A Framework for Critical Interactive System Formal Modelling and Analysis

Ismail Mendil(B)

IRIT/INPT-ENSEEIHT, 2 rue Charles Camichel, 31071 cedex 7 Toulouse, France
ismail.mendil@toulouse-inp.fr

1 Introduction

The human-computer interface (HCI) is the component of an interactive system that allows users to interact with a system. Interactive system development does not follow the same life cycle as software system development. The essential differences lie in the iterative nature of the interactive system development. Hence, the completion of such systems requires usually several iterations. Throughout iterations, the requirements undergo many changes due to the evolution of customer’s needs and user feedback after experiencing the prototypes. Furthermore, the formalization of user interaction requirements into HCI specification is a complex task that needs a thorough observation of user behaviour. The challenge is tougher in the case of critical HCI (cockpits, medical systems, etc.). Indeed, critical HCI requires to be designed and built such that safety is put at the forefront of the requirements dictated by standards and norms.

Formal methods offer both theoretical background and support tool enabling the validation and verification of system specification beforehand, i.e. before the system is put into production. In fact, such methods allow dealing with abstract mathematical models of the system on which mathematical proofs are performed. So, different kinds of properties (e.g. safety and usability) are proved, hence providing a higher level of confidence in the system. Several modelling formalisms and tools have been developed to model, analyse and animate HCI models, however, they do not offer HCI domain knowledge integration at modelling level.

In our Ph.D. project, we aim to complete an operational framework for formal verifying and validating critical interactive systems with a special emphasis on the aeronautic field.

2 Challenges

As part of the ANR FORMEDICIS project, an abstract-level language describing the interaction between the system and the user is being developed: FLUID (Formal Language for User Interface Design). The core raison d’être of this language is to allow the design and analysis of critical interactive systems at a high level of abstraction. This endeavour faces, however, several challenges:

1 ANR (French National Research Agency), https://anr.fr/Projet-ANR-16-CE25-0007.
Defining of a domain theory to formalize HCI design knowledge.
- Ensuring effective integration of the user’s requirements in the design process (Human In The Loop).
- Endowing FLUID language with verification capabilities.
- Supplying FLUID with animation extension for the sake of validation.
- Assessing and evaluating the FLUID expressivity and its easiness of use.
- Contributing to the certification of critical interactive systems.

3 FLUID Language

3.1 FLUID Language Main Features

In this section, we sketch the FLUID language definition. It allows describing and verifying the system behaviour. Moreover, it addresses the lack of integration of domain-specific knowledge in HCI development at modelling level. FLUID models are built out of the INTERACTION component which consists of three parts: contextual part (DECLARATION), behaviour part (STATE and EVENT) and properties part (ASSUMPTIONS, EXPECTATIONS and REQUIREMENTS) (Fig. 1).

FLUID models are state-based with interleaving asynchronous event-driven semantics. It allows the designer to express the user requirements in different ways either as general properties (e.g. usability, safety) abstracted away from requirements, or individual scenarios conveying some expected storyboard (SCENARIOS.NOMINAL) or, at the contrary, precise forbidden scenarios (SCENARIOS.NONNOMINAL).

Variables and events in FLUID models are annotated with tags whose definitions lie in domain theory. Hence, we split the domain-specific knowledge and constraints apart from the system model. In addition, proof obligations are generated to integrate the HCI domain constraints into the verification and validation process.

3.2 Analysing FLUID Models: Event-B and ICO

In our work, two formalisms and two tools are devised to support FLUID language. For the purpose of verifying FLUID models, Event-B\(^2\) will be used. It

\(^2\) http://wiki.event-b.org/.
is a formal method for system-level modelling and analysis. Basically, it relies on set theory and first-order logic to formalize systems into models and uses refinement to represent systems at different abstraction levels. Furthermore, it leverages mathematical proof theory for discharging proof obligations [1]. The Event-B language defines two main building blocks to model, in principle, any system: firstly, a context construct for describing the static characteristics of a system through carrier sets, constants, axioms and theorems, secondly, a machine building block for expressing the dynamic aspects through variables, invariants, theorems, variants and events. Rodin\(^3\) [2] is an Integrated Development Environment (IDE) for Event-B modelling language based on Eclipse. It has many features making the modelling and V&V process easier and more efficient. Rodin features project management, incremental model development, proof assistance, model checking, animation and automatic code generation.

For the sake of animation and validation of FLUID models, ICO [6] associated to its PetShop [7] tool will be used as a basis. ICO formal modelling is dedicated to expressing and describing interactive systems. Interactive Cooperative Objects follows the object oriented-paradigm so it incorporates concepts such as dynamic instantiation, classification, encapsulation, inheritance and client/server relationships. It provides means to model the static side of the interactive system inspired by the object-orientation, and it uses the Petri Nets notation to express the behavioural side. Moreover, the ICO notation is fully supported by the PetShop CASE tool.

4 Methodology and Approach

In order to bridge the gap and build a framework resolving the identified challenges listed in Sect. 2, we envision to meet the objectives we present hereafter:

- Modelling HCI domain using Event-B theory extension as supported by Rodin Theory Plugin\(^4\). Indeed, HCI domain theories connected to design models via tags and the automatic proof obligation generation shall be viewed as a major feature of our approach.
- Developing a transformation schema to embed FLUID models in Event-B [1] [4] to incrementally design interactive system models [3]. The schema needs to integrate the specification and verification of scenarios and properties. This dual specification of requirements is a key feature of our framework addressing the HCI development process, in particular handling of storyboards.
- Defining and implementing transformation rules in order to dive FLUID and/or Event-B models into ICO for the sake of animation using PetShop.
- For the purpose of assessment and evaluation of the framework, case studies will be developed and extended to comply with the HCI domain-specific constraints. In addition, formal methods (Event-B, ICO, LIDL [5]) will be compared, on the basis of different criteria (expressivity, conciseness, etc.), to our framework through development of case studies.

\(^3\) http://www.event-b.org/install.html.
\(^4\) http://wiki.event-b.org/index.php/Theory_Plug-in.
5 What’s Next?

This article framed the problem tackled in our Ph.D. research project and identified its coarse-grained challenges: mainly, the need for a comprehensive abstract-level HCI modelling framework covering the development cycle of critical HCI. Then, the high-level objectives are set down and enumerated.

Within our Ph.D. work, the next steps are twofold. Firstly, we plan to develop an HCI domain theory to enrich models at the early stages of system development. This theory must encompass the domain knowledge and it connects to the system model via the tag notation. Next, the transformation schema needs to be formalized and implemented, allowing the translation from FLUID into Event-B and ICO. Then, in order to demonstrate the effectiveness, cases studies are planned to be developed based on the novel framework. Finally, a development process will be defined around the novel framework so to facilitate and structure HCI developments. Ultimately, we believe that a successful landing of our project would provide more confidence to tackle the bigger problem of extending our framework to other system development fields.

References

1. Abrial, J.R.: Modeling in Event-B - System and Software Engineering. Cambridge University Press, Cambridge (2010)
2. Abrial, J.-R., et al.: Rodin: an open toolset for modelling and reasoning in Event-B. Int. J. Softw. Tools Technol. Transf. 12(6), 447–466 (2010)
3. Aît-Ameur, Y., et al.: Encoding a process algebra using the Event B method. STTT 11, 239–253 (2009). https://doi.org/10.1007/s10009-009-0109-2
4. Geniet, R., Singh, N.K.: Refinement based formal development of human-machine interface. In: Mazzara, M., Ober, I., Salaün, G. (eds.) STAF 2018. LNCS, vol. 11176, pp. 240–256. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04771-9_19
5. Lecrubier, V.: A formal language for designing, specifying and verifying critical embedded human machine interfaces. Ph.D. thesis. ISAE - Université Toulouse, June 2016. https://hal.archives-ouvertes.fr/tel-01455466
6. Navarre, D., et al.: ICOs a model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. ACM Trans. Comput. Hum. Interact. 16, 18:1–18:56 (2009)
7. Palanque, P., Ladry, J.-F., Navarre, D., Barboni, E.: High-fidelity prototyping of interactive systems can be formal too. In: Jacko, J.A. (ed.) HCI 2009. LNCS, vol. 5610, pp. 667–676. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02574-7_75