. In the first phase of the process cycle, the observation phase, information such as work practice, existing artefacts, business and organizational constraints are gathered. Other approaches such as MUSE (Lim and Long, 1994) argue in the same way although the proposed process is different. In that paper, we claimed that in a real life safety critical system, such as in Air Traffic Control (ATC), it is unlikely that the whole domain will be analyzed in detail due to the quantity of data required. This problem will also result in gaps in understanding the scope, concept and function of the new system.

A rich source of information can be obtained from past experiences with similar systems. Since there is such a large amount of data to be gathered, experts can focus on case studies to understand more about the usability of a system and its safety. However, the process cycle (see Figure 1) does not detail how the information is gathered, who will gather it, or how the information will be recorded and reused.

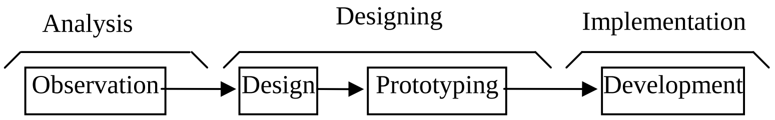

Figure 1 - Schematic view of the Process cycle

Sharing and Embedding Information

Gathering information is not a goal per se. The result of this activity should be used to feed other phases in the design process. This feeding cannot be left informal nor at the discretion of those responsible for these other phases. In addition, not all types of information are closely enough related to build useful bridges between them. On the other hand, some sources of information are so close that, not merging and cross validating them would certainly result in poorly designed and inconsistent systems.

For instance, scenarios and task models both convey information about user activities. It is thus possible to check that scenarios and task models (for the same activity) convey not only the same information but also the same sequencing of operations.

Similarly scenarios and system models both deal with the same operational system and thus ought to contain compatible and coherent information which should be checked at all stages of the development process.

These examples have not been chosen randomly. Indeed, scenarios are the perfect candidate as the corner stone of the consistency and coherence process.

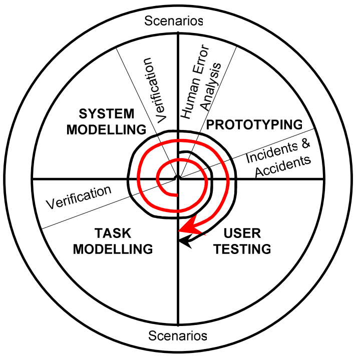

Figure 2 - Scenarios as a corner stone from (Palanque and Navarre, 2000)

Formalizing Information

There is a significant amount of literature on design process for interactive systems design the more referred to being the ones including prototyping activities and evaluations (Dix, 1998 and Hartson and Hix 1992). However little research exists on formalizing the process of 1) documenting the information such that experts of other domains can understand and reuse information for their analysis, 2) refining the information to share only what is necessary and 3) embedding data from one domain to another, all for input into the system design.

Modeling Principles: We promote the use of formal notations so that we can verify the properties of interactive safety-critical systems. Without such notations there are few means for designers to address reliability. However, formal notations may not be adequate for recording information that is idiosyncratically fuzzy and incomplete such as information gathered in the very early phases of the development process. Besides, it is important to note that in most cases, each model will be created by a different person with a different background within a different specialist domain which is likely to influence the kind notation they are able to master. Although it is most likely that one specialist will develop one or several models, they may also contribute to many more models. Thus the relationship between models and specialists can be considered as a many-to-many (M:N). That is, one specialist may contribute to one, zero or many models and one model can receive contributions from one, zero or many specialists. Even for a system that is not safety-critical, it is still necessary to ensure the system's efficiency and reliability but this kind of issue is more salient for this type of system.

Examples of Models: The following section provides an overview of the multiple models used in User Centered Design (UCD) approaches. A number of which can be supported using the UML (Rumbaugh et al., 1997). For example the domain model is supported by class and object diagrams, and the application model which includes the commands and data for the application providers, are the main focus of UML. Some models are only partially accounted for. Task models and scenarios can be described informally and incompletely using UML use cases. Other models are not at all considered in UML for example, user model, platform model and presentation model (Bastide & Palanque, 2003).

We hereafter present more precise information about some particularly relevant models for interactive systems design.

Requirements Model: The functional and non-functional requirements of a system are defined in the requirements model. Requirements describe in a declarative way what a system is supposed to do. The description of a requirement models using a precise and un-ambiguous (i.e. formal) notion allows analysing the model and identifying errors or inconsistencies. In addition, tools can generate tests from the requirement models useful for verifying that a system behaves as the original requirements prescribe (Palanque et al., 1997 and Campos and Harrison, 1997).

Task Model: A task model (Diaper and Stanton, 2004) is a representation of user tasks (in order to reach a certain goal) often involving some form of interaction with a system, influenced by its contextual environment. Task models are used for planning and during various phases of user interface development for example. The models are usually developed by human factor's specialists following an extensive task analysis phase. For the design of interactive safety critical systems, task models can be advantageous for checking the properties of the future system.

User Model: A user model is a collection of information about a user and is a key component for providing flexibility and adaptation. They can incorporate generic information (valid over a wide range of potential users) such as (Card et al., 1983, Fitts 1954, Barnard and May 1994) and represent information about perception, cognition or interaction. Other user models are aimed at representing information for specific users such as (PUMA Blandford and Good, 1997 and OSM Blandford and Connell 2003). This information can be for instance, fed into a system model in the design phase in order to improve flexibility or in the evaluation phase in order to compute predictive performance evaluation (Palanque and Bastide,1997).

Environmental Model: An environmental or contextual model is developed by inspecting aspects of the environment of a current or future system. Information is gathered using techniques such as observation, documentation analysis or interviews. Examples of elements to be studied include location, temperature, artifacts, duration, social aspects and cultural ethics. The model can be used to identify causes of human behavior. Clearly, this can be beneficial for the development of an interactive safety critical system since contextual factors are a way of providing useful adaptation of the system to environmental changes.

Platform Model: A platform model includes a description of the platform and some platform specific characteristics. These models contain information regarding constraints placed on the UI by the platform such as the type of input and output devices available, computation capabilities… The model contains an element for each platform that is supported, and has attributes belonging to each element describing the features and constraints. Although this type of model is particularly useful for ensuring cross-platform compatibility of systems, they are critical when a given system is expected to be made available to several users working with different software and hardware environments.

System Model: System model is, by far, the one that has been studied the most as it is the main raw material of system construction. In the field of interactive systems, most contributions come from the field of software engineering and have been more or less successfully adapted to the specificities of this kind of systems. Since the mid 80s several formalisms have been proposed that were addressing system modeling either at a very high level of abstraction (Dix and Runciman, 1985, Harrison and Dix, 1990) (such as trying to capture the essence of interaction) or at a lower level in order to provide detailed modeling in order to support development activities (Paterno and Faconti, 1992, Palanque and Bastide, 1990). Specific issues raised by interactive systems modeling include, system state, system actions, concurrency, both quantitative and qualitative temporal evolution, input device management, rendering, interaction techniques …

Presentation Model: A presentation model details the static characteristics of a user interface, its visual appearance. The model contains a collection of hierarchically-ordered presentation elements such as sliders, windows and list boxes as far as WIMP user interfaces are concerned. For post-WIMP interfaces such graphical elements include icons, instruments … (Beaudouin-Lafon, 2000 and Van Dam 1997). Current state of the art in the field of safety critical interactive systems is also addressing these issues. For instance, ARINC 661 specification (ARINC 661, 2001) provides a detailed description of interactive components and their underlying presentation platform for new generation of interactive cockpits.

Architectural Model: An architectural model is a high level model of the application which describes the basic building blocks of the application. Examples of established architectural models are Seeheim model (Green, 1985) which makes explicit the user interface part of the application and the Arch model (Bass et al., 1991) which is an extension of the Seeheim model putting even more emphasis on the UI part. The Arch model divides all user interface software into the following functional categories, Functional Core, Functional Core Adapter, Dialogue, Logical Interaction and Presentation. From a modeling point of view, these components are usually dealt with individually. Various modeling techniques are applied to deal with these components and the following section address some of them i.e. domain model (related to functional core modeling) dialogue model and device model (a sub-part of the presentation component).

Domain Model: A domain model is an explicit representation of the common and the variable properties of the systems in a domain and the dependencies between the variable properties. (Czarnecki and Eisenecker, 2000). The model is created by data collection, analysis, classification and evaluation. The term domain covers a wide range of interpretations, for example, the problem domain, business domain and the system/product domain.

Theses models are necessary to understand the domain in which the future system will be built. In the field of safety critical systems the various domains involved (such as ATC, military systems …) have already received a lot of attention. Domain models are readily available and are meant to be exploited before dealing with any system within that domain.

Dialogue Model: A dialogue model is a collection of hierarchically-ordered user-initiated commands that define the procedural characteristics of the human-computer dialogue in an interface model. (Puerta, 2002). Dialogue modeling has been regarded as a particularly hard to tackle issue. A lot of work has been devoted to it and the notations used have evolved in conjunction with interaction techniques. For instance, early work focused on modal interaction techniques (Parnas 1969) and evolved to WIMP interaction styles (Bastide & Palanque 1990) to reach recent and more demanding interaction techniques as in (Dragicevic et 2004 DSVIS) for multimodal interaction.

Device Model: Input and output devices are a critical part of the interactive systems as they represent the bottleneck via which the interaction between users and system takes place. Their behavior is sometimes very complex even though it may be perceived as simple by the users. This complexity may lie in the device itself (as for haptic devices such as the Phantom (Massiem and Salisbury; 1994)) or in the transducers in charge of extending the behaviors of the devices (such as extending the behaviour of a mouse to cope with double or triple clicks that embed temporal constraints) (Buxton 1986, Accot et al.; 1996). Device models can also be viewed as a person's understanding of how a device works (Satchwell, 1997). In the field of safety critical systems describing the behavior of such devices is critical as it makes precise the interaction techniques.

Multi Type Data

The data obtained and analyzed by various domain experts can be considered as multi-type data. We have distinguished between two main types of data, pre-design data and post-design data. That is, data that is available before a system has been designed, and data that is available after a system is designed. This distinction and its impact on systems design are explained in more detail in the following sections.

Pre-design data: Data can be obtained throughout the design process before the system has been developed. Of course, much of this data can be made available and used for evaluation purposes, once a system has been designed. However; we have labeled it pre-design data because the techniques can be applied without the need of the current system.

Within this category of pre-design data, data can be further classified according to the properties of the data obtained. That is, formal or informal, complete or incomplete for example. Figure 3 illustrates on a three-dimensional cube, four examples of techniques that can be applied to obtain data before the system has been designed. By formal and informal we mean whether there only one interpretation of the models or not. Complete and incomplete refer to the fact that the model contains a sub set of the relevant information or deals exhaustively with it. Finally, high and low-level data refer to level of abstraction at which the information is dealt with.

To illustrate the complexities surrounding multi-type data, we have provided an example of seven techniques positioned in the Multi-Type Data Cube. Some of the examples presented in more detail later in this section, have been extracted from previous work on a mining accident case study (Basnyat et al. 2005).

This type of presentation is used because of the overlapping properties of the techniques. For example, a Petri-net is considered (in this paper) as formal, complete and low level even though it is possible to use them to represent other type of data.

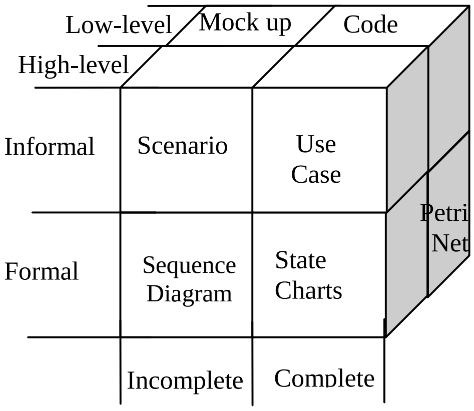

Figure 3 – Multi-Type Data Cube

To give a very brief overview, the case study is a fatal US mining accident (Mine Safety and Health Administration 2002). A Quarry and Plant system is designed to produce cement. However, the part we focus on is the delivery of waste fuel used to heat the plant kilns. The Waste Fuel Delivery System is comprised of two separate liquid fuel delivery systems, the north and the south. Each system delivers fuel to the three plant kilns independently and cannot operate at the same time.

Example of low level formal complete data: Figure 4 provides a simple Petri-net which models the ability to switch from the north waste fuel storage tank to the south waste fuel storage tank using a manual shut off valve.

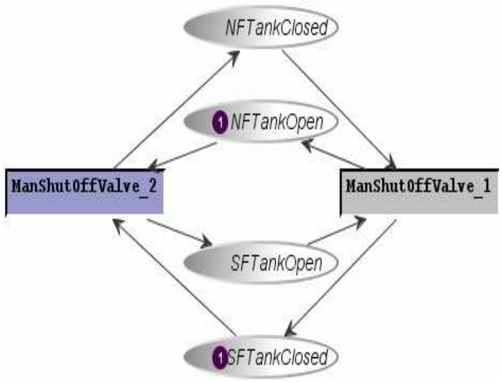

Figure 4 - Formal low level and complete data modeling using Petri-nets

Example of incomplete, informal and low level data: In safety-critical interactive systems design, scenarios can be used to elucidate the particular chain of events that lead to an accident but can also be used to identify alternate failure scenarios that might cause future adverse events. In this particular case study, it could be argued that as a result of the user's actions described in the following brief scenario, a 'hammer effect' occurred causing a fatal explosion. "Mr X closed the valves (after bleeding them) as quickly as possible because of the threat of fuel spreading."

One of the problems associated with ensuring consistency, reliability, efficiency and error-tolerance in the design of an interactive safety-critical system, lies in the probable limited use of fruitful information.

Scenarios can be used in line with many techniques, such as task modeling, a priori and a posterior i.e. for design or evaluation activities. A careful identification of meaningful scenarios allows designers to obtain a description of most of the activities that should be considered in the task model. (Paterno & Mancini, 1999). Example of incomplete, formal and high level data: Figure 5 illustrates the event-based sequence diagram that can be used to map out what happened in the lead-up to an adverse event.

Post-design data: The second distinction of data we have made is post-design data. By this, we mean data that can only be obtained once the system in mind has been designed. Examples of such are usability analysis, incident and accident reports or the use of metrics for risk analysis (Fenton and Neil, 1999).

The design of a safety-critical interactive system must be grounded on concrete data, of which may be of multiple source and of multiple type. However, an additional way to compliment and enhance a system's safety is to take into account as much information from previous real life cases. One such type of data is an incident or accident report. To date, input to a safety-critical interactive system design from an incident or accident report has not been considered in a systematic way. We believe these reports can be extremely fruitful to the design of safer safety critical systems. In most cases, these reports are used by assigned experts to analyse why an incident or accident occurred and what could be changed to prevent future similar scenarios from occurring. In contrast, we suggest using the reports to improve future design. To be more concrete, we have implemented this approach on the same mining accident case study previously mentioned.

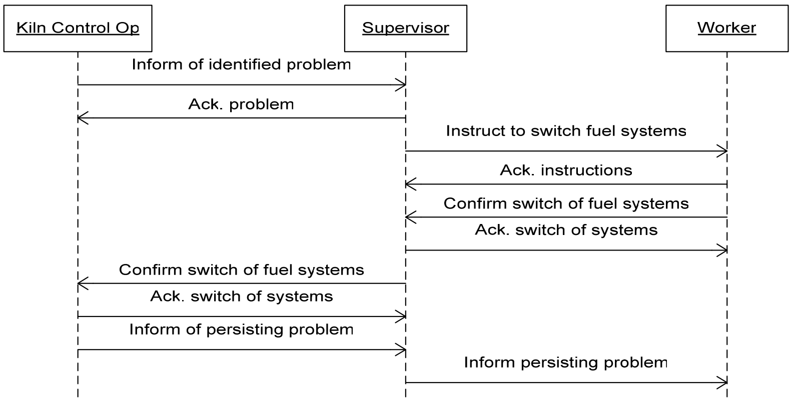
Kiln Control Op
Supervisor
Worker
Inform of identified problem
Ack. problem
Instruct to switch fuel systems
Ack. instructions
Confirm switch of fuel systems
Ack. switch of systems
Confirm switch of fuel systems
Ack. switch of systems
Inform of persisting problem
Inform persisting problem

Figure 5 - High-level data, communication sequence diagram

The reports allowed us to achieve two things, 1) obtain and 2) deduce important information that could be embedded into future waste fuel delivery systems of mining plants. Such information obtained includes:

- Add additional fire sensors in the waste fuel containment area to detect heat from fire and activate the fire suppression system more rapidly. Ensure the Programmable Logic Controller (PLC) connectors are properly installed.
- Implement procedures requiring all equipment operators and their supervisors to review manufacturers' instructions and recommendations to ensure machinery and equipment is operated according to manufacturer's guidelines.
- Install audible and/or visual alarm systems in the waste fuel containment area.
- Ensure equipment is installed according to the manufacturer's requirements. Develop procedures and schedules and monitor them to ensure that the required maintenance is performed

Information deduced after implementing and analyzing the results of various safety analysis techniques resulted in the following findings. The system should be designed such that:

- A waste fuel delivery system cannot be started without being primed first.
- Motors cannot be turned on without fuel available in the pipes.
- Air is bled from the pipes before a fuel delivery system is turned on.
- Air cannot be bled while a waste fuel delivery system is on.

- An emergency shutdown button should available to operators.

Multi-Source Data

The data gathered and analyzed for input into a safety-critical interactive system design is collected by multiple specialists of a wide-array of domains. This is due to the nature of safety-critical systems that range from cockpits to surgical equipment to mining instruments to name just a few but also to the variety of information that has to be gathered and the fact that this information stems from multiple domains of expertise. This combination of diverse specialists and diverse domains adds to the complexity of design of a safety-critical system. The following sections describe several such specialists and domains and the input they have on the design.

Human Factors: Human factors is a domain which aims to put human needs and capabilities at the focus of designing technological systems to ensure that humans and technology work in complete harmony, with the equipment and tasks aligned to human characteristics (Ergonomics Society).

Examples of human factors specialists are production engineers, health and safety- practitioners and interface designers. These are just a number of experts in the human factors field who all bring advantages to the design of the system. However, the complexity increases when considering the background of these experts and the ways in which their analyses will vary according to their backgrounds.

Health and Safety Practitioners: Occupational Health and Safety (H&S) practitioners are trained in the recognition, evaluation and control of hazards which place people's safety and health at risk in both occupational and community environments.

Techniques employed by H&S practitioners include risk assessments, postural analysis, legal and organizational factors, work equipment. As with most occupations, health and safety practitioners also have wide ranging educational backgrounds. Such as psychology, anthropometry or physiology. This results in multiple perspectives and methods of working on the same system.

Interface Designers: An Interface Designer is responsible for the presentation of the interface part of an application. Although the term is often associated to computing, the interactive part of a system can include controls and displays in many domains such as military aircraft, vehicles, audio equipment and so on. The educational background of an interface designer can be varied, computer science, graphics design or again psychology. It is probable that a psychologist and a computer scientist will base their interface designs on different principles. Stereotypically, for example, a psychologist may wish to ensure correct colors are used, whereas a computer scientist will want to employ the latest programming techniques with a flashy interface. Both perspectives can be advantageous to the overall design.

Engineering: Systems engineering is an interdisciplinary process referring to the definition, analysis and modeling of complex interactions among many components that comprise a natural system (such as an ecosystem and human settlement) or artificial system (such as a spacecraft or intelligent robot), and the design and implementation of the system with proper and effective use of available resources. (University of Waterloo). In the mining case study, mechanical and automation engineers were involved. However, other types of engineers include hardware, software and systems engineers. The combination of these engineers assists in the system development process.

Hardware Engineer: In reference to the case study, we can assume that the hardware engineers would have been responsible for the design and development of plant components such as the motors, grinders and fuel tank.

Software Engineers: The software engineers in the mining case study would have been responsible for the design and development of applications running on the hardware. Programs include the PLC software and the ‘F’ system software.

Mechanical Engineers: A mechanical engineer can have a variety of responsibilities such as, the design and improvement of machines and mechanisms, organization and maintenance of computer controls for production processes or even selection and installation of equipment for indoor environment control.

Automation Engineer: Automation engineers design, build and test various pieces of automated machinery. This can include electrical wiring, tooling, software debugging etc. One of the main fields of an automation engineer is to design automation systems from a collection of single components of different distributors.

Engineering and the Case Study: A combination of the work performed by the above mentioned engineers can be considered as partial cause for the fatal accident in the case study. One of the events leading to the accident was the failure of the PLC to automatically de-energize the fuel in the pipes when it received signals that the pressure was too high. This automated procedure operated as follows. A monitoring 'F' system received signals from temperature and pressure sensors located on fuel lines. The 'F' system transmits data to the PLC which raises audible and visible alarms in the control room. However, during the accident, the PLC was not connected and therefore did not automatically de-energize the pressure in the pipes.

Certification: Certification is a phase of the development process specific to safety critical systems. This activity involves independent organizations responsible for providing clearances prior to the actual deployment of the systems. This activity has a significant impact over the development process as its successful accounting is perceived by designers and developers as one of the main targets to achieve. Indeed, in case of certification failure, the whole development can be stopped and most of the time restarted with many negative economical and technological consequences. For this reason, certification authorities have developed specific development processes that 'guarantee' the quality of the product by means of structured and reliable processes. For instance DO 178 B (RCTA 1992) is a document describing such a design process widely used in the aeronautical domain.

Incident and Accident Analysts: Incident and accident analysts are interested in understanding system 'failures' and human 'error' often using accident analysis techniques and incident reporting techniques. (http://www.dcs.gla.ac.uk/research/gaag). Such analysts have varying educational backgrounds in computer science for example.

Since we are particularly interested in the domain of safety-critical systems, we have provided definitions of an incident and accident from the Federal Aviation Administration (FAA). An aircraft accident means an occurrence associated with the operation of an aircraft which takes place between the time any person boards the aircraft with the intention of flight and all such persons have disembarked, and in which any person suffers death or serious injury, or in which the aircraft receives substantial damage. (49 CFR 830.2). An aircraft incident is an occurrence other than an accident, associated with the operation of an aircraft, which affects or could affect the safety of operations. (49 CFR 830.2)

Ultimate Goals

The above mentioned issues increase complexity in the design of interactive safety critical systems due to the necessary ultimate goals of embedding reliability, usability, efficiency and error tolerance with the end product. Without such ultimate goals the development process would be far less cumbersome. This is a very important aspect of the work presented here as it points out the issues that are specific to the type of applications we are considering and thus less relevant to others more commonly considered.

Consistency

Consistency is a means to achieve reliability, efficiency, usability and error-tolerance of a system. This can be achieved by means of systematic storage of gathered information into models and the development of techniques for cross models consistency checking.

Model Coherence: One of the problems associated with interactive safety-critical design is the lack of coherence between multiple viewpoints and therefore multiple design models, of the same world. We believe there should be coherence between these design models to reduce the likelihood of incidents or accidents in the safety-critical systems domain. Some work on model-based approaches has tried to address these issues but there is still a lot to do before design methods actually provide a framework to support this critical activity. Indeed, it is still not commonly agreed that there should be a framework for multiple models as some current research argues that models of one kind could be generated from models of the other kind. For instance (Paternò et al., 1999) proposes to generate user interfaces from task models while (Lu et al. 1999) proposes to generate task models from system models.

A Generic Framework for Ensuring Coherence: Although highly beneficial, it is unlikely that all techniques from all domains of all types of experts will be applied to the design of any given system. This is an unfortunate reality and this is why we are trying to focus on providing a unifying framework to help ensure that data of multiple domains can be gathered, refined and embedded into the design of the system.

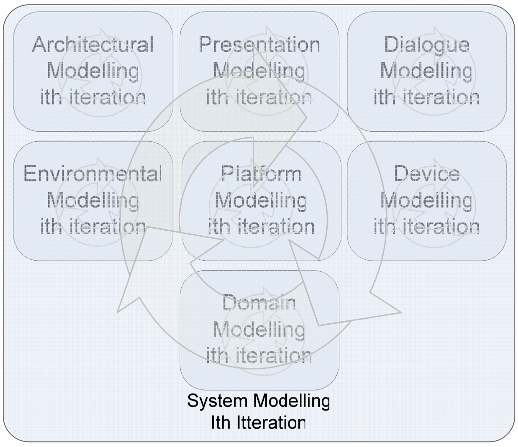
Architectural Modelling ith iteration
Presentation Modelling ith iteration
Dialogue Modelling ith iteration
Environmental Modelling ith iteration
Platform Modelling ith iteration
Device Modelling ith iteration
Domain Modelling ith iteration
System Modelling Ith Itteration

Figure 6 - Ingredients of the system model

As previously mentioned, formalizing this unified procedure is a way of ensuring that there are no ambiguities, that the description of the models and information is precise, that the framework allows reasoning about the system and to ensure consistency and coherence throughout the design and development process.

Figure 6 presents the various ingredients of the system part as described in the section detailing various types of models. This component is reproduced in Figure 7 where interactions with other models is emphasized. Figure 7 presents, as a summary and in a single diagram the set of information, data and processes.

Need For Systematic Tools Support: The complexity of design in the field of safety critical interactive systems clearly requires tool support for the creation, edition; formalisation; simulation, validation; verification of models and information, ability to check for inconsistencies; means for sharing and embedding data; cross-checking of hybrid models … To date, tools exist for the support of individual models, CTTe (Paterno et al., 2001) for supporting various activities around task modeling (edition, simulation, verification …), Petshop (Bastide et al., 1999) for supporting various activities around system modeling. Despite some preliminary work about interaction (Navarre et al. 2001) integration needs are still to be addressed.

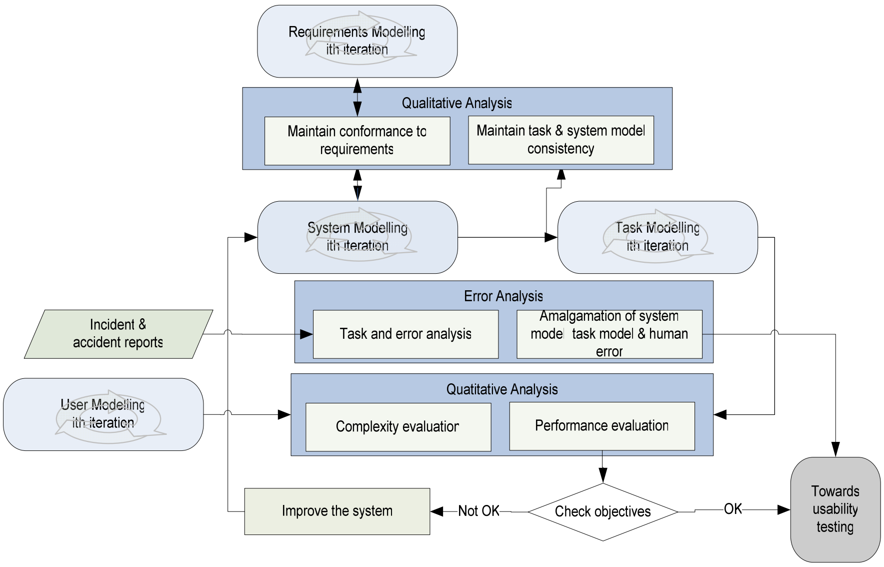

Figure 7 - Generic Modeling Framework

Conclusion

This paper discussing methodological issues, advocates the use of models for the design of interactive safety critical systems. It claims that the issues raised by the design of such systems require the use of systematic ways to support the gathering, refinement and storage of data. This data is, by nature, multi-disciplinary and thus requires a multi-notation approach to support individually each discipline.

However, this multi-notation approach calls for additional means in order to support additional activities such as verification of models consistency. Besides, in order to alleviate the burden for developers and designers, software tools supporting their activities are also at the core of the applicability of such an approach.

We are currently studying methods for integrating the necessary models for safety critical interactive systems design. To date, we have devised two approaches for integrating the task model and system model while taking into account human errors. One approach uses scenarios are bridge between the two (Navarre et al. 2001). The second approach uses task patterns as a means of cross-checking properties between the two models. This work is part of more ambitious work dealing with multiple models for safety critical interactive systems in several application domains including satellite command and control room, interactive cockpits for military and civilian aircrafts, command and control rooms for drones and air traffic control workstations.

References

Accot, J., Chatty, S., Palanque, P.(1996)A Formal Description of Low Level Interaction and its Application to Multimodal Interactive Systems, In Proceedings of the Third Eurographics workshop on Design, Specification and Verification of Interactive Systems, (DSV-IS 96) F. Bodard & J. Vanderdonckt Eds. Springer Verlag 1996.pp. 92-104

ARINC 661 Cockpit Display System Interfaces to User Systems. Arinc Specification 661. April 22, 2002. Prepared by Airlines Electronic Engineering Committee.

Barnard, P. and May, J. (1994) Interactions with Advanced Graphical Interfaces and the Deployment of Latent human Knowledge. Interactive Systems: Design, Specification and Verification. DSVIS 1994 pp15-49

Bass, L., Little, R., Pellegrino, R., Reed, S., Seacord, R., Sheppard, S., and Szezur, M. R. (1991). The Arch Model: Seeheim Revisited. User Interface Developpers' Workshop. Version 1.0 (1991)

Bastide, R., Palanque, P., Sy, O, Duc-Hoa Le, Navarre, D. (1999) PetShop a case tool for Petri net based specification and prototyping of Corba Systems. Tool demonstration with Application and Theory of Petri nets ATPN'99, Williamsburg (USA), LNCS Springer Verlag, 1999.

Bastide, R and Palanque, P. (2003) UML for Interactive Systems: What is Missing in Workshop on Software Engineering and HCI, INTERACT 2003, IFIP TC 13 conference on Human Computer Interaction.

Bastide, R., Navarre, D., Palanque, P. and Schyn, A. (2004) A Model-Based Approach for Real-Time Embedded Multimodal Systems in Militart Aircafts. Sixth International Conference on Multimodal Interfaces. ICMI'04 October 14-15, 2004 Pennsylvania State University, USA.

Basnyat, S., Chozos, N., Johnson, C., and Palanque, P. (2005) Multidisciplinary perspective on accident investigation. Submitted to the Special issue of Ergonomics on Command and Control.

Beaudouin-Lafon, M. (2000). Instrumental interaction: an interaction model for designing post-WIMP user interfaces. CHI 2000: 446-453

Blandford, A. & Connell, I. (2003) Ontological Sketch Modelling (OSM): Concept-based Usability Analysis Proc. Interact 2003. 1021-1022. IOS Press.

Blandford, A. and Good, J. (1997) Programmable user models - exploring knowledge transfer between devices. PUMA working paper WP5.

Booch, G., Rumbaugh, J., Jacobson, I. (1999) The Unified Modelling Language User Guide. Addison-Wesley

Buxton, W. & Myers, B. (1986). A study in two-handed input. Proceedings of CHI '86, 321-326

Campos, J. C. and Harrison, M. D. (1997) Formally verifying interactive systems: A review. In M. D. Harrison e J. C. Torres, editors, Design, Specification and Verification of Interactive Systems '97, Springer Computer Science, pp 109--124. Springer-Verlag/Wien, Junho 1997.

Card, S.K., Moran, T.P. & Newell, A. (1983). The Psychology of Human-Computer Interaction, Lawrence Erlbaum, New Jersey

Carroll, J. M. (1995). Introduction: the scenario perspective on system development. In J. M. Carroll (Ed.) Scenario-based design: envisioning work and technology in system development (pp. 1-18). New York: John Wiley & Sons, Inc.

Clarkson, M. B. E. (1995). A stakeholder framework for analyzing and evaluating corporate social performance. Academy of Management Review, 20: 39-48

Czarnecki, K and Eisenecker U. W. (2000). Generative Programming—Methods, Tools, and Applica-tions. Addison-Wesley, 2000. ISBN 0-201-30977-7

Diaper, D. and Stanton, N.A. (2004) The Handbook of Task Analysis for Human-Computer Interaction. Lawrence Erlbaum Associates.

Dix, A. and Runciman, C. (1985). Abstract models of interactive systems. People and Computers: Designing the Interface, Ed. P. J. &. S. Cook. Cambridge University Press. pp. 13-22.

Dix, A., Finlay, J., Abowd, G., & Beale, R. (1998). Human-computer Interaction. Prentice Hall, Second Edition, Prentice Hall Europe. ISBN: 0-13-239864-8.

Fenton N. E. and Neil M, (1999). Software Metrics and Risk, Proc 2nd European Software Measurement Conference (FESMA'99), TI-KVIV, Amsterdam, ISBN 90-76019-07-X, 39-55, 1999.

Fitts, P. M. (1954) "The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement." Journal of Experimental Psychology 47. pp. 381-91

Green, M. (1985). Report on Dialogue Specification Tools", in G. Pfaff (ed.). User Interface Management Systems. New York: Springer-Verlag, 1985, 9-20.

Harrison, M and Dix, A. (1990) State Model of Direct Manipulation in Interactive Systems. In Formal Methods in Human-Computer Interaction. Cambridge Series on HCI. Edited by M. Harrison and H. Thimbleby. P.129

Hix, D. and Hartson, H.R. (1993). Developing user interfaces: ensuring usability through product and process. John Wiley and Sons, New York. ISBN: 0-471-57813-4.

Lim, K.Y and Long, J.B (1994) The MUSE Method for Usability Engineering. Cambridge University Press, Cambridge.

Lu, S., Paris, C., Vander Linden, K. (1999) Toward the automatic construction of task models from object-oriented diagrams. Engineering for Human-Computer Interaction. Kluwer Academic Publishers. 169-180.

Massiem T. H. and Salisbury J. K.. (1994) The phantom haptic interface: A device for probing virtual objects. In Proc. of the ASME Winter Annual Meeting, Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, Chicago, IL, USA, November.

Navarre, D., Palanque, P., Bastide, R., Paternó, F., and Santoro, C. (2001). "A tool suite for integrating task and system models through scenarios." In 8th Eurographics workshop on Design, Specification and Verification of Interactive Systems, DSV-IS'2001; June 13-15. Glasgow, Scotland: Lecture notes in computer science, no. 2220. Springer

Norman, D. A. (1986). Cognitive Engineering. In D. A. Norman & S. Draper (Eds.). User centered system design: New perspectives in human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc

NTSB Title 49 – Transportation, Subtitle B – Other Regulations Relating to Transportation, Chapter VIII – National Transportation Safety Board. Part 830 - Notification and Reporting Of Aircraft Accidents or Incidents and Overdue Aircraft, and Preservation of Aircraft Wreckage, Mail, Cargo, and Records. Section 830.2 Definitions.

Palanque, P and Bastide, R (1990) Petri nets with objects for specification, design and validation of user-driven interfaces. In proceedings of the third IFIP conference on Human-Computer Interaction, Interact'90. Cambridge 27-31 August 1990 (UK)

Palanque, P and Bastide, R. (1997). Synergistic modelling of tasks, system and users using formal specification techniques. Interacting With Computers, Academic Press, 9, 12.

Palanque, P., Bastide, R., Paternò (1997) Formal Specification as a Tool for Objective Assessment of Safety-Critical Interactive Systems. INTERACT 1997: 323-330

Palanque, P., Navarre, D. (2000) Gaspard-Boulinc, H. (2000) MEFISTO Method version 1. The Mefisto Project ESPIRIT Reactive LTR 24963 Project WP2-7. September 2000.

Parnas, D.L (1969). On the use of transition diagrams in the design of a user interface for an interactive computer system. In Proceedings 24th National ACM Conference, pp. 379-385.

Paternò F., Mancini, C. (1999) Developing Task Models from Informal Scenarios, Proceedings ACM CHI'99, Late Breaking Results, ACM Press, Pittsburgh, May 1999.

Paternò. F. and Faconti, G. (1992), in Monk, Diaper & Harrison eds. On the Use of LOTOS to Describe Graphical Interaction People and Computers VII: Proceedings of the HCI'92 Conference, Cambridge University Press, pp.155-173, September, 1992.

Paternò, F., Breedvelt-Schouten and N. de Koning. (1999). Deriving presentations from task models. In Engineering for Human-Computer Interaction. Kluwer Academic Pub. pp. 319-338.

Paternò, F., Mori, G. and Galimberti, R. (2001) CTTE: An Environment for Analysis and Development of Task Models of Cooperative Applications, In ACM Proceedings of (SIGCHI'2001), March 31-April 5, Seattle, WA. (Extended Abstracts). 21:22

Puerta, A.R. (2002) The MECANO Project: Comprehensive and Integrated Support for Model-Based Interface Development. In (CADUI'02), pp. 19-35.

RTCA/DO-178B. Software Considerations in Airborne Systems and Equipment Certification, December 1. http://www.rtca.org/ (1992)

Rumbaugh, J, Jacobson, I and Booch, G. (1997) Unified Modeling Language Reference Manual, ISBN: 0-201-30998-X, Addison Wesley, est. publication December 1997

Satchwell, R.E. (1997) Using Functional Flow Diagrams to Enhance Technical Systems Understanding. Journal of Industrial Teacher Education. Volume 34, Number 2. Winter 1997

Stirewalt, K., and Rugaber., S. (1998) Automating UI Generation by Model Composition. Automated Software Engineering 13th IEEE International Conference October 13-16, 1998

United States Department Of Labor Mine Safety And Health Administration Report Of Investigation Surface Area Of Underground Coal Mine Fatal Exploding Pressure Vessel Accident January 28, 2002 At Island Creek Coal Company Vp 8 (I.D. 44-03795) Mavisdale, Buchanan County, Virginia Accident Investigator Arnold D. Carico Mining Engineer Originating Office Mine Safety And Health Administration
District 5 P.O. Box 560, Wise County Plaza, Norton, Virginia 24273 Ray Mckinney, District Manager Release Date: June 20, 2002

van Dam, A. (1997) Post-WIMP User Interfaces, Communications of the ACM 40, 2, 1997, 63-67

Websites

Enterprise Architect User Guide Version 4.5
http://www.sparxsystems.com.au/EAUserGuide/index.html?sequencediagram.htm

http://www.ergonomics.org.uk/ergonomics/definition.htm

University of Waterloo. What is systems design engineering?
http://sydewww.uwaterloo.ca/SystemsDepartment/WhatIsSystems/whatissystems.html.

http://www.dcs.gla.ac.uk/research/gaag