Proceedings of the Workshop on Formal Methods in Human-Machine Interaction (Formal H)

Edited by:

Matthew L. Bolton, SJSU Research Foundation/NASA Ames, USA

Asaf Degani, General Motors R&D, Israel

Philippe Palanque, ICS-IRIT, Université Toulouse III, France

Proceedings of a workshop sponsored by the

IFIP Working Group 13.5 on Human Error, Safety, and System Development

and held at the Imperial College in London

May 28, 2012

FMHFE.com
October 2012

Organizers:

Matthew L. Bolton, SJSU Research Foundation/NASA Ames, USA
Asaf Degani, General Motors R&D, Israel
Arnab Majumdar, Imperial College, London, U.K.
Philippe Palanque, ICS-IRIT, Université Toulouse III – France

Program Committee:

Ait Ameur, Yamine
Bass, Ellen
Bowen, Judy
Boy, Guy
Bredereke, Jan
Campos, José Creissac
Combéfis, Sébastien
Curzon, Paul
d'Ausbourg, Bruno
Feary, Michael
Feigh, Karen
Gellatly, Andrew W
Giannakopoulou, Dimitra
Harrison, Michael
Javaux, Denis
Johnson, Chris
Lütdke, Andreas
Oishi, Meeko
Paternò, Fabio
Pecheur, Charles
Ramesh, S.
Reeves, Steve
Ruksenas, Rimvydas
Rushby, John
Sampath, Prahladavaradan
Thimblebly, Harold
Trujillo, Maite
Tsimhoni, Omer

Table of Contents

Toward an Integrated Model Checking, Theorem Proving and Simulation Framework for Analyzing Authority and Autonomy

Ellen J. Bass[1] **Matthew L. Bolton**[2] **Karen M. Feigh**[3] **Elsa L. Gunter**[4] **John Rushby**[5]

ORIGIN AND UNDERLYING PRINCIPLES

In complex systems, human operators are responsible for a wide array of activities including monitoring the system during normal operations, making minor adjustments when operational requirements change, diagnosing problems when unusual situations arise, programming any associated automation, coordinating with team members and other collaborators, and taking over when abnormal situations and emergencies occur. In some domains, roles and responsibilities may shift between human and automation based on environmental situations, regulations, and procedures. New methods must be able to analyze concepts of operation for distributed autonomous and semi-automated systems including their human operators.

No single analysis framework can address the combinatorial explosion resulting from such system complexity. Agent-based simulation has shown promise toward modeling such complexity but requires a tradeoff between fidelity and the number of simulation runs that can be explored. Model checking can verify that the modeled system meets safety properties but they require that the components are of sufficiently limited scope. Thus leveraging these types of analysis methods synergistically can help to verify operational concepts that address the allocation of authority and autonomy.

To make the analyses using these techniques more efficient, we claim that common representations for model components, methods for identifying the appropriate safety properties, and techniques for determining the set of analyses to run are required. In addition, automated tools to create appropriate inputs and to interpret outputs are necessary. Methods to move between levels of abstraction and from one analysis technique to another are also required. Finally methods to ensure that the techniques are addressing their analysis goals are necessary.

Our work begins to address these needs (see Figure 1). By developing work, agent, environment and automation modeling languages, protection envelope-based methods to define and refine system safety properties, a simulation architecture, simulation trace analyses that ensure the simulation's design meets the intended analysis goals, abstraction methods that enable model checking analyses to provide useful information, and associated analysis support tools, our work focuses on verification methodologies and techniques that support human-automation interaction analysis.

MODELED RELATIONSHIPS

Declarative models with common representations of agents semantically describe the relationships and interactions between the system components. A simulation framework, WMC (for Work Models that Compute) models the complex, heterogeneous dynamics of systems that include physical systems, humans, and automated agents [12,13]. Human work is a response to the situation, with strategies chosen based on conditions in the physical environment; the allocation of responsibility within the team; and agent status – its expertise, the demands placed on it, and resources, such as available time and information. Actions are organized using an abstraction hierarchy [14,15]: at the bottom are the resources and actions and, at higher levels, more aggregate functions provide descriptions that relate the detailed actions to the specific goals of the work.

Enhanced Operator Function Model with Communication (EOFMC) is an XML-based language for describing task analytic models with human-human coordination and human-automation interaction [2,6]. Each human operator model is a set of task models that describe goal-level activities. Activities decompose into lower level activities and eventually atomic human actions. Decomposition operators specify the cardinality of and temporal relationship between the sub-activities or actions. EOFMC models teamwork as shared tasks: coordinated group activities undertaken by two or more human operators while allowing for human to human communication. The EOFMC language has formal semantics that specify how an instantiated model executes [2]. We have developed tools to translate instantiated EOFMs into formal models capable of being evaluated by

[1] Department of Systems and Information Engineering, University of Virginia, Charlottesville, VA USA, ejb4n@virginia.edu

[2] San José State University Research Foundation, NASA Ames Research Center, Moffett Field, CA USA, matthew.l.bolton@nasa.gov

[3] Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA USA, karen.feigh@gatech.edu

[4] Department of Computer Science, University of Illinois, Urbana – Champaign, Urbana, IL USA, egunter@cs.uiuc.edu

[5] Computer Science Laboratory, SRI International, Menlo Park, CA USA, rushby@csl.sri.com

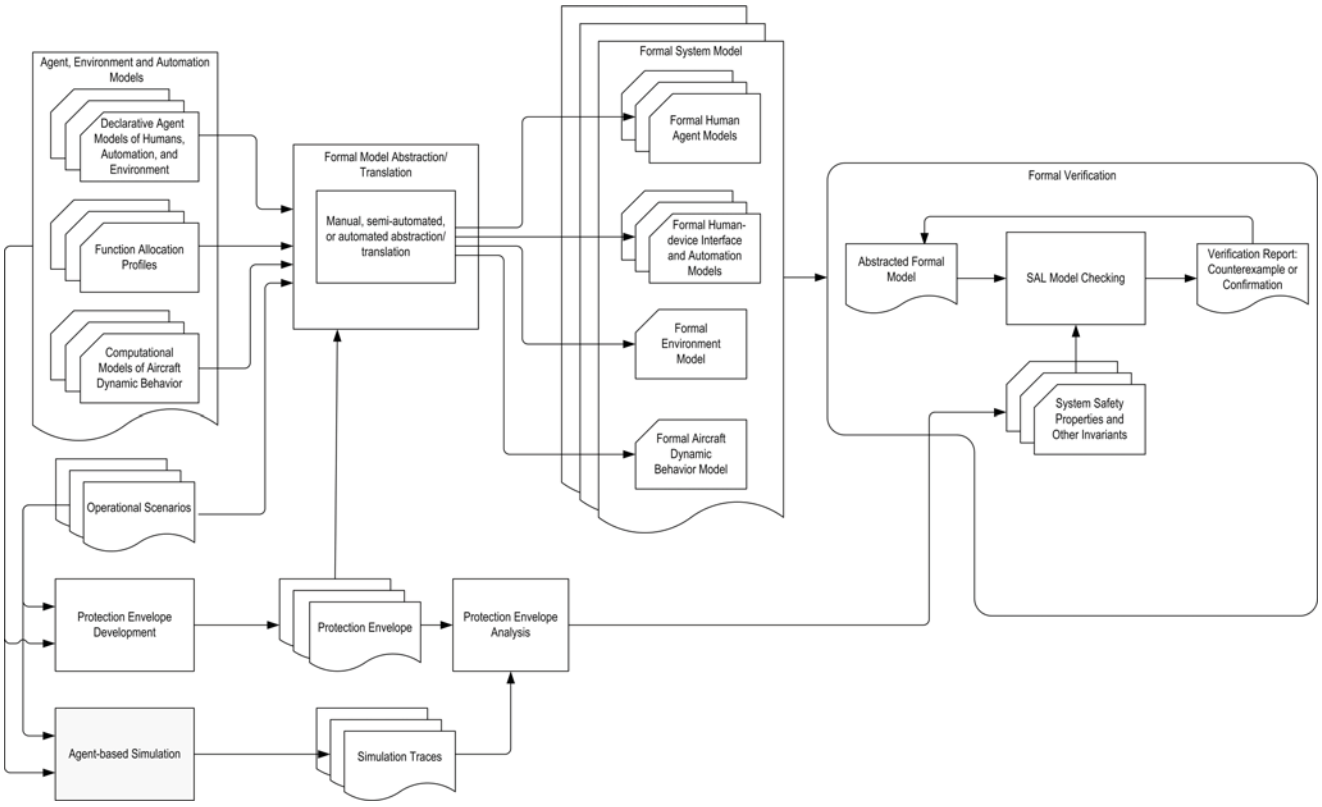


Figure 1. Using agent modeling, agent-based simulation, trance analysis, counterexample-guided abstraction refinement, and model checking to evaluate human-automation interaction.

the model checkers in the Symbolic Analysis Laboratory (SAL) [7] and the theorem prover Isabelle [10].

Human operator knowledge in EOFMC is embedded in task structures (including strategic knowledge specifying when activities can execute). To support model checking analyses with modeled human knowledge of the automation, we have modified the relational abstraction approach of [16] so that we can assert a relation as our model. This is sufficient for our purposes because our relations are conservative (i.e., admit more behaviors than would an accurate model). We construct very approximate models to begin with, then, if we discover an interestingly anomalous scenario (e.g., one in which the pilot's mental mode is "descend" but the airplane is climbing), we refine the model until the scenario becomes realistic or is found to dissolve as an artifact of excessive approximation. Once we have developed a realistic scenario with the model checker, we attempt to reproduce it in a high-fidelity simulation.

To identify and model safety properties, we use the protection envelope [8, 18-19] (Figure). Safe sequences are those in which the actions of the operator and system never lead to a domain-dependent concept o f loss. Sequences that are not safe are hazardous. Effective sequences are ones in which a domain-dependent concept of progress is accomplished. Among these are the recommended sequences in which the operator follows the steps in the task description. There may be ways to make progress that are not recommended. For example, the recommended procedures might describe one of many ways to meet the goal, where those not recommended ways might be hazardous. Warned sequences are not always hazardous; they are often aimed at making a sequence non-hazardous by enabling the system designer to make key assumptions about operator behavior. In protected sequences, the operator may vary from recommended or effective procedures without straying into hazardous territory. We envision this "protection envelope" as an engineered set of properties of the system that form a specified subset of safe behaviors—that is not safe by luck but rather safe by design.

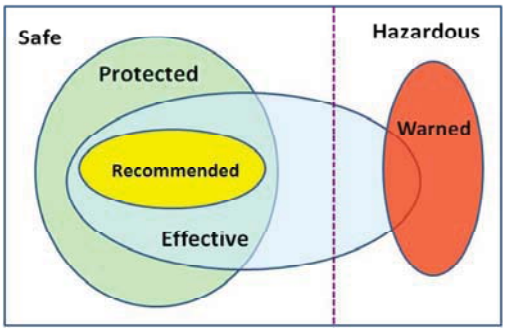


Figure 2. Protection envelope framework for identifying safety properties.

The protection envelopes can be succinctly specified by using logical properties. We use Linear Temporal Logic (LTL) [11] formulae for this purpose. LTL formulae are usually checked against finite state automata. However we need a model that can identify different entities, the actions performed by different entities in different situations and is able to capture the evolution of the system through the combined actions of the entities. The model offering all these characteristics is the Concurrent Game Structures (CGSs) [1], a type of automata that moves from state to state according to the actions of a set of agents. Specifying the protection envelope using LTL allows us to verify inclusion of a behavior in the protection envelope by simply checking whether the CGS corresponding to that behavior satisfies the protection envelope property [18].

With respect to simulation trace analysis, we can formally encode and analyze traces to assess safety and effectiveness requirement conformation [17]. Our work demonstrates that, with the help of faithful abstractions, we can obtain valuable insights about simulated traces from the formal verification procedures irrespective of the size of the simulation traces. The combination of simulation trace generation and formal verification provide feedback that may (i) assess the appropriateness of the requirement specifications, (ii) suggest possible infidelity in the simulation modules and (iii) delineate design errors in the original system.

PROBLEMS ADDRESSED

While some analyses are exhaustive with respect to possible human action choices, others focus on representative and/or well established patterns of human operator deviations from normative behavior.

APPLICATIONS

We are using Continuous Descent Arrival (CDA) scenarios to drive our work. A CDA procedure allows aircraft to continuously descend from high altitude directly into the ILS glide slope without any level flight segment at low altitude. Conventional approach procedures typically employ periods of constant altitude and speed. These constant segments simplify the air traffic control tasks for spacing and sequencing traffic by providing periods of well-defined vertical and speed behavior. CDA aims to eliminate the level altitude segments and their associated thrust transients at low altitude in order to keep the aircraft higher and at lower thrust prior to intercepting the ILS, thereby reducing noise exposure on the ground below.

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

The work described herein is part of on-going research. Not only do some of the methods require more development to become standalone tools, the integration of the methods into a coherent analysis framework requires more attention.

ACKNOWLEDGMENTS

This work was funded in part by NASA award NNA10DE79C and NSF Award 0917218. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NASA and NSF.

REFERENCES

1. Alur, R., Henzinger, T.A. & Kupferman, O. (2002). Alternating time temporal logic. *Journal of the ACM (JACM), 49(5)*, 672–713.
2. Bass, E.J., Bolton, M.L., Feigh, K.M., Griffith, D., Gunter, E., Mansky, W., & Rushby, J. (2011). Toward a multi-method approach to formalizing human-automation interaction and human-human communications. *2011 IEEE International Conference on Systems, Man, and Cybernetics*. October 9-12, 2011, Anchorage, Alaska, 1817-1824.
3. Bass, E.J., Feigh, K.M., Gunter, E. & Rushby, J. (2011). Formal modeling and analysis for interactive hybrid systems. *4th International Workshop on Formal Methods for Interactive Systems (FMIS)*, June 21, 2011, Limerick, Ireland.
4. Bolton, M.L. & Bass, E.J. (accepted). Using model checking to explore checklist-guided pilot behavior. *The International Journal of Aviation Psychology.*
5. Bolton, M.L. & Bass, E.J. (2011). Evaluating human-automation interaction using task analytic behavior models, strategic knowledge-based erroneous human behavior generation, and model checking. *2011 IEEE International Conference on Systems, Man, and Cybernetics*. October 9-12, 2011, Anchorage, Alaska, 1788-1794.
6. Bolton, M.L., Siminiceanu, R.I., & Bass, E.J. (2011). A systematic approach to model checking human-automation interaction using task-analytic models. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, 41(5),* 961-976.
7. De Moura, L., Owre, S., & Shankar, N. (2003). The SAL language manual. CSL Technical Report SRI-CSL-01-02 (Rev. 2). Menlo Park, CA: SRI International. http://sal.csl.sri.com./doc/language-report.pdf
8. Gunter, E. L., Yasmeen, A., Gunter, C. A., & Nguyen, A. (2009). Specifying and analyzing workflows for automated identification and data capture. In *Proceedings of the 42nd Hawaii international conference on system sciences* (pp. 1–11). Los Alamitos: IEEE Computer Society
9. Mansky, D. & Gunter, E. (in press). Using locales to define a rely-guarantee temporal logic. Accepted to *Interactive Theorem Proving (ITP) 2012*.
10. Paulson, L.C. (1986). Natural deduction as higher-order resolution, *Journal of Logic Programming, 3(3)*, 237-258.
11. Pnueli, A. (1977). The temporal logic of programs. *Proceedings of the 18th IEEE Symposium Foundations of Computer Science (FOCS 1977*), 46-57.
12. Pritchett, A. R., Feigh, K. M., Kim, S. Y., and Kannan, S. (under review). Work models that compute to support the design of multi-agent socio-technical systems.

Submitted to *IEEE Transactions on System Man and Cybernetics, Part A: Systems and Humans*.

13. Pritchett, A. R., Kim, S. Y., Kannan, S. & Feigh, K. M. (2011). Simulating situated work. *2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA),* Miami Beach, FL.
14. Rasmussen, J. (1979). On the structure of knowledge - A morphology of mental models in a man-machine system context. Risø-M-2192, 1979.
15. Rasmussen, J. (1983). On the structure of knowledge - A morphology of mental models in a man-machine system context. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-13(3),* 257-266.
16. Sankaranarayanan, S. & Tiwari, A. (2011). Relational abstractions for continuous and hybrid systems. In G. Gopalakrishnan and S. Qadeer (Eds.): Computer Aided Verification (CAV) Lecture Notes in Computer Science, *6806*, pp. 686–702.
17. Yasmeen, A.,Feigh, K.M., Gelman, G. & Gunter, E.L. (accepted). Formal analysis of safety-critical system simulations. *2nd International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS 2012)*.
18. Yasmeen, A. & Gunter, E. (2011). Automated framework for formal operator task analysis. *Proceedings of the 2011 International Symposium on Software Testing and Analysis(ISSTA '11)*. July 17th-21st, 2011, Toronto, ON, Canada, 78-88.
19. Yasmeen, A. & Gunter, E. (2011). Robustness for protection envelopes with respect to human task variation. *2011 IEEE International Conference on Systems, Man, and Cybernetics*. October 9-12, 2011, Anchorage, Alaska, 1809-1816.

Position Paper: Formal verification and the prevention of systematic user error

Paolo Masci, Rimvydas Rukšėnas, Huayi Huang, Paul Curzon, Michael Harrison
School of Electronic Engineering and Computer Science
Queen Mary University of London
{paolo.masci,rimvydas,huayih,paul.curzon,michael.harrison}@eecs.qmul.ac.uk

ORIGIN AND UNDERLYING PRINCIPLES

User error is often systematic. It arises as a result of poor human-computer system design, either of interactive devices or of the socio-technical system in which the devices and their users are embedded. For example, when a nurse enters the wrong rate into an intravenous infusion pump it might be the nurse's fault but it could also be poor pump design, poorly designed processes, wrong information from the pharmacy or wrong identification of the patient. The problem is to design resilient systems that prevent such systematic human error. Verification tools are required to support design, to provide evidence that design will reduce user error and harm, and to aid in the process of identifying the reasons why an incident occurred. We have explored a variety of approaches including: formal generic models of user behavior to analyze plausible user actions [22]; compliance of device interface behavior to interaction design principles [19, 20, 5, 11]; distributed cognition approaches to analyze information flows and the use of information resources in socio-technical systems [16, 17, 9]; systematic tool-based methods to help investigators identify systemic error conditions in the system during forensic investigations following adverse events [18]. Our approaches are built on top of two automated reasoning tools developed at SRI: the Symbolic Analysis Laboratory (SAL) [7] and the Prototype Verification Systems (PVS) [21].

Generic user model. Cognitive science knowledge is formulated as generic assumptions about plausible user actions. These assumptions are instantiated to both a device model and intended user activities. Our generic user model (GUM) takes this approach capturing generic assumptions relating to the salience of actions, the timing of actions and so on. This GUM is combined with a model of the device and the resulting system analyzed to detect possible erroneous paths where user goals are not achieved. Errors such as those arising from cognitive mismatches and performance issues have been explored for an intravenous infusion pump [26]. Errors arising from post-completion errors and other cognitive mistakes have been explored in relation to a fire engine dispatch system [22]. In both cases the results of the model have been compared with experimental data. The activity is carried out in close collaboration with cognitive psychologists.

Compliance to interaction design principles. Automated tools can help determine if interaction design principles have been consistently implemented in a device interface. An example of an interaction design principle we analyzed is *predictability* [19, 20]. Predictability concerns the ability of a user to determine the outcome of future interactions. It is an example of a formalizable design principle that has potential to be 'generative'. That is it can capture relevant human factors concerns in such a way that software engineers can implement systems that are effective relative to those concerns [28]. When an interface does not comply with a design principle, precise insights can be obtained for answering questions such as: (i) What design changes could be applied to make the design compliant? An answer to this question may provide useful insights to device manufacturers about the effect of different features in interaction design. (ii) Under what conditions does the design become compliant to the design principle? An answer to this question would provide insights for user training, in that we can check whether a reasonably simple strategy exists (other than resetting the device for example) that allows one to circumvent envisioned issues.

Information flows in socio-technical systems. Information resources deployed in the environment constrain the activities carried out by individuals. These constraints, when semantically correct and tight enough, facilitate the analysis of plausible user trajectories and provide insights about how to design artefacts and devices so as to make the path to achieving a task apparent. Hutchins was one of the first to articulate these concepts in his distributed cognition framework [12]. The basic idea of distributed cognition is that cognitive activities of individuals are not confined in individuals' heads, but span across artefacts and technologies. As such, cognitive activities are distributed throughout the whole system. Therefore, it is possible to reason about these activities by studying how information resources are generated, transformed and propagated within the system. We are carrying out this analysis in close collaboration with field investigators [16, 17].

Automated tools to support forensic investigations. Foren-

sic investigations following adverse events aim to identify the circumstances surrounding an incident. The ultimate aim of the investigation is to identify the latent factors that lead to an adverse event, and modify the system design so as to avoid the recurrence of similar events. In [18], we have explored a systematic tool-based approach that can help investigators to reason about systemic error conditions that caused an adverse event. The basic idea is that a systematic analysis of how available information resources (or the lack of them) may shape user action can help investigators focus on systemic causes rather than just on causal chains. The proposed approach uses a PVS higher order-logic model describing how information resources may have influenced the actions of those involved in the incident. Proof obligations generated by PVS are used to identify situations where available resources may afford unsafe user actions. This approach is not intended to replace existing accident analysis methods, such as the Australian Transport Safety Bureau investigation analysis framework [1], Ladkin's Why-Because analysis [14], or Leveson's STAMP [15]. Rather the aim is to further support the investigators' awareness about the circumstances surrounding an incident, enhancing the final recommendations.

MODELED RELATIONSHIPS

The approaches we are using can capture human factors concerns at two different levels: at a micro-level, the human-machine dyad is analyzed in detail to identify plausible user behaviors and verify interaction design principles; at a macro-level, the wider socio-technical system is considered, and the modeled relationships consider the way information resources afford user actions. This wider perspective on the system is used both in a proactive way, to identify possible hazards linked to the use of information resources, and in a retrospective way, to analyze the circumstances surrounding an incident.

Plausible user behavior. In the context of modeling user behavior a model of the device is related to a generic model that captures user behavior by describing plausible actions. Thus, a relation is defined between device actions and user actions which is not necessarily identical. Properties of the GUM relate to characteristics of user actions, for example salience and the extent to which different notions of salience have priority over each other. This becomes important when exploring sequences of actions that are likely to be taken by users to achieve their goals. An example is associated with slip errors [22]. Slip errors can have severe consequences in safety-critical contexts. Often opportunities for making such errors can be reduced by good design. Despite there being several cognitive theories which account for routine procedural action, no attempt had been made to develop a method that specifically highlights designs that allow users to make systematic slip errors. We addressed the problem by describing a series of formal modeling experiments that aimed to capture, in an abstract way, the cognitive aspects of an infusion pump programming task. Our GUM was used to conduct these model-based experiments. The aim was to provide several plausible formal accounts of our experimental findings. Our formal modeling experiments suggest that the Soft Constraints Hypothesis [10] is an appropriate explanation for why people select different programming strategies, leading to a good match between model predictions and experimental results. We also looked at the relation between the planning and reactive aspects in producing plausible user behaviors [25].

Interaction design principles. For the analysis of interaction design principles, in [19, 20] we have explored the possibility of verifying *predictability* of the interactive data entry system of commercial drug infusion pumps. Predictability is an interaction design principle that concerns whether a, possibly expert, user can determine the effect of an action on the basis of the persistent observable state of the device (e.g., what is shown on the device displays). In [5], we shifted the analysis from the interactive data entry system to the wider functionalities provided by the drug infusion pump, and verified interaction properties such as mode clarity and consistency of actions. Related work that directly links with ours includes: Degani and Heymann's work [8], which describes a systematic approach for evaluating whether a device interface provides the necessary information for allowing operators to perform specified tasks correctly and reliably; Rushby's work on mode confusion [27, 3], where model checking approaches are used for comparing plausible mental models developed by users and the actual implementation of the system; Bolton and Bass' work [4], where SAL [7] is used to verify whether normative tasks are properly supported by device interface functionalities.

Information resources and user actions. When extending the analysis to the wider socio-technical system, we perform a systematic analysis of information flows as they happen in actual practice (e.g., according to what field investigators observe in contextual studies) and as described in normative behavior (e.g., according to written protocols and user manuals). The aim of the analysis is to help domain experts and field investigators to identify situations where the flow of information may afford wrong or unsafe user actions. In [16, 17], we demonstrated that a pragmatic and relatively simple use of the PVS [21] theorem proving system can support field investigators studying socio-technical systems by helping them to uncover latent situations linked to potential hazards due to the observed use of information resources. This analysis complements classical task-based analysis, such as that described in [4, 2], in that the focus is on how information resources are transformed and propagated within the system rather than on the sequence of activities carried out by individuals.

Retrospective analysis of resource constraints. In [18], we explored the possibility of developing a systematic tool-based method that raises questions about the circumstances surrounding an adverse event. The approach offers a practical and systematic way to apply a distributed cognition perspective to incident investigations, focusing on how available information resources (or the lack of them) may shape user action, rather than just on causal chains. This perspective sup-

ports a deeper understanding of the more systemic causes of incidents. The analysis is based on a higher order-logic model describing how information resources may have influenced the actions of those involved in the incident. The PVS theorem proving system is used to identify situations where available resources may afford unsafe user actions. The method is illustrated by re-analyzing some aspects of an accident involving a drug infusion pump [13]. The method found issues beyond that related to direct causes of the particular incident, as well as insight related to other issues that could lead to future mishaps. It is necessary however to carry out more case studies to further explore the benefits of our approach.

PROBLEMS ADDRESSED

Our analysis considers multiple perspectives (from single human-automation interactions to the wider socio-technical system) and integrates formal methods and empirical studies with the aim to obtain a richer analysis of the interactive behavior of a system. We believe that multiple methods and empirical techniques are needed to analyze interactive systems, rather than a single modeling language and environment, as each approach can highlight a different aspect of the system. From the case studies analyzed to date, we have some evidence that formal methods and empirical studies are not alternative approaches for studying interactive systems, but instead they complement and refine each other. Others have also pointed out this concept in the past, see for instance [29].

APPLICATIONS

We used the generic user model to uncover errors and performance issues arising from post-completion steps and other cognitive mistakes in relation to an ATM cash point [23], infusion pumps [26] and a fire engine dispatch system [22]. We used the approach also for identifying potential security problems in authentication interfaces [24]. We applied the analysis of interaction design principles to detailed specifications of commercial drug infusion pumps [19, 20, 5, 11]. In [9], we applied a resource-based analysis to discover potential problems in the interface of a control process system model. This kind analysis has also been used in combination with field study data to study an emergency medical dispatch system [16, 17]. In [18], we re-analyzed some aspects of a medical incident described in a comprehensive investigation report involving a drug infusion pump [13].

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

The approach to user modeling is still in its early stages, particularly in relation to the possibility of using this as a methodology for development. Several issues are currently being addressed to solve this problem.

1. Provide a way of assimilating appropriate cognitive principles into the GUM so that the resulting GUM is comprehensible and acceptable to psychologists giving confidence that appropriate assumptions have been made. By this means the models predictions can be compared with their experimental results.
2. Provide a framework to support the development of device models that capture properties of the device to be used experimentally at an appropriate level of abstraction.
3. Deal with the problems of scale in developing these models combining model checking and theorem proving technology.

The use of automated reasoning tools to support existing semi-structured methodologies also needs further development, particularly in relation to effective methods for feeding back proof obligations to non-experts in formal methods. In the case studies considered to date, the expressiveness of the PVS specification language makes it possible to overcome several pre-conceived ideas of field researchers about the possible limitations of translating informal descriptions into mathematical specifications. Also, the PVSio extension for animating specifications is allowing us to engage with field investigators, in a limited way, when checking the correctness of the specification. The packaging of automated reasoning tools is a major barrier when engaging with non-experts in formal methods. We are exploring ways to mitigate this by developing ad hoc GUIs, such as that of the IVY tool [6], that allow one to explore simulation traces or generate them interactively through simple push button style interfaces.

ACKNOWLEDGMENTS

Funded as part of the CHI+MED: Multidisciplinary Computer-Human Interaction research for the design and safe use of interactive medical devices project, EPSRC Grant Number EP/G059063/1, and Extreme Reasoning, Grant Number EP/F02309X/1.

REFERENCES

1. Australian Transport Safety Bureau. *Analysis, causality and proof in safety investigations*. 2007. ATSB transport safety research report, AR-2007-053.
2. Bass, E., Bolton, M., Feigh, K., Griffith, D., Gunter, E., Mansky, W., and Rushby, J. Toward a multi-method approach to formalizing human-automation interaction and human-human communications. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on* (oct. 2011), 1817–1824.
3. Bass, E. J., Feigh, K. M., Gunter, E. L., and Rushby, J. M. Formal modeling and analysis for interactive hybrid systems. *ECEASST 45* (2011).
4. Bolton, M. L., and Bass, E. J. Formally verifying human—automation interaction as part of a system model: limitations and tradeoffs. *Innovations in Systems and Software Engineering 6*, 3 (Sept. 2010), 219–231.
5. Campos, J., and Harrison, M. Modelling and analysing the interactive behaviour of an infusion pump. *ECEASST 45* (2011).
6. Campos, J. C., and Harrison, M. D. Interaction engineering using the IVY tool. In *Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems*, EICS '09, ACM (New York, NY, USA, 2009), 35–44.
7. de Moura, L., Owre, S., Rueß, H., Rushby, J., Shankar, N., Sorea, M., and Tiwari, A. SAL 2. In *Computer Aided*

Verification: CAV 2004, R. Alur and D. A. Peled, Eds., vol. 3114 of *Lecture Notes in Computer Science*, Springer-Verlag (July 2004), 496–500.

8. Degani, A., and Heymann, M. Formal verification of human-automation interaction. *Human Factors 44*, 1 (2002), 28–43.

9. Doherty, G., Campos, J., and Harrison, M. *Interactive Systems. Design, Specification, and Verification*. Springer-Verlag, Berlin, Heidelberg, 2008, ch. Resources for Situated Actions, 194–207.

10. Gray, W. D., Sims, C. R., Schoelles, M. J., and Fu, W. The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. *Psychological Review 113* (2006), 461–482.

11. Harrison, M., Campos, J., and Masci, P. Reusing models and properties in the analysis of similar interactive devices. *Submitted for publication to Innovations in Systems and Software Engineering (2012). Preprint available at* `http://tinyurl.com/masci-QMpreprints`.

12. Hutchins, E. *Cognition in the Wild*, new ed. The MIT Press, September 1995.

13. ISMP Canada. Fluorouracil incident root cause analysis report. `http://www.ismp-canada.org/download/reports/FluorouracilIncidentMay2007.pdf`.

14. Ladkin, P., Sieker, B., and Sanders, J. *Safety of Computer-Based Systems*. Springer-Verlag, Heidelberg and London. Draft version from July 27, 2011.

15. Leveson, N. A new accident model for engineering safer systems. *Safety Science* (2004), 237–270.

16. Masci, P., Curzon, P., Blandford, A., and Furniss, D. Modelling distributed cognition systems in PVS. *ECEASST 45* (2011).

17. Masci, P., Curzon, P., Furniss, D., and Blandford, A. Using PVS to support the analysis of distributed cognition systems. *Submitted for publication to Innovations in Systems and Software Engineering (2012). Preprint available at* `http://tinyurl.com/masci-QMpreprints`.

18. Masci, P., Huang, H., Curzon, P., and Harrison, M. D. Using pvs to investigate incidents through the lens of distributed cognition. In *NASA Formal Methods*, A. Goodloe and S. Person, Eds., vol. 7226 of *Lecture Notes in Computer Science*, Springer (2012), 273–278.

19. Masci, P., Rukšėnas, R., Oladimeji, P., Cauchi, A., Gimblett, A., Li, Y., Curzon, P., and Thimbleby, H. On formalising interactive number entry on infusion pumps. *ECEASST 45* (2011).

20. Masci, P., Rukšėnas, R., Oladimeji, P., Cauchi, A., Gimblett, A., Li, Y., Curzon, P., and Thimbleby, H. The benefits of formalising design guidelines: A case study on the predictability of drug infusion pumps. *Submitted for publication to Innovations in Systems and Software Engineering (2012). Preprint available at* `http://tinyurl.com/masci-QMpreprints`.

21. Owre, S., Rajan, S., Rushby, J., Shankar, N., and Srivas, M. PVS: combining specification, proof checking, and model checking. In *Computer-Aided Verification, CAV '96*, R. Alur and T. A. Henzinger, Eds., no. 1102 in Lecture Notes in Computer Science, Springer-Verlag (New Brunswick, NJ, July/August 1996), 411–414.

22. Rukšėnas, R., Back, J., Curzon, P., and Blandford, A. Verification-guided modelling of salience and cognitive load. *Formal Aspects of Computing 21*, 6 (Nov. 2009), 541–569.

23. Rukšėnas, R., Curzon, P., Back, J., and Blandford, A. Formal modelling of cognitive interpretation. In *Proceedings of the 13th international conference on Interactive systems: Design, specification, and verification*, DSVIS'06, Springer-Verlag (Berlin, Heidelberg, 2007), 123–136.

24. Rukšėnas, R., Curzon, P., and Blandford, A. Modelling and analysing cognitive causes of security breaches. *Innovations in Systems and Software Engineering 4*, 2 (2008), 143–160.

25. Rukšėnas, R., Curzon, P., and Blandford, A. Modelling rational user behaviour as games between an Angel and a Demon. In *Proceedings of Software Engineering and Formal Methods*, IEEE Comp. Society Press (2008), 355–364.

26. Rukšėnas, R., Curzon, P., Blandford, A., and Back, J. Combining human error verification and timing analysis: A case study on an infusion pump. *Submitted for publication to Formal Aspects of Computing.*

27. Rushby, J. Using model checking to help discover mode confusions and other automation surprises. *Reliability Engineering and System Safety 75*, 2 (Feb. 2002), 167–177. Available at `http://www.csl.sri.com/users/rushby/abstracts/ress02`.

28. Thimbleby, H. In *Human-Computer Interaction — INTERACT'84*, B. Shackel, Ed., North-Holland (1985), 661–666.

29. Wright, P., Fields, B., and Merriam, N. *From formal models to empirical evaluation and back again*. Formal methods in human-computer interaction. Berlin, Springer, 1997, ch. 13, 283–314.

Automatic Generation of Full-Control System Abstraction for Human-Machine Interaction

Sébastien Combéfis
Computer Science and Engineering Dept.
Université catholique de Louvain
Place Sainte Barbe, 2
1348 Louvain-la-Neuve, Belgium
Sebastien.Combefis@uclouvain.be

Charles Pecheur
Computer Science and Engineering Dept.
Université catholique de Louvain
Place Sainte Barbe, 2
1348 Louvain-la-Neuve, Belgium
Charles.Pecheur@uclouvain.be

ORIGIN AND UNDERLYING PRINCIPLES

Automated systems are increasingly complex, making it hard to design interfaces for human operators. Human-machine interaction (HMI) errors like automation surprises are more likely to appear and lead to system failures or accidents as testified by several cases detailed in the literature [9, 13, 16]. Researchers in psychology, human factors and ergonomics have been working on HMI issues for several years. Since the mid-1980s, researchers are investigating the use of formal methods to analyse behavioural aspects of HMI. Initially focused on the analysis of specific situations and on the system and it properties [17, 3], the field moved to more generic results based on theories like graph theory, model-checking or theorem proving [19, 2, 8].

Different questions might be asked in the analysis of HMI. The first kind of problem is the *verification* of some properties such as: *"May a system exhibit potential mode confusion for its operator?"* or *"No matter in which state the machine is, can the operator always drive the machine into some recover state?"*. Another kind of problem is the *generation* of some elements that help in a correct interaction, such as user's manuals, procedures and recovery sequences or user interfaces.

Recently, Degani et al. [11] pioneered a new approach consisting in the automatic generation of a user mental model for a system model described as a statechart. In this context, a mental model is not meant to capture a human cognitive model; rather, it is meant to capture the implicit and intended model of operation according to which the system developer designs the system.

The work we are pursuing follows the work of Degani et al. by defining formally the problem of automatic generation of a user mental model satisfying properties which allows a perfect user who follows that model exactly to operate the system without being surprised during the interaction. The definition of these properties, which we call *full-control* [7], and the development of corresponding verification and generation algorithms, is the core of our work.

This paper describes the proposed approach to formally analyse HMI. The remainder of the paper is organised as follows. The first section draws up the motivation of this work and poses the context and the problem that is tackled. The second section presents our techniques to generate automatically system abstractions. The next section presents briefly the prototype tool that has been developed. The fourth section discusses the ongoing work and perspectives for the proposed approach and finally the conclusion sums up our contribution.

MODELLED RELATIONSHIPS

Automatic generation of mental models needs to be driven by the intended characteristics of the resulting models. The *full-control* property [7] formalizes the following notion of a correct mental model: a user following a full-control mental model will know at any point how to command or observe the system to achieve a goal, based on the history of previous commands and observations performed. The models are formally represented as enriched labelled transition systems (LTS) where a distinction is made between the actions [12]. *Commands* are executed by the user on the system (inputs) and *observations* are controlled by the system and just observed by the operator (outputs). *Internal actions* are purely internal to the system, not observable by the user at all. All those aspects are detailed in [7]. In more recent work, we are also considering additional state-based observation and we show that those new enriched models can be translated into the initial framework. Figure 1 shows the graphical representation of a system model of a vehicle transmission system example coming from [11].

Generating a minimal full-control mental model from a given system model helps to get a better understanding of the system. The full-control property captures the knowledge an operator needs to have about a system to be able to control it properly. Such a mental model can be used to build training materials such as user manuals [18]. Providing a system that the user can learn, minimizing her memory load, and allowing her to operate it without error is a desirable usability property [15].

PROBLEMS ADDRESSED

This work proposes the full-control property to highlight an aspect of a good system abstraction which will ensure good

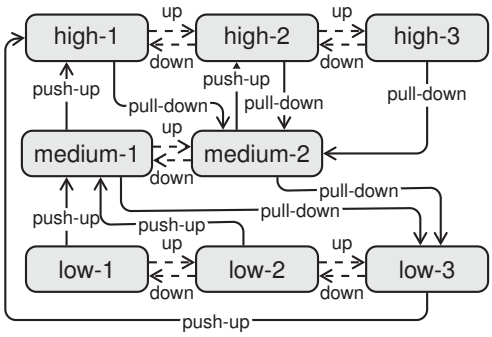


Figure 1. The vehicle transmission system example.

human-machine interaction. Algorithms to check that property and to generate minimal full-control system abstractions have been developed, based on a reduction approach [7] and also on a learning approach [5]. Also, an analysis methodology and an associated framework for using such algorithms in a practical setting to support the design and analysis of HMI systems are proposed in [4, 6]. The proposed framework can be used for modelling HMI systems and analyzing models against HMI vulnerabilities. The analysis can be used for validation purposes or for generating artifacts such as mental models, manuals and recovery procedures; it can also be used to help redesign or update a system model to avoid detected vulnerabilities.

The core contribution of this work is the automatic generation of minimal full-control system abstraction given a system model. As introduced in the previous section, the full-control property ensures that a user following a user mental model satisfying the property will always keep control of the system and furthermore will be able to execute all the possible interactions with the system. That is, if at any time during the interaction, a command can be executed on the system, the user will know it. Moreover if an observation occurs, the user will not be surprised as he will expect it according to his user mental model.

Two algorithms were developed in [7, 5], which are focused on the automatic generation of a minimal full-control mental model for a given system. The first is based on the definition of a bisimulation-based relation between the states of the system, stating which of them can be merged together because they can be handled similarly from the standpoint of the operator. The second uses a learning algorithm which iteratively builds mental model guesses. The algorithm relies on a teacher to answer whether proposed execution sequences must, may or cannot be part of the mental model. The teacher uses the system model to answer such queries.

Figure 2 shows the minimal full-control system abstraction for the vehicle transmission system example. As illustrated, the operator does not need to know the difference between the three high states of the system, and between the two medium states. For the low states, the operator must distinct the three states in order to be able to control the system (according to the full-control property). In practice, it means that the operator must pay attention to the up and down observations in order to control the system.

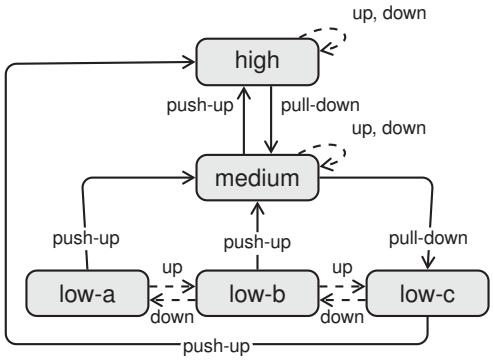


Figure 2. The minimal full-control mental model for the vehicle transmission system.

APPLICATIONS

A prototype tool has been implemented in Java. The prototype is based on the *Java Pathfinder* (JPF) model-checker [1]. This section briefly presents the tool.

The first part of the framework aims at providing tools for encoding models using statecharts [10], a widespread graphical notation to model systems. The statecharts can be designed in any existing tool which supports export in XMI file, e.g. ArgoUML [1]. The statechart is converted into a Java program encoding it, following the conventions of the JPF statecharts extension [14]. With that extension, the resulting Java program can be explored and used by the JPF model checker, for example to check temporal logic properties. Finally, the framework uses JPF with the JPF statecharts extension to explore all the transitions of the complete behaviour of the system and builds the full expanded LTS.

The second part of the framework consists of the analysis and mental model generation part. It is possible to check whether a mental model allows full-control of a given system. The tool takes two LTSs as input (a system and a mental model) and outputs true if the mental model allows full-control of the system and false otherwise. It is also possible to generate a minimal full-control mental model given a system model as input (provided such a model exists). Both algorithms from [7, 5] (reduction and learning) can be used. The tool produces an LTS corresponding to one minimal full-control mental model or says that no such model exists providing a problematic sequence from the system.

The benefit of using JPF is that it is a versatile model checker. It can therefore be used to perform additional types of analysis on the statechart model, for example application-specific safety properties as supported by the JPF framework.

The generation of minimal full-control system abstraction can be used in several phases of the design process. During the design of the system, the approach can be used to control if the system could be controlled through the existence of a minimal full-control abstraction. The system abstraction can also reveal clues about the system complexity as a mental model

[1] `http://argouml.tigris.org/`.

should not be ideally too large to fit in the human memory. One application of the proposed techniques is to help in the design phase so that system are designed in a way to ensure the possibility for an operator to control it without being surprised during the interaction.

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

Most system models used in the literature includes state-based observations. In the current framework, labelled transition systems are used which means that all the information is on the transitions. We are currently working on more general models where there is also information on the states. The first results tend to prove that this new problem can be translated in the current setting.

Full-control property may be too strong for some kind of analyses since it forces all the commands that are available on the system to be present in the user mental model. In some particular situations, what is interesting for the operator is to be able to only control a certain subset of the full behaviour of the system. We are currently exploring a variant of the full-control property where the user is not required to know exactly all the possible commands of the system but only those who are of interest to the user, for example described in a user task model.

This work describes a formal framework for the analysis of human-machine interactions, with a focus on controllability aspects of the system based on a distinction between commands and observations. The analysis is based on a formal characterization of an adequate control of the system by the user. That characterization, captured by the full-control property, is used as a validation criterion for system models during the design process cycle. The full-control property is a desirable property since it helps to prevent the operator from being surprised when interacting with a system. Two algorithms, one based on a reduction approach and one based on a learning approach, have been proposed. The framework has been implemented in Java within the JPF model checker environment.

REFERENCES

1. JavaPathfinder (JPF). http://babelfish.arc.nasa.gov/trac/jpf/.
2. Campos, J. C., and Harrison, M. D. Model checking interactor specifications. *Automated Software Engineering 8*, 3–4 (2001), 275–310.
3. Campos, J. C., and Harrison, M. D. Systematic analysis of control panel interfaces using formal tools. In *Proceedings of the 15th International Workshop on the Design, Verification and Specification of Interactive Systems*, no. 5136 in Lecture Notes in Computer Science, Springer-Verlag (July 2008), 72–85.
4. Combéfis, S., Giannakopoulou, D., Pecheur, C., and Feary, M. A formal framework for design and analysis of human-machine interaction. In *Proceedings of the 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2011)*, IEEE (Oct. 2011), 1801–1808.
5. Combéfis, S., Giannakopoulou, D., Pecheur, C., and Feary, M. Learning system abstractions for human operators. In *Proceedings of the 2011 International Workshop on Machine Learning Technologies in Software Engineering (MALETS 2011)*, ACM (New York, NY, USA, Nov. 2011), 3–10.
6. Combéfis, S., Giannakopoulou, D., Pecheur, C., and Mehlitz, P. A JavaPathfinder extension to analyse human-machine interactions. In *Proceedings of the Java Pathfinder Workshop 2011* (Nov. 2011).
7. Combéfis, S., and Pecheur, C. A bisimulation-based approach to the analysis of human-computer interaction. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2009)*, G. Calvary, T. N. Graham, and P. Gray, Eds., ACM (New York, NY, USA, July 2009), 101–110.
8. Curzon, P., Rukšėnas, R., and Blandford, A. An approach to formal verification of human-computer interaction. *Formal Aspects of Computing 19*, 4 (Nov. 2007), 513–550.
9. Degani, A. *Taming HAL: Designing Interfaces Beyond 2001*. Palgrave Macmillan, Jan. 2004.
10. Harel, D. Statecharts: A visual formalism for complex systems. *Science of Computer Programming 8* (June 1987), 231–274.
11. Heymann, M., and Degani, A. Formal analysis and automatic generation of user interfaces: Approach, methodology, and an algorithm. *Human Factors: The Journal of the Human Factors and Ergonomics Society 49*, 2 (Apr. 2007), 311–330.
12. Javaux, D. A method for predicting errors when interacting with finite state systems. How implicit learning shapes the user's knowledge of a system. *Reliability Engineering and System Safety 75* (Feb. 2002), 147–165.
13. Leveson, N. G., and Turner, C. S. Investigation of the therac-25 accidents. *IEEE Computer 26*, 7 (July 1993), 18–41.
14. Mehlitz, P. C. Trust your model - verifying aerospace system models with JavaPathfinder. In *Aerospace Conference, 2008 IEEE* (Mar. 2008), 1–11.
15. Nielsen, J. The usability engineering life cycle. *Computer 25* (Mar. 1992), 12–22.
16. Palmer, E. Oops, it didn't arm. — a case study of two automation surprises. In *Proceedings of the 8th International Symposium on Aviation Psychology* (1996), 227–232.
17. Rushby, J. Using model checking to help discover mode confusions and other automation surprises. *Reliability Engineering and System Safety 75*, 2 (Feb. 2002), 167–177.
18. Thimbleby, H. Creating user manuals for using in collaborative design. In *Proceedings of the Conference Companion on Human Factors in Computing Systems*, ACM (New York, NY, USA, 1996), 279–280.
19. Thimbleby, H., and Gow, J. Applying graph theory to interaction design. In *Engineering Interactive Systems 2007/DSVIS 2007*, J. Gulliksen, Ed., no. 4940 in Lecture Notes in Computer Science, Springer-Verlag (2008), 501–518.

Hierarchical Model of Human Guidance Performance based on Interaction Patterns in Behavior

Bérénice Mettler and Zhaodan Kong
Interactive Guidance and Control Lab ($i^{\mathcal{G}}_{\mathcal{C}}$Lab)
Department of Aerospace Engineering and Mechanics, University of Minnesota
Minneapolis, MN 55455, USA
mettler@umn.edu, kong@aem.umn.edu

ABSTRACT

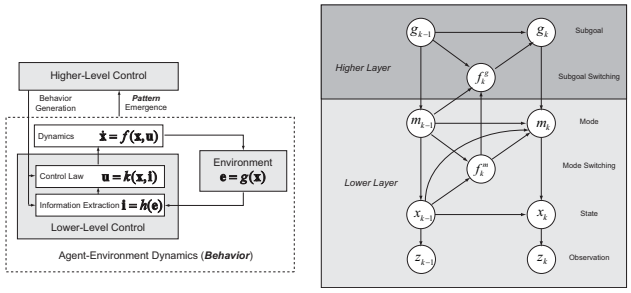
This paper describes a framework for the investigation and modeling of human spatial guidance behavior in complex environments. The model is derived from the concept of interaction patterns, which represent the invariances or symmetries inherent in the interactions between an agent and its environment. These patterns provide the basic elements needed for the formalization of spatial behavior and determine a natural hierarchy that can be unified under a hierarchical hidden Markov model.

ORIGIN AND UNDERLYING PRINCIPLES

Spatial behavior has been an active research topic in psychology and robotics over the past few decades. What fascinates researchers is the ability of trained humans to spontaneously generate behavior for problems that are often not tractable from a computational standpoint [1]. Take driving a car for instance, it involves a driver, a car (the driver and the car together can be taken as the agent), as well as the surrounding environment. All three have their own dynamics. The driver needs not only to comprehend the dynamics of each single component, but also needs to have a holistic understanding of the dynamics of the entire agent-environment system.

Furthermore, these types of problems generally involve processes that obey quite different principles. Sensing and perception are often considered to be probabilistic, while cognition and action are considered to be deterministic [2, 3]. A driver or pilot has to integrate all these processes while factoring in the overall goal of the task, e.g., driving to a specified location safely and as fast as possible.

Theories regarding the organization of behavior can be categorized into two main schools: model-based approaches and non-representational approaches. Most non-representational approaches like tau coupling [4], or more recently models based on information processing and dynamical principles [5], provide useful explanations of the perception-action loop in behavior. However, the behavior is almost always formulated in terms of some low-dimensional dynamics without specific meaning. Their main limitations is that they cannot explain complex behaviors involving the composition of several behaviors, such as those that result from more complex environments with a remote goal state.



(a) (b)

Figure 1. (a) A conceptual pattern-based hierarchical guidance model. (b) Hierarchical hidden Markov model of guidance behavior, comprising two layers with the interaction pattern serving at the link between the two.

Model-based approaches [6, 7], on the other hand, tend to focus either on the perception or the action side; they are rarely presented under a unified framework. If they are, they are most often based on the generic "sense-model-plan-act" model, which due to its rigidity makes it challenging to explain the flexible and adaptive capabilities of human spatial behavior.

This short paper only highlights the key concepts and results. For a comprehensive treatment of the concepts that we introduce, as well as the details regarding the experiments, the algorithms and the results, please refer to [8, 9, 10].

MODELED RELATIONSHIPS

Considering the range of complexities involved in the agent-environment dynamics and the perception-cognition-action processes, one of the fundamental question that need to be addressed in the study of spatial behavior is what aspects of these dynamics are fundamental in explaining how spatial behavior is organized. Our modeling framework is built on the analysis of the agent-environment dynamics as illustrated in Fig. 1(a) and focuses on the understanding and characterizing the emerging interaction patterns, and how these can then

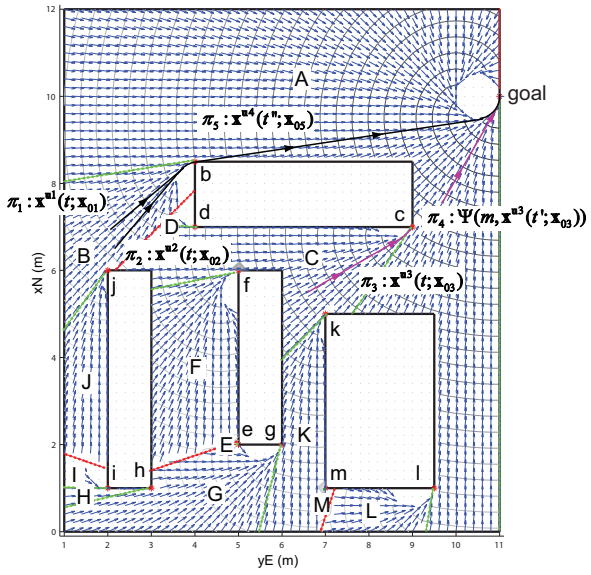


Figure 2. Optimal guidance behavior of Dubins' vehicle for an environment with multiple obstacles.

help formalize the analysis of behavior and the development of a model like the HHMM shown in Fig.1(b).

Agent-Environment Dynamics and Interaction Patterns

The dynamics of an agent can be described as:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{1}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ is the agent state, $\mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control law. In order for an agent to perform a guidance task, it should be able to perceive the world via $\mathbf{i}(t) = h(\mathbf{e}(t))$, where $\mathbf{i}(t)$ is the information and $\mathbf{e} \in \mathcal{E}$ is the environment state, which can be represented as $\mathbf{e}(t) = g(\mathbf{x}(t))$, and choose an appropriate action $\mathbf{u}(t)$ to bring itself from an initial state $\mathbf{x}_0$ to a goal state $\mathbf{x}_f$. The process of choosing the right action can be written in the form of a control policy as $\mathbf{u}(t) = k(\mathbf{x}(t), \mathbf{i}(t))$. Putting all these components together results in the following closed-loop agent-environment dynamics:

$$\dot{\mathbf{x}} = f(\mathbf{x}, k(\mathbf{x}, h(g(\mathbf{x})))) \tag{2}$$

The collection of all the agent state trajectory, $\{\mathbf{x}(t) : t \in [t_0, t_f]\}$, together with the corresponding environment state trajectory, $\{\mathbf{e}(t) : t \in [t_0, t_f]\}$, with respect to this closed-loop dynamics is our formal definition of *guidance behavior* [9].

We can introduce two types of relationships over guidance behavior. One relationship $\sim_g$ is defined by extending the concept of motion primitive [11]: two trajectories, $\overleftarrow{s_i}^L$ and $\overleftarrow{s_j}^L$[1], satisfy $\overleftarrow{s_i}^L \sim_g \overleftarrow{s_j}^L$ if there exist a $m \in M$ and control histories, $\overleftarrow{u}_i^L$ and $\overleftarrow{u}_j^L$ such that:

$$(\Psi(m, \overleftarrow{s}_i^L), \overleftarrow{u}_i^L) = (\overleftarrow{s}_j^L, \overleftarrow{u}_j^L) \tag{3}$$

where $L \in \mathbb{Z}^+$, M is some finite-dimensional Lie group, and Ψ is the action of this group M on the state manifold $\mathcal{X}$, $\Psi := M \times \mathcal{X} \to \mathcal{X}$. The following two conditions need to be held for all $m \in M$, $\mathbf{x} \in \mathcal{X}$, $\mathbf{e} \in \mathcal{E}$, $t \in [t_0, t_f]$ and all $u \in \mathcal{U}$ in order to satisfies (3):

$$\Psi(m, \mathbf{x}^{\mathbf{u}}(t; \mathbf{x}_0)) = \mathbf{x}^{\mathbf{u}}(t; \Psi(m, \mathbf{x}_0)) \tag{4}$$

and

$$\Psi|\mathcal{E}(m, \mathbf{e}^{\mathbf{u}}(t; \mathbf{e}_0)) = \mathbf{e}^{\mathbf{u}}(t; \Psi|\mathcal{E}(m, \mathbf{e}_0)) \tag{5}$$

where $\Psi|\mathcal{E}$ is the restriction of mapping Ψ in $\mathcal{E}$ and it can be well defined on the assumption that the environment state can be written in the form of some relative quantities, $\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_r$. (4) implies that if $t \to (\mathbf{x}(t), \mathbf{u}(t))$ is an integral curve of (1), so is $t \to (\Psi(m, \mathbf{x}(t)), \mathbf{u}(t))$. (5) can be interpreted similarly.

The other relationship $\sim_s$ is defined using the concept of causal state [12]:

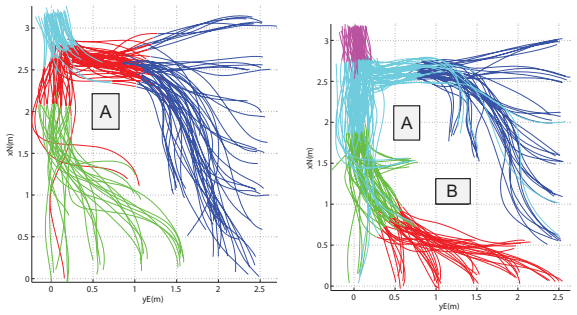
$$\overleftarrow{s_i}^K \sim_s \overleftarrow{s_j}^L \Leftrightarrow P(\overrightarrow{S}|\overleftarrow{s_i}^K) = P(\overrightarrow{S}|\overleftarrow{s_j}^L) \tag{6}$$

for all semi-infinite $\overrightarrow{S} = s_0 s_1 ...$, where $K, L \in \mathbb{Z}^+$ and P stands for probability. Since, in this paper, we are only concerned with deterministic systems, we can then assign P equal to one. In this setting, the state s_0 is then called a *subgoal*, in the sense that, from this state on, the two trajectories, $\overleftarrow{s_i}^K$ and $\overleftarrow{s_j}^L$, will follow the same trajectories $\overrightarrow{S}$. We will drop the length variables K and L here and denote the members of any length in the set $\overleftarrow{S}$ by $\overleftarrow{s}$.

It can be easily verified that both $\sim_g$ and $\sim_s$ are equivalence relationships. Thus, if $\overleftarrow{s} \in \overleftarrow{S}$, one type of equivalence class over $\overleftarrow{S}$ can be defined in the following two steps (the order of these two operations is interchangeable): $[\overleftarrow{s}] = \{\overleftarrow{s}' \in \overleftarrow{S} : \overleftarrow{s}' \sim_s \overleftarrow{s}\}$ and $[[\overleftarrow{s}]] = \{[\overleftarrow{s}]' \in [\overleftarrow{s}] : [\overleftarrow{s}]' \sim_g [\overleftarrow{s}]\}$. Each equivalence class $[[\overleftarrow{s}]]$ is called an *interaction pattern* to reflect the fact it captures invariances inherent in the agent-environment interactions.

For the guidance behavior of Dubins' vehicle as shown in Fig. 2, the agent state and the environment state are invariant with respect to a translation and a rotation about a vertical axis, or to the actions of the symmetry group $M = SE(2)$. Each element of M can be written in the form of a 3×3 matrix $m(\psi, \mathbf{t})$, with rotation angle ψ and the translation vector $\mathbf{t} = [t_x, t_y]'$. For the example trajectories shown in Fig. 2, according to (3) and (6), we have $\pi_1 \sim_s \pi_2$ and $\pi_3 \sim_g \pi_4$. And taking the $\sim_s$ equivalence (e.g., equating the two black trajectories) results in the partitions of the state space (encircled by obstacle boundaries, red and green lines, which correspond

[1]For computational convenience, both the $\sim_g$ and $\sim_s$ relationships are defined on the symbolic representations of guidance behavior instead of the continuous form (2), which can be obtained through a quantization, $q : \mathcal{X} \times \mathcal{E} \to \mathcal{A}$, where $\mathcal{A}$ is a finite set of symbols which is called the state alphabet. With such a quantization, the set of all trajectories can then be written as: $\overleftarrow{S} = \{\overleftarrow{s}_i^L : s_{i-L+1}, ..., s_i, s \in \mathcal{A}, L \in \mathbb{Z}^+, i \in \mathbb{Z}\}$. The controls can be quantized similarly.



(a) (b)

Figure 3. Trajectory segmentation results for two different tasks.

to repelling and attracting manifolds, respectively [10]), and subsequently taking the $\sim_g$ equivalence (e.g., equating the two purple trajectories) results in two interaction patterns: one corresponds to approaching the subgoals from the left, such as guidance behaviors in partition A, and the other corresponds to approaching the subgoals from the right, such as guidance behaviors in partition B. Actually, after a mirror reflection symmetry is added to group M, only a single interaction pattern is left.

Experimental Investigation and Validation

Next we proceeded to investigate the agent-environment dynamics and validate the concept of interaction pattern using experimental data. The approach together with the necessary computational tools from machine learning, system identification and pattern recognition are summarized in the following five steps (I-V).

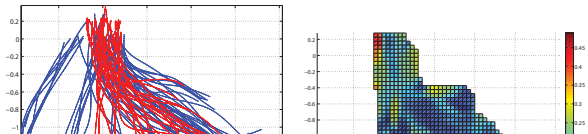
Experimental trajectory data was collected from agile guidance tasks performed with a miniature remote control helicopter [10] (Fig. 3) in our interactive guidance and control lab [13]. The data was represented as $\mathbf{x}^n(i), i = 1, ...N_n, n = 1, ..., N$ with N as the number of trajectories and N_n as the number of data point for trajectory n. The helicopter planar rigid-body motion is fully characterized by four variables $\mathbf{x}^n(i) := [x^n(i), y^n(i), v^n(i), \psi^n(i)]'$, where $[x^n(i), y^n(i)]'$ is the position, $v^n(i)$ is the speed and $\psi^n(i)$ is the course angle.

(I) Symbolic representation and subgoal identification: Transformation of the trajectory data into a symbolic representation and identifying pairwise subgoals based on the definition (6). The transformation is done by quantizing the state space into mutually exclusive cells according to q. Each cell is a letter of the state alphabet $\mathcal{A}$. If a data point $\mathbf{x}^n(i)$ falls within a cell, it is represented by the corresponding letter $s^n(i)$. Once the transformation is done for each data point, the original measurement data $\mathbf{x}^n(i), i = 1, ...N_n, n = 1, ..., N$ is transformed into its corresponding symbolic form as $s^n(i), i = 1, ...N_n, n = 1, ..., N$.

(II) Subgoal clustering and trajectory segmentation: The extracted subgoals are clustered applying Isomap, multidimensional scaling and K-means methods. The experimental trajectories are then clustered into $\sim_s$ equivalent segment clusters based on their membership subgoal.[2] The original trajectory sample data are augmented by their memberships as follows:

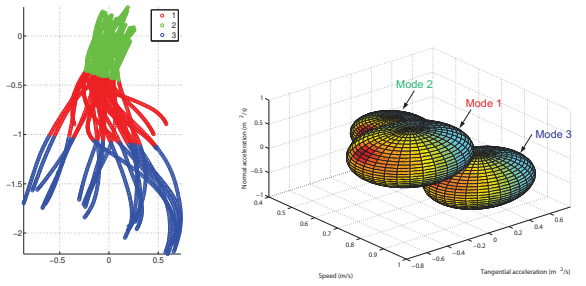
$$[\mathbf{x}^m(i)', \xi^m(i)]', i = 1, ...M_m, m = 1, ..., M$$

with M as the number of trajectory segments, M_m as the number of data point for trajectory segment m, and $\xi^m(i)$ as the membership of data point $\mathbf{x}^m(i)$ with $1 \leq \xi^m(i) \leq K^*$. The segmentation results for two tasks are shown in Fig. 3.



(a) (b)

Figure 4. Matching results. (a) Superimposed trajectory segments. (b) The relative difference between correspondence points.



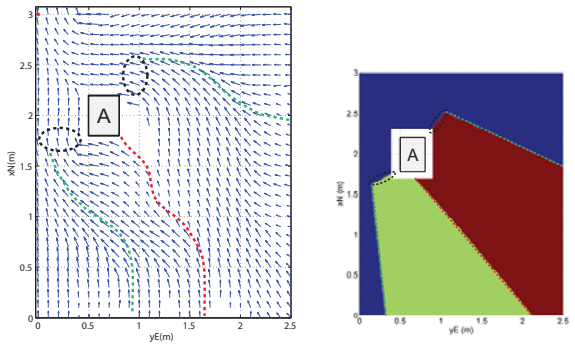
(a) (b)

Figure 5. (a) Data points according to their mode memberships. (b) shows a typical fitted distribution of velocity, tangential and normal accelerations for a segment cluster.

(III) $\sim_g$ equivalence analysis: To prove that one segment cluster ξ_1 is symmetric to another cluster ξ_2 according to $\sim_g$ equivalence definition (3) or they both belong to a same interaction pattern, it suffices to show that for any trajectory $\mathbf{x}^u(t; \mathbf{x}_0)$ from cluster ξ_1, there exists an action m from the symmetry group M and a trajectory $\mathbf{x}^{u'}(t'; \mathbf{x}'_0)$ from cluster ξ_2 such that $\mathbf{x}^{u\prime}(t'; \mathbf{x}'_0) = \Psi(m, \mathbf{x}^u(t; \mathbf{x}_0))$ holds and all the m's are the same, and vice versa. This evaluation was formulated as a parameter optimization problem. The matching results for two of the clusters are shown in Fig. 4.

(IV) Analysis of dynamical characteristics: The dynamical characteristics of guidance behavior in each segment clusters (as sample interaction patterns) is analyzed using piecewise affine (PWA) model [15]. A PWA system is defined by the

[2]The application of our clustering operation is based on the assumption that the "observed" subgoals extracted from step (I) are the expression of some "hidden" subgoals, where the number of hidden subgoals is much smaller than the observed ones. The distribution of the observed subgoals can be modeled by a mixture of Gaussians as follows [14].



(a) (b)

Figure 6. (a) Vector field computed from experimental data. (b) Predicted partition. In both of these two figures, black lines mark the locations of subgoals, red line marks the location of repelling manifold, and green lines mark the locations of attracting manifolds.

following state-space equations:

$$\mathbf{x}(t+1) = A_i\mathbf{x}(t) + B_i\mathbf{u}(t) + d_i, \text{ for } \mathbf{x} \in \mathcal{X}_i, \mathbf{u} \in \mathcal{U}_i \quad (7)$$

where $\{\mathcal{X}_i\}_{i=1}^{l_m}$ and $\{\mathcal{U}_i\}_{i=1}^{l_m}$ are polyhedral partitions of $\mathcal{X}$ and $\mathcal{U}$ (each partition can be called a mode), and d_i is the noise term. As shown in Fig. 5, for each segment cluster, three modes with distinguished characteristics can be identified. They are a starting mode (mode 3) m^s, a coasting mode (mode 1) m^c, and an approaching mode (mode 2) m^a.

(V) Meta-behavioral analysis: The transition among the segment clusters and their spatial boundaries are investigated based on general dynamical systems concepts. The transition boundaries among the patterns can be approximately characterized as attracting and repelling manifolds [9, 10]. The functional form of time-to-go (TTG) function is first learned from experimental data and then a wavefront method based on optimality principles can be used to derive the partitions: the subgoals are determined as the locations where the wavefront defined by the learned TTG function meet the vertices of obstacles; the repelling manifolds correspond to the locations where two wavefronts originating from either a goal or a subgoal meet; and the attracting manifolds correspond to subgoals and their directions are determined by the gradient of the wavefront. Fig. 6 shows the predicted partitions that result from this method compared to the original partitions.

Integration under a Hierarchical Hidden Markov Model

In summary, steps I and II correspond to taking the $\sim_s$ equivalence. Each extracted segment cluster can be seen as a sample interaction pattern. In step III, the $\sim_g$ equivalence of these segment clusters is evaluated. The results from the first three steps confirm that interaction patterns do exist in human guidance behavior and that they can be explained using equivalence concepts. Following, in step IV and V, the micro and macro organizational principles within and across these patterns are investigated. Here the results show that the transitions between modes within and across the patterns can be described through simple rules.

The analysis of guidance behavior based on interaction patterns suggests that guidance behavior follows a natural and systematic hierarchical organization. The overall system can be formalized using a hierarchical hidden Markov model (HHMM) as shown in Fig. 1(b). For the example in this paper, state $\mathbf{x}$ is taken as $[x, y, v, \psi]'$ and the measurement is taken to be the same as $\mathbf{x}$. Mode m can take on three values: m^s, m^c and m^e. The edges or dependencies among m and x at different times, along with the Boolean mode switching node f^m, are learned in step IV. Together, they correspond to the PWA systems learned from the interaction patterns. Similarly, the edges among subgoals g and the Boolean goal switching mode f^g can also be learned from experimental data in step V.

PROBLEM ADDRESSED

Our method based on an analysis of the guidance behavior in terms of agent-environment dynamics enabled to identify that the keystone in the organization of spatial behavior represents the invariances inherent to guidance behavior. These were described as interaction patterns and then used to formalize the guidance behavior under a hierarchical HHMM.

Similar efforts of building formal models to study human behavior can also be found in vision [3] and motor control [6]. Our framework distinguishes itself by encompassing the entire perception-cognition-action loop. Furthermore, compared to some non-representational frameworks [4, 5], thanks to the hybrid nature of our model, our framework can be easily extended and generalized to investigate more complex scenarios and behaviors.

Our framework also provides an avenue for understanding the organizational mechanisms humans and animals may utilize in order to reduce the burden of planning as well as to enable flexible and adaptive behavior. Our model suggests that high-level planning can be performed using an interaction pattern library, which can be understood as the repertoire of guidance capability that accounts for the agent-environment interactions. The cardinality of this library is much smaller than that of the entire state space. The results also show that explicit details of the agent's dynamics are not necessary for planning; it is how those dynamics manifest in the interaction patterns that really matter.

The HHMM model shows that once a composition of interaction patterns has been elaborated, the pilot must primarily monitor whether the subgoal corresponding to the currently employed interaction pattern is attained and whether the interaction pattern remains valid. As long as the goal is not attained, the same subgoal and information extraction law $h(.)$ and control law $k(.,.)$ are applied. Once it is, a new subgoal, information extraction law $h(.)$ and control law $k(.,.)$ are initiated.

Finally, it is important to underscore, that our framework relies largely on a data-driven approach. Assumptions regarding the nature and mechanisms underlying guidance behavior are kept to a minimum; the knowledge used to build the key elements of our model is almost entirely derived from the invariances that exist in the interactions between the agent and

the environment. The details about the functional form of our interaction patterns, the number of them, the laws dictating the transitions between one pattern to the next could in principle all be learned from experimental data.

APPLICATIONS

The HHMM model provides both descriptive and predictive capacities. This makes it useful for a range of engineering and scientific applications. Being able to predict the pilot's behavior and performance based on the environment, task and mission elements is relevant to a number of applications. The model could be used as part of an active cueing system. Predicting behavior allows to identify potential failure states and then alerting the pilot and/or switching control modality.

The hierarchic model delineates the relevant functions and levels of representation. This knowledge can be used to determine the different modalities of interactions available to an operator and will help determine the design specifications for a broader range of human-machine control modalities.

Another application for our framework is the development of novel planning and control algorithms for autonomous systems. For instance, one ongoing challenge in robotics is the brittleness of robot's performance [2]. The gained knowledge of the principles that dictate the organization of spatial behavior will support our understanding of the adaptive guidance capabilities and in turn help design algorithms needed to operate in less structured and partially known environments.

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

Due to the limited number of scenarios, only the functional form of the time-to-go are learned while the organizational rule is assumed to follow some optimality principle. Although the prediction of the locations of subgoals and boundaries are reasonable, data from a broader range of experiments are needed to learn the high-level transition laws from experimental data.

In terms of development opportunities, the modeling framework provides a new way to study perceptual and control mechanisms. The organization of behavior based on the interaction patterns must be intimately linked to the perceptual mechanisms. In fact these patterns are the manifestation of the perception-action loop. We are currently conducting experiments with eye tracking device determine relationships between attention patterns and behavior. These experiments will help us account for the specific perceptual mechanisms in the agent-environment model.

Following the same vein, the functional description provided by the model makes it possible to understand what potential measurements can be used to investigate operator workload and attention. Brain imaging and brain activity analysis is still often treating the brain as a black box. Our hierarchic model provides a more precise picture of the type of activation levels (control, perceptual, planning) expected as a function of the stage in the task.

REFERENCES

1. B. Mettler, "Structure and organizational principles of agile behavior: Challenges and opportunities in cognitive engineering," *Journal of Cognitive Critique*, vol. 3, 2011.
2. M. Campbell, M. Egerstedt, J. How, and R. Murray, "Autonomous driving in urban environments: approaches, lessons and challenges," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4649–4672, 2010.
3. D. Mumford, "Pattern theory: the mathematics of perception," *Arxiv preprint math/0212400*, 2002.
4. D. Lee, "Guiding movement by coupling taus," *Ecological Psychology*, vol. 10, no. 3-4, pp. 221–250, 1998.
5. W. Warren, "The dynamics of perception and action." *Psychological Review*, vol. 113, no. 2, p. 358, 2006.
6. D. Wolpert and Z. Ghahramani, "Computational principles of movement neuroscience," *Nature Neuroscience*, vol. 3, pp. 1212–1217, 2000.
7. E. Todorov, "Optimality principles in sensorimotor control," *Nature Neuroscience*, vol. 7, no. 9, pp. 907–915, 2004.
8. Z. Kong and B. Mettler, "Guidance primitives and interaction patterns: foundations for formal modeling of human guidance behavior," *IEEE Transactions on Robotics*, Submitted.
9. ——, "Foundations of formal language for humans and artificial systems based on intrinsic structure in spatial behavior," in *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, 2011.
10. ——, "An investigation of spatial behavior in agile guidance tasks," in *Proceedings of the 2011 IEEE International Conference on Systems, Man, and Cybernetics*, Anchorage, AK, 2011.
11. E. Frazzoli, M. Dahleh, and E. Feron, "Maneuver-based motion planning for nonlinear systems with symmetries," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1077–1091, 2005.
12. C. Shalizi and J. Crutchfield, "Computational mechanics: Pattern and prediction, structure and simplicity," *Journal of Statistical Physics*, vol. 104, no. 3, pp. 817–879, 2001.
13. B. Mettler, N. Dadkhah, Z. Kong, and J. Andersh, "Hardware and software environment for human interactive and autonomous guidance research," in *2012 International Conference on Unmanned Aircraft Systems (ICUAS'12)*, 2012.
14. C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.
15. G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari, "A clustering technique for the identification of piecewise affine systems," *Automatica*, vol. 39, no. 2, pp. 205–217, 2003.

A Longitudinal Systemic Framework for Identifying the Organizational Precursors to Flawed Human Automation Interaction in Safety Critical Domains

Simone Rozzi
Interaction Design Center
School of Information and Engineering Science
Middlesex University
Hendon NW4 4BT
London, UK
simone.rozzi@gmail.com

INTRODUCTION

This paper introduces an early framework intended to support the retrospective analysis of the inter-organizational and organizational precursors to safety critical automation. Notably, beyond its intended benefits, automation brings a set of undesired effects or anomalies on practitioners' performances. The framework is helpful to understand how and why organizations may end up deploying automated systems presenting such anomalies, and how they can possibly became able to control them. Consistent with the area of organizational resilience [1-3], the framework aims at helping organizations to monitor their blind spots and update their risk models on safety critical automation. This can help them to avoid a drift into poor development and implementation. The knowledge generated by the framework is intended to be exploited by managers and policy makers for identifying potential joint blind spots in the organization(s) they oversight, and for facilitating corrective actions in relation to automation deployment programs and policies.

PROBLEM ADDRESSED

Notably, automation deployment in safety critical systems has been plagued with a set of undesired effects on practitioners' performances [see e.g. 4; 5-7]. Anomalies or flaws on the human automation interface have been documented over a variety of industries, and include, for instance, the ineffective redistribution of practitioners' workload, the presentation of nuisance or ambiguous alerts, error-inducing displays, information prone to unintended and undesirable uses. Besides disrupting operational practices, such automation related anomalies are also critical as they might introduce new types of complex system failure [see e.g. 8].

Despite the availability of user centered philosophies and methods, organizations deploying safety critical automation are still exposed to the risk of deploying problematic human automation interaction. Hence, besides process–based (mechanistic) explanations of automation failure (e.g. 'lack of user involvement', 'poor consideration of end user requirements', 'lack of training, 'designers error' [see e.g. 9; 10; 11], it is important for organizations to develop broader holistic explanations of how and why they might end up deploying and operating suboptimal automated tools. This might sensitize them over their joint blind spots and areas of biased decision-making influencing the development, adoption, implementation, and operation of novel technology. At the same time, this is not an easy task because much theorizing on organizational failure has focused on the social and organizational precursors to accidents and disasters, but not on automation or Human Machine Interaction (HMI) failure specifically.

PROPOSED FRAMEWORK

The framework focuses on the qualitative modeling of the (inter) organizational response to automation anomalies related to a given automated tool. By (inter) organizational response it is meant the way an automation anomaly gets identified, interpreted, debated, transmitted across the diverse hierarchical levels and organizational boundaries of the social system influencing automation development, adoption, and implementation, in a given industry. Notably, such social system can be quit complex, including end users, designers, human factors and safety experts, international standard developers, regulators (national and international), unions, software vendors, etc. Each of these organizational actors might influence the design of the tool, as well as policies, regulations and standards related to it.

The response might be optimal, in that these organizational actors have the commitment, structure, resources, and expertise to "detect" a given anomaly, classify it as a relevant safety, HMI or human factors issue(s), and ultimately implement and monitor the related remedial action(s). Alternatively, the response might be suboptimal, in that the same organizations are unable to control a given anomaly. Examples from literature on disasters show that a safety critical anomaly may remain undetected or ambiguous for years despite being noted, or, if known, might get accepted as normal under existing models of risks, such as for the Challenger O-ring erosion and the Columbia foam strikes [12].

Ultimately, such definition of organizational response builds on socio-technical models of safety risk management [3; 13], literature on technological innovation systems [14; 15], and works on sociological analysis of anomalies and threats evolution within organizations [16-18].

In order to produce an explanation of the 'organizational response' to automation–anomalies, it is necessary to focus on the historical evolution of an automated tool and its anomalies within the social system adopting that tool. Such historical–organizational focus is needed for understanding the insiders' rationale behind decisions and events that, albeit having influenced design, adoption, implementation, and operational aspects apparently well, might have inadvertently induced the occurrence of 'uncontrolled' automation anomalies on the tool in use. Overall, this retrospective investigative process is characterized by a 'dual focus':

1. On the one hand the viewpoints of the operators at front end of the work system, i.e. the intended users of automation, such as air traffic controllers or pilots. Their view is needed to understand the actual role of the tool in use, the possible conflicts with the operational practice, the potential for unintended use, and so on.
2. On the other hand, the viewpoints of the other stakeholders at the blunt end, i.e. supervisors, managers, R&D directors, national and international regulators, policy makers [19]. These latter views are needed to understand, for instance, the interests, the rationales, the cultural frames, and the organizational objectives behind the decisions to develop, adopt, and implement the tool.

MODELED RELATIONSHIPS

Within the social system engaged in the adoption of a given automated tool, the framework allows analysts and policy makers to consider the following dynamics:

1. How knowledge and expertise (i) are developed by or (b) have flown into the organizations adopting a novel piece of automation. The development of the capabilities to set up a novel piece of automation is a challenging task and depends on the specific innovation strategy that can be enforced and sustained within a given organization. For instance, within the European Air Traffic Management (ATM) system, a large service provider might be pioneering automation deployment having internal in-house resources needed to perform R&D concerning the development and parameterization of a specific tool. When such resources are lacking, other (smaller) adopters might opt for an imitation strategy, with a focus on integrating the expertise needed to set up the alarm that is available somewhere outside of the company. Expertise acquisition might be realized, for instance, through personnel poaching, participation in networks of expertise, cross-transfer of personnel, assistance from external consultants. Not accounting for how the knowledge and expertise needed to set up an automated tool can be generated and diffused across the organizations operating in a given industry, leaves room for both problematic local implementations, e.g. automated alarm plagued with a high rate of false alarms [20], and asymmetrical performances at system wide level.

2. Integration between (a) software vendor and (b) service provider. Effective integration, between a software vendor selling safety critical automation and the organization purchasing and deploying it, is a key component for ensuring the efficient implementation of an automated tool. While a successful integration requires an optimal blend of software development/parameterization expertise (vendor) and operational expertise (client), problems might arise due to incomplete contracting, poor alignment, presence of hidden costs, lack of experience with the specific system on the client side, the vendor, or both. For instance implementation might turn out to be more challenging than expected, while the vendor willingness to accommodate client's requests decreases as the project reaches the contractual end [20]. Or again, the novel piece of automation might be out of the scope of existing regulations, so that the manufacturer might opportunistically adapt some components from a previous client site, which however may poorly fit with the actual client's operational context. Noteworthy, quality assurance practices (e.g. development and maintenance of safety and/or human factors case, adoption of user centered design processes) might receive inadequate attention by the software vendor and the client organization if not mandatory under existing legislation. Ultimately, not only an organization may end up with a suboptimal implementation, but existing contractual obligations might even prevent quick recovery actions. So, one regulatory implication following the analysis of software vendor-service providers integration in a safety critical domain could consist in tracing software vendor performance and in defining vendors black lists similar to those already contemplated in civil aviation.

3. Balance between (a) incentives for adoption and (b) availability of control mechanism. This aspect can be observed both at international and national levels. For instance, incentives and target objectives mandating or promoting the adoption of novel automation, as defined for instance by international modernization programmes (e.g. SESAR in Air Traffic Management), should be balanced by the presence of proper regulations and guidance material. In one of our cases, concerning the European ATM system, we detected that this might not be the case and that initial efforts to promote the widespread adoption of an alarm system across the ANSPs of different countries might not be matched by the existence of appropriate standards and regulations. In turn, this opened the way to asymmetric implementations across different countries. Analogous

dynamics might occur at local company level: the company adopting the tool might be oriented towards the acquisition of the latest and most sophisticated technology available on the market, for instance because of cultural reasons (this is the choice rewarded by senior management) or political reasons (there is a need for the company to claim to have an efficient infrastructure in place), ultimately paying little attention to the operational needs of end users.

4. Structure of HMI and Safety Debates. Ideally, in an optimal organizational learning situation, HMI and safety issues are openly debated by organization members. The organization should promote open confrontation and revision of the assumptions and worldviews held by different organization members about (ambiguous) issues raised by the introduction and operation of novel automation.

However, this might not be the case. For instance, during the introduction of a novel automated tool, there might be a tendency for staff (i) to de-rate some relevant HMI and safety issues to avoid holding up the project; (ii) to delay addressing some issues under the assumption that these might be solved through future training and procedures, when this may not be the case in fact; (iii) to ignore valuable safety ideas because no one has time to listen, as this might produce additional delays. [21.] In these cases, effective confrontation on safety issues is mostly inhibited by activities oriented toward short-term productivity and commercial objectives, with the end result of leaving potential issues yet to be addressed when the final automated will enter operation.

Furthermore, debates might be plagued by a tendency to keep premises out of scrutiny and to solve lack of consensus by mean of power differences. Such tendencies are known to inhibit organizational learning. Not explicating the premises behind the courses of action intended by different parties hampers confrontation, ultimately leaving opposing parties trapped in conflicting positions [22]. Solving lack of consensus through the use of informal or formal power relationships means that the dominant party will ultimately judge what is the most appropriate interpretive frame to adopt over an ambiguous issue [23]. In our analysis of the Safety Recommendations issued by the National Transportation Safety Board to the Federal Aviation Administration and targeting a ground alarm system (the MSAW) [25], we observed the two mentioned tendencies to contribute to keep one safety debate between the two agencies focused over an issue—alarm design improvement—that was offset from the central problem—the frequent exposure to nuisance alerts. This misalignment lasted for nearly three decades.

APPLICATIONS

To date, the proposed framework has been retrospectively applied to in two in-depth investigations focused on (a) identifying the inter-organizational and organizational precursors to problethe European Air Traffic Management System [20; 24] and on (b) identifying the inter-organizational sources of inaction over problematic alarm implementation in the US Air Traffic Management System [25].

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

At the present stage the framework is grounded on two case studies only, therefore we plan to refine it through subsequent case studies, within and beyond ATM. This strategy is expected to improve the reliability of the framework and refine the modeled relationships. Also, we plan to involve policy makers and planners of automation initiatives to evaluate the potential and usefulness of the framework as a reflective tool in the support of automation programmes and policies.

It should be noted that one factor limiting the improvement of the framework is the scarcity of data on automation failure in safety critical domains. While theorizing on organizational disasters can rely extensively on official investigation reports, produced by accident investigation bodies, the same cannot be said in relation to unsuccessful or failed automation implementations. The official investigation of this latter class of events does not appear to be an institutionalized practice yet, especially when no accident or near miss has unveiled a criticality with the automated system in question. Hence, events of automation failure live mainly on the narratives ad memories of insiders, who in turn might be reluctant to openly discuss failure occurred within their organizations, unless this is very distant in time.

ACKNOWLEDGMENT

The author wish to thank Dr Barry Kirwan and Mr Andy Kilner, EUROCONTROL Experimental Center, for having promoted the positive initiation of this work and for their useful guidance during data collection. Also, the authors wish to express their gratitude to Ben Bakker, EUROCONTROL Headquarter, for having facilitated access to the Safety Net Task Force community (SPIN), and all of the SPIN members, in particular those directly involved in data collection and site visits. This work is part of a PhD programme sponsored in part by EUROCONTROL Experimental Center, Brétigny-sur-Orge, France. The views expressed herein do not necessarily reflect the official view or policy of the agency.

REFERENCES

[1] Woods, D. D. and Hollnagel, E. Prologue: Resilience engineering concepts. In *Resilience engineering. Concepts and precepts*, E. Hollnagel, D. D. Woods, and N. Leveson (Eds), Ashgate, 2006.

[2] Hollnagel, E., Nemeth, C. P., and Dekker, S. *Resilience engineering perspectives: remaining sensitive to the possibility of failure*. Ashgate Pub Co, 2008.

[3] Leveson, N. G. *Engineering a Safer World. System Thinking Applied to Safety*. On line publication: http://sunnyday.mit.edu/safer-world/safer-world.pdf, 2011.

[4] Woods, D. D. The alarm problem and directed attention in dynamic fault management. *Ergonomics* 38(11) (1995), 2371–2393.

[5] Sarter, N. B. and Woods, D. D. Automation surprises. In *Handbook of Human Factors and Ergonomics*, G. Salvendy (Ed), Wiley, 1997 Jan 1.

[6] Pritchett, A. R. Reviewing the role of cockpit alerting systems. *Human Factors and Aerospace Safety* 1(1) (2001), 5–38.

[7] Perry, S. J., Wears, R. L., and Cook, R. I. The role of automation in complex system failures. *Journal of Patient Safety* 1(1) (2005), 56–61.

[8] Perrow, C. *Normal Accidents. Living with High- Risk Technologies*. Princeton University Press, Princeton, New Jersey, 1999.

[9] Cardosi, K. Human factor lessons learned in the design and implementation of air traffic control systems. *The Controller, 11-15, first quarter* (1998), 11–15.

[10] Leveson, N. G. *System safety engineering: Back to the future*. On line publication: http://sunnyday.mit.edu/book2.pdf, 2002 Jan 1.

[11] Kinnersley, S. and Roelen, A. The contribution of design to accidents. *Safety Science* 45(1-2) (2007), 31–60.

[12] Vaughan, D. System Effects: On Slippery Slopes, Repeating Negative Patterns, and Learning from Mistake? In *Organizations at the Limit: Lessons from the Columbia Disaster*, W., A., Starbuck and M. Farjoun (Eds), Blackwell Publishing, Ltd, 2005 Jan 1.

[13] Rasmussen, J. Risk management in a dynamic society: a modeling problem. *Safety science* 27(2–3) (1997), 183–213.

[14] Sung, T. K. and Carlsson, B. The evolution of a technological system: the case of CNC machine tools in Korea. *Journal of Evolutionary Economics* 13(4) (2003), 435–460.

[15] Carlsson, B., Jacobsson, S., Holmén, M., and Rickne, A. Innovation systems: analytical and methodological issues. *Research policy* 31(2) (2002), 233–245.

[16] Vaughan, D. Organizational rituals of risk and error. In *Organizational encounters with risk*, B. Hutter and M. Power (Eds), Cambridge University Press, 2005 Jan 1.

[17] Edmondson, A. C., Roberto, M. A., Bohmer, R. M. J., Ferlins, E. M., and Feldman, L. R. The Recovery Window: Organizational Learning Following Ambiguous Threats. In *Organization at the Limit: lessons from the Columbia disaster*, W. H. Starbuck and M. Farjoun (Eds), Blackwell Publishing, 2005.

[18] Regester, M. and Larkin, J. *Risk issues and crisis management in public relations: A casebook of best practice*. Kogan Page Ltd, 2008.

[19] Rozzi, S., Amaldi, P., and Kirwan, B. Identifying how automation can lose its intended benefit along the development process: A research plan, in *Proceedings of the 9th bi-annial international conference on Naturalistic Decision Making* (London, UK, 2009).

[20] Rozzi, S., Amaldi, P., and Kirwan, B. IT innovation and its organizational conditions in safety critical domains: The case of the Minimum Safe Altitude Warning system, in *Proceedings of the 5th IET International Conference on System Safety* (Manchester, UK, October 2010), IET, 1-7.

[21] Humphreys, P., Kirwan, B., and Ternov, S. A Safe Approach to Transition: Key Elements for Transition Success. EUROCONTROL technical document n. 405 (October 2006).

[22] Argyris, C. and Schön, D. A. *Organizational Learning II: Theory, Method, and Practice*. Addison-Wesley Publishing Company, Reading Mass, 1996.

[23] Milliken, F. J., Lant, L., and Bridewell-Mitchell, E. Barriers to the Interpretation and Diffusion of Information about Potential Problems in Organizations: Lessons from the Space Shuttle Columbia. In W. A. Starbuck and M. Farjoun (Eds), Blackwell Publishing, 2005.

[24] Rozzi, S. and Amaldi, P. Organizational and Interorganizational Precursors to Problematic Automation in Safety Critical Domains. A Longitudinal Ethnographic Case Study from the Air Traffic Management Domain, in *Proceedings of the ATACCS 12* (London, Uk, May 2012).

[25] Amaldi, P. and Rozzi, S. Inter-Organizational Safety Debate: The Case of an Alarm System from the Air Traffic Control Domain. *International Journal of Sociotechnology and Knowledge development* 4(1) (2012), 30–47.

A formal language for next generation cockpits user interfaces specification

Vincent Lecrubier
ONERA/DTIM
Toulouse
vincent.lecrubier@onera.fr

Bruno d'Ausbourg
ONERA/DTIM
Toulouse
ausbourg@onera.fr

Yamine Aït-Ameur
ENSEEIHT
Toulouse
yamine@enseeiht.fr

ABSTRACT
User interfaces take an important role in commercial success of past and future aircraft programs. They have a direct influence on flight safety, crew members opinion of the aircraft and the efficiency of aircraft operations. As computing power and complexity of systems increase, user interfaces must become more and more effective in order to allow human operators to deal with this complexity. The goal of this project is to develop a formal language which will enhance the development process,verification and validation of these user interfaces.

Author Keywords
Human Machine Interface; User Interface; Design; Specification; Verification; Validation; Formal Language.

ACM Classification Keywords
H.5.2. User Interfaces: Evaluation/methodology

General Terms
Design; Human Factors; Languages; Reliability; Security; Verification.

ORIGIN AND UNDERLYING PRINCIPLES
Critical embedded UIs (*user interfaces*) conception is a discipline which lies at the limits of two software engineering domains: embedded software development and user interface development. As a consequence, critical embedded user interfaces must comply with a set of conflicting constraints coming from these two different domains.

While embedded software makes robustness a priority, UIs must be flexible and configurable. While embedded systems have limited ressources, UIs must be complete and integrate lots of functionalities. While critical software have strong timing constraints, UIs must be user friendly and let the user interact at his own pace. While critical systems are often real-time, UIs are mostly event-driven.

This set of conflicting constraints is a strong limitation for critical UIs, and makes their development particularly costly and time-consuming.

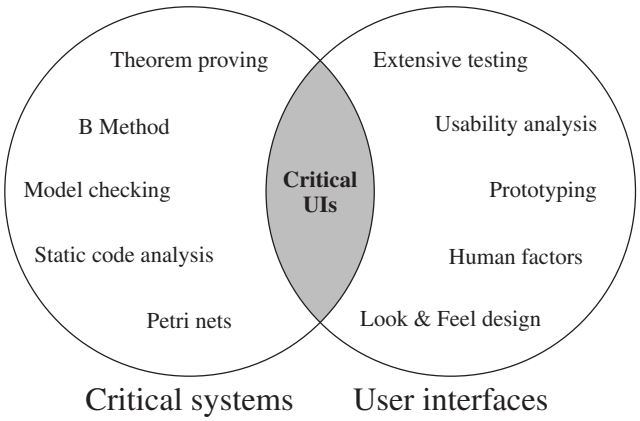


Figure 1. Critical UIs : collision of two clashing domains

The apparition of model-based development has been a big leap ahead for non-critical UI development, allowing for more complex UIs to be designed in a quicker and cheaper way. However, critical UIs did not really profit yet from this new fresh air. They indeed lack development methodologies and tools which would be compatible with safety requirements while allowing quick and efficient development.

A good critical UIs development process should take into account the methods and tools in use for both of these domains (see Fig. 1).

This project tends to make a constructive use of formal methods in critical UIs development. As so, it relies on previous works on this subject. Many works were already performed on formal modelling of software aspects of UIs by suggesting two main approaches.

The first one is based on proof systems where a model of the system is described by sets of variables, sets of operations, sets of events, timing properties and invariants. Operations must preserve invariants and properties. To ensure the correctness of these specifications, proof obligations are generated and must be proved. So, for example, Z and VDM were used to define atomic structures of interactions ([11],[12]) and HOL (High Order Logic Theorem Prover) was used to check user interface specification ([7]). B system was also used for the an incremental specification design of interactive systems ([3], [2]).

The second one is based on the evaluation of logical properties of a system on a state transition system. This technique was used, for example, to verify formally with SMV some

Submitted for review.

properties of interactive systems ([8]). Model checking was used in ([14],[5]) where the user and the system are modelled by object Petri nets (ICOs). In [15], Murϕ is used for modelling an autopilot system and a mental model in order to analyse cockpit interfaces and to detect some potential problems. Model checking was also coupled with static analysis of program codes in [10] or [9] to extract and abstract a formal model of an interactive system and to check various properties on it modelling and analysis of human-automation interaction has also been demonstrated [6].

This project aims at inverting the approach in [10] or [9] by defining a language which would allow to formaly describe model the behaviour of UIs. The model perimeter is the *dialog controller* and *Logical interaction* of the UI Arch Model from [4]. The model will be using abstractions to perform model checking on the high level behaviour of UIs.

MODELLED RELATIONSHIPS

The language goal is to model the UAs (*user applications*) which are applications that comply with the ARINC 661 Specification. ARINC 661 is the industry standard for Cockpit Display System Interfaces to User Systems [1].

This formal language would take into account aspects of both abstract UIs and embedded systems :

- UIs (structure, internal behaviour, human interaction, interface with systems)
- Critical systems (resources footprint, timing constraints, reliability, integrity)

A typical model specified using this language would be a closed system, with a static structure, interacting with two types of abstract external objects (see Fig. 2):

- Display System (e.g. CDS (*Cockpit Display System*))
- Application (e.g. Other aircraft application)

The system would be specified by organizing abstract components in order to generate the desired behaviour. The modelled system would then allow to generate different projections:

- Models to be checked
- ARINC 661 compliant code
- Test cases and scenarios for validation

PROBLEMS ADDRESSED

As of today, in order to pass qualification, UIs must undergo a massive, long test procedure, like many other pieces of embedded software. However, the fact that UIs interact directly with humans makes the conception process much heavier. Indeed, fully automatic generation of UIs still does not yield satisfactory results, and extensive automatic testing is not possible (see Fig. 3).

As a consequence, UIs have to be tested manually during the verification and validation phases. Since these tests happen at the end of the conception loop, errors found during manual tests can become excessively costly to correct, needing to get back in the conception loop.

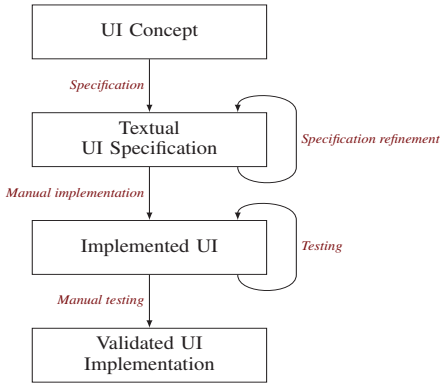


Figure 3. Classical UI conception process

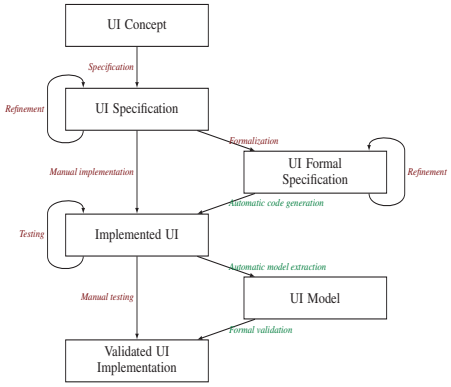


Figure 4. State-of-the-art UI conception process

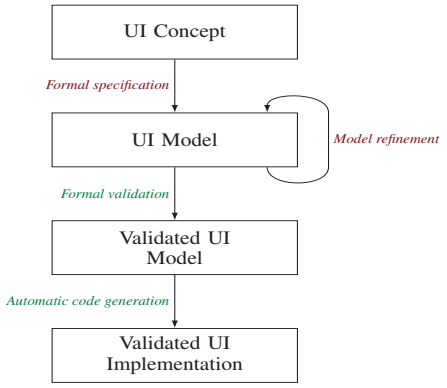


Figure 5. Proposed UI conception process

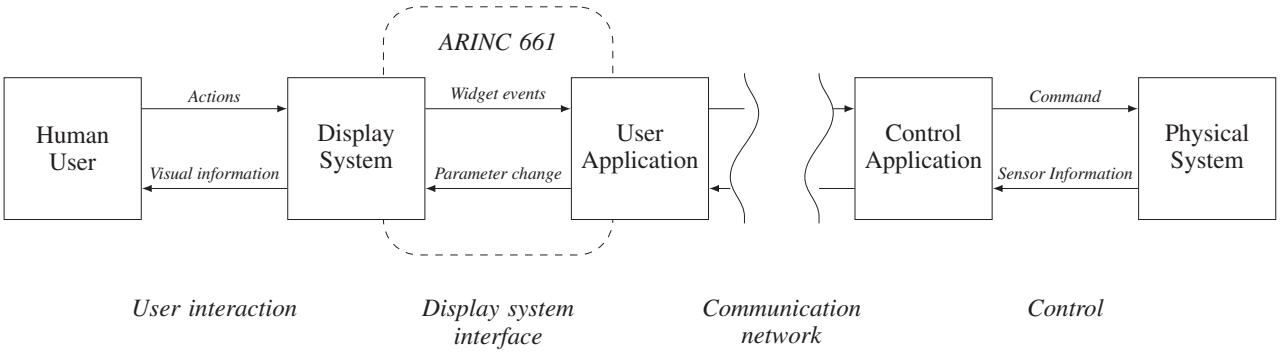


Figure 2. Aircraft UIs architecture. The UA interacts with the CDS one one side, and with other aircraft applications on the other side.

New tools have been developed in order to enhance this heavy process. Some of these tools can be used in order to generate an abstract model of the UI, taking its code as an input, allowing to perform formal validation on the implemented UI [9]. Other methods allow to specify UIs using specific languages in order to validate critical parts [13].

However, a common language allowing an efficient collaboration of these different tools does not exist yet. That is why critical UIs development process can become complex and costly (see Fig 4).

Progress toward a straightforward conception process (see Fig 5) is required to the conception of future UIs. The existence of a formal language taking into account all the aspects of critical UIs (see Fig 1) would greatly help to fulfil this goal.

APPLICATIONS

The application field of this formal language will be formal specification, verification and validation of ARINC 661 compliant UAs.

By enhancing the development process, this approach aims at reducing development costs of critical UIs. The approach could allow to shorten testing phases and to reduce development cycles time.

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

The language will be limited to embedded critical UIs complying with the ARINC 661 specification. Thus, the following limitations will exist:

- Graphical user interfaces only
- No dynamic instantiation of widgets (UI structure specified at conception time, no modifications at run time)
- No concern about rendering and other low level concerns (Dealt with the CDS, outside of our scope)

In a first approach, we will target the work around modelling the complexity of UI dynamic behaviour, dropping the following concerns:

- Look&Feel (formatting, colouring, screen structure)
- Human factors (usability analysis, ergonomics)

However, the language will be designed to be as expandable as possible, in order to ease future developments.

REFERENCES

1. ARINC. *Specification 661*, supplement 5, draft 1 ed., March 2012.
2. At-Ameur, Y., Brhole, B., Girard, P., Guittet, L., and Jambon, F. Formal verification and validation of interactive systems specifications. from informal specitications to formal validation. In *Conference of Human Error, Safety and Systems Development, HESSD* (Toulouse, 2004).
3. At-Ameur, Y., Girard, P., and Jambon, F. Using the B formal approach for incremental specification design of interactive systems. In *Engineering for Human-Computer Interaction*, S. Chatty and P.Dewan, Eds., vol. 22, Kluwer Academic Publishers (1998), 91–108.
4. Bass, L., Little, R., Pellegrino, R., Reed, S., Seacord, R., Shepard, S., and Szezur, M. The arch model: Seeheim revisited. In *Proceedings of the 1991 User Interface Developers Workshop*, ACM (Seeheim, 1991).
5. Bastide, R., Navarre, D., and Palanque, P. A tool-supported design framework for safety critical interactive systems. *Interacting with Computers 15*, 3 (juin 2003), 309–328.
6. Bolton, M., and Bass, E. Evaluating human-automation interaction using task analytic behavior models, strategic knowledge-based erroneous human behavior generation, and model checking. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on* (oct. 2011), 1788 –1794.
7. Bumbulis, P., Alencar, P., Cowan, D., and Lucena, C. Validating Properties of Component-Based Graphical User Interfaces. In *Eurographics Workshop on Design, Specification, and Verification of Interactive Systems (DSV-IS96)*, S. Verlag, Ed. (1996), 347–365.
8. Campos, J. C., and Harrison, M. D. Formally verifying interactive systems: A review. In *Eurographics Workshop on Design, Specification, and Verification of Interactive Systems (DSV-IS97)*, M. D. Harrison and J. C. Torres, Eds., Springer (1997), 109–124.

9. Cortier, A. *Contribution à la validation formelle de systèmes interactifs Java*. PhD thesis, Universite Paul Sabatier - Institut Supérieur de l'Aéronautique et de l'Espace (ISAE-Supaero), 2008.
10. d'Ausbourg, B., Seguin, C., and Roché, P. Assisting the automated validation process of user itnerfaces. In *The 20th International Conference on Software Engineering* (Kyoto, Japan, 1998).
11. Duke, D., and Harrison, M. Abstract Interaction Objects. In *Proceedings of Eurographics conference and computer graphics forum*, vol. 12 (1993), 2536.
12. Duke, D., and Harrison, M. Towards a Theory of Interactors. Tech. Rep. 7040, Amodeus Esprit Basic Research Project, 1993. System Modelling/WP6.
13. Madani, L. *Utilisation de la programmation synchrone pour la spécification et la validation de services interactifs*. PhD thesis, Université Joseph Fourier - Grenoble 1, 2007.
14. Palanque, P. A., Bastide, R., and Sengès, V. Validating interactive system design through the verification of formal task and system models. In *Proceedings of the IFIP TC2/WG2.7 Working Conference on Engineering for Human-Computer Interaction*, Chapman & Hall, Ltd. (London, UK, UK, 1996), 189–212.
15. Rushby, J. M. Analyzing cockpit interfaces using formal methods. *Electr. Notes Theor. Comput. Sci. 43* (2001), 1–14.

Modelling and systematic analysis of interactive systems

Michael D. Harrison
Queen Mary University London
and Newcastle University
UK
michael.harrison@newcastle.ac.uk

José C. Campos
University of Minho
and HASLab/INESC TEC
Portugal
jose.campos@di.uminho.pt

Paolo Masci
Queen Mary University London
UK
paolo.masci@gmail.com

Nigel Thomas
Newcastle University
UK
nigel.thomas@newcastle.ac.uk

INTRODUCTION

Two aspects of our research concern the application of formal methods in human-computer interaction. The first aspect is the modelling and analysis of interactive devices with a particular emphasis on the user device dyad. The second is the modelling and analysis of ubiquitous systems where there are many users, one might say crowds of users. The common thread of both is to articulate and prove properties of interactive systems, to explore interactive behaviour as it influences the user, with a particular emphasis on interaction failure. The goal is to develop systematic techniques that can be packaged in such a way that they can be used effectively by developers. This "white paper" will briefly describe the two approaches and their potential value as well as their limitations and development opportunities.

THE ANALYSIS OF INTERACTIVE DEVICES

Origin and Underlying Principles

This research has been concerned with the analysis of interactive devices such as medical infusion pumps, in-car air conditioning systems and flight management systems. In order to analyse these devices a tool has been developed (IVY) [3] which provides a front end to a model checker. The aim has been to develop models using a simple notation that is orientated around action, producing a textual representation of a finite state model. These device models are then subjected to systematic analysis using properties based on a set of standard templates. Counter-examples that are generated are visualised in a format that aims to ease mutual exploration with domain and human factors specialists. The IVY tool supports this approach to the analysis of interactive systems in a number of ways.

- IVY supports the editing of models. These models are described in Modal Action Logic (MAL) which is a deontic logic of actions that allows focus on the actions that the user engages in when using the device. The tool translates MAL models into SMV [6] and invokes NuSMV [5] to check properties of the model.
- IVY provides property patterns and the means of instantiating property templates associated with the patterns. The patterns are designed to probe aspects of the interactive behaviour of the device systematically in a process that is similar to the application of "usability heuristics". By using the IVY property editor it is possible to define a battery of properties to which the device can be subjected. These properties explore mode as well as the relationship between attributes of the device and what is visible about the device.
- IVY provides a trace visualizer that eases the exploration of counter-examples when properties fail. These counter-examples provide material for scenarios that can depict problematic situations in the use of the device.

This tool has also been used to explore the use of information resources to restrict analysis to paths that are "plausible" from the perspective of human factors or domain specialists [8]. Information resource constraints make explicit the information that it is assumed the user will use in order to help decide what to do next to achieve a goal.

Modelled Relationships

These models of interactive devices focus on actions. Models are designed to make explicit whether or not a state attribute or action is visible to a user. State attributes and actions have also been added to the model to capture the activities and meta-variables that reflect the use for which the device is intended. These activities and meta-variables will have been determined by studying the work that the device is designed to be embedded within. Properties checked of the model include determining the relation between attributes specified to define modes and those that indicate variables that are relevant to modes. Properties are also concerned with determining that intended goals of the

device are reachable subject to resource constraints in order to generate plausible paths that can be further explored by domain and human factors experts.

Problems Addressed

The central problem of this work is to provide a systematic means of analysis of interactive systems that is objective and can be performed by analysts who are not experts in the use of formal methods. A number of standard patterns have been developed that can be instantiated for the model in question using the IVY tool. Traces of counter-examples are visualized to aid the construction of appropriate scenarios.

One example of such a standard property is whether the user of a device can recover from a wrong action. This property has a standard template and can be expressed in CTL [6] as.

$$AG(attribute = value \rightarrow AX(action1 \rightarrow EX(action2) \,\&\, AX(action2 \rightarrow (attribute = value)))))$$

The Alaris infusion pump (one of the systems studied) includes chevron buttons that allow the incrementing or decrementing of data as it is entered. This property template can be instantiated by applying it to single chevron up buttons and single chevron down buttons *sup* and *sdown.* When the device is in a mode determining infusion rate entry (*entrymode=rmode)* when the infusion rate is not locked *(!rlock*) the property is expressed as.

$$AG((infusionrate = IVAL1 \,\&\, entrymode=rmode \,\&\, !rlock) \rightarrow AX(sup \rightarrow (EX(sdown) \,\&\, AX(sdown \rightarrow infusionrate = IVAL1))))$$

IVAL1 is a meta-variable that ranges over the possible values of *infusionrate*.

Applications

Recent work has concerned the development of models of medical infusion pumps developed by three manufacturers [3,4]. These designs are the result of interesting and in some cases subtly different design decisions. Two large models have been constructed. The models reuse a common module that captures the characteristics of the underlying pump. Both models have been analysed using a battery of properties that have been instantiated for the two models so that similar properties can be checked of the two devices. The applicability of the tool in the aerospace context is also being investigated.

Limitations and Development Opportunities

MAL provides a notation that coincides well with the interaction structure of scalable systems. However the size of the models generated if interaction details are to be captured can be very large and this means that model checking is either impossible in ordinary available computer technology or requires turn around times of hours rather than minutes. Off the shelf model checkers do not exploit the multi-processor capabilities of modern computers. An alternative approach that is being explored is to translate models systematically into the specification language of PVS so that properties that are most appropriate for theorem proving, particularly concerned with the visibility of aspects of the underlying state of the model, can be proved more efficiently than would be possible using model checking. Early steps towards this work can be found in [7]. Theorem proving remains a relatively difficult procedure and therefore standard formats and procedures are also being explored, to make it easier for analysts to develop models and prove properties.

Even if MAL is well suited to model large systems, two further issues related to the modelling approach deserve attention.

- On the one hand the use of IVY requires that a model be developed for the sole purpose of carrying out the analysis. This represents a significant barrier towards adoption, especially in HCI. Solving this means either finding alternatives to the (semi-) automated development of the models, or alternative notations that better integrate into a development process. In the first case, work has been done in reverse engineering user interface code with the goal of producing models of the supported interaction [13]. However, this means that the analysis can only be performed once the system has been (at least partially) developed. An interesting alternative would be to consider the generation of models from the design artefact. Storyboards are envisaged as a possibility (indeed tools such as CogTool [11] use a similar approach). In the second case, a tabular version of the language – with lines representing actions, and columns representing attributes – could be envisaged as a more "engineer friendly" notations to express the models, see for example [1].
- On the other hand, the step from analysing the model to certifying the final system remains a challenge.

THE ANALYSIS OF MOBILE AND UBIQUITOUS SYSTEMS

Origin and Underlying Principles

This research is concerned with the analysis of systems that combine public displays and hand-held personal devices. These systems provide relevant and tailored information to users in physical environments such as hospitals, shopping malls, airports or office environments. The success of such systems depends on effective testing and user evaluation. They must be natural to users, enabling an enhanced experience of the place in which the system is situated. The evaluation of these systems is often impractical within their designed target environments. We are exploring predictive models of the interactive behaviour of these environment designs that include an understanding of the proposed context. Properties are required that relate to how the smart environment enhances or otherwise the collective user experience of complex spaces.

Modelled Relationships

Systems are described as activities in PEPA [10]. To illustrate the modelled relationships a smart system will be briefly illustrated that supports, by means of public displays, visitor routing to a particular location, where the system is aware of the location and destination of visitors.

The PEPA model consists of processes modeling the behavior of visitors, arbitrators, slot managers, slots and places. Slots and places are instantiated for each particular location. The display consists of a number of slots. The slot manager and the arbitrator, in each location, ensure that requested information is displayed and that no two slots show the same information. Groups of visitors are defined with the same starting location and final destination. For example, a visitor starting in location *a* and heading for location *d* first tries to get a place in location *a* where it is possible to see the display (*lasd* is the action of trying to acquire a place in location *a*). Once a place has been acquired the visitor engages in action *laee* to find out where to go next. The request is engaged as soon as there is an available display slot that displays the information. When the information is displayed the visitor releases the place in location *a* (action *lasu*) and receives the information (i.e. any of the matching destinations in the process *VisEdRec*). The visitor then proceeds to the indicated next location (e.g. *VisEdtoLb* means that the incomer with final destination *d* now first needs to proceed to location *b*). The arrival at the final destination is modelled by the process that remains in the state V isEdArrived forever.

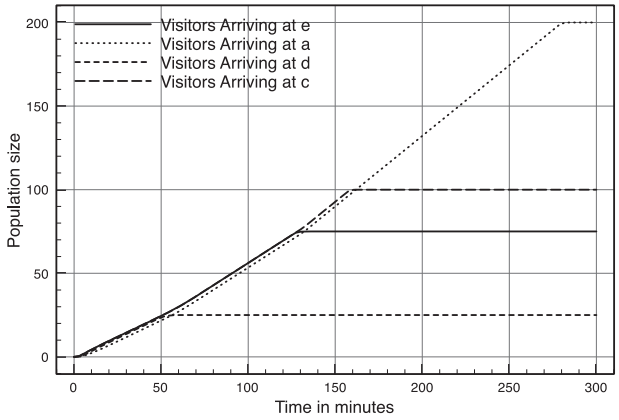
```
VisEytoLx = (lxsd, s).(lxey, a).(lxsu, s). VisEyRecx
VisEytoLy =  (lesy, s).(lyey, a).(lysu, s). VisEyArrived
VisEyRecx = (eexle, r).VisEytoLe +
            (eexle, r).VisEytoLa +
            (eexle, r).VisEytoLb +
            (eexle, r).VisEytoLc +
            (eexle, r).VisEytoLd
VisEyArrived = (nop, a). VisEyArrived
```

The model of a visitor has three rate parameters, modeling the average time needed to perform the related activity. The average duration of activities is defined by their rates. Rates are assumed to be measured in minutes. So, for instance, letting $s = 10$ implies that the average time a visitor needs for sitting down or standing up is 6 sec. (i.e. 0.1 min.). Rate $a = 2$ models the average time a visitor needs to make a request equal to 30 seconds. The rate $r = 1$ models the average time to receive the requested information equal to 1 minute. It is further assumed that visitors are arriving over a certain period of time and heading for different destinations.

A number of factors could have an impact on the person, or people, in the environment affecting their experience of it. The relevance of these factors depends on physical context. Prior to analysis they could be assessed through some form of user evaluation which can be converted into properties of the model. These factors could include:

- visibility and interpretation of display directions
- continuing visibility of directions whatever the user's location
- sense of progress towards the destination
- ability to remember the route having completed it once
- a broader sense of the building and the facilities it offers
- how long to wait before the display is relevant to them (either in terms of time or number of refreshes of the display)
- guaranteed time to arrival
- impact if many users need to recover from some scheduling change due to congestion in the environment
- a sense of congestion, that there are too many people in the surrounding space

Preliminary results have involved using fluid flow models of the systems to provide average behaviors. For example, notions of congestion can be addressed by exploring the arrival of visitors to various locations in the building.



Problems Addressed

We are interested in stochastic properties of systems involving multiple people, that is crowds of people interacting with the system [9,12]. The technique uses PEPA and a combination of fluid flow and simulation techniques.

We have also explored a mixed approach connecting formal (Petri-net based) models of the ubiquitous systems with virtual reality simulations of the target environment in order to support different levels of analysis, from empirical studies to formal verification [14].

Applications

Early applications have included the exploration of emergency egress in an office building and out-patient behaviour in the context of a hospital department.

Limitation and Development Opportunities

This work is in early stages. The results relating to emergency egress are promising [12] and have produced results that are consistent with other simulations of the

same emergency egress problem. There are several opportunities for future work and limitations to overcome. For example,

- The work is based on a stochastic process algebra (PEPA) [10] and the notation is not conducive to the expression of large models. Relatively small models are quite unwieldy to represent. We are however working on alternative approaches to developing simulations that may be more expressive and scalable.
- The semantics of the approach is based on exponential memory-less distributions and this approach may not be the most appropriate for the kind of problems we wish to tackle. PEPA uses fixed rates rather than functional rates and many of the scheduling problems, that can be solved dynamically using a smart environment to improve flow, require functional rates
- We need to quantify the characteristics of these environments that capture the experience that people within the environment. We can measure flow in terms of delay or slack time for example, but how do we quantify the frustration that users suffer? There are models of emotion that might help us here [15].

ACKNOWLEDGMENTS

José Campos funded by the ERDF – European Regional Development Fund – through the COMPETE Programme, and by National Funds through the FCT – Portuguese Foundation for Science and Technology – the APEX project PTDC/EIA-EIA/116069/2009. Michael Harrison and Paolo Masci are funded by the CHI+MED project: UK EPSRC Grant Number EP/G059063/1. Nigel Thomas is funded by the AMPS project: Analysis of Massively Parallel Stochastic Systems EPSRC EP/G011389/1.

REFERENCES

1. Abowd, G.D., Wang, H-M. and Monk, A.F. A formal technique for automated dialogue development. DIS'95: Proceedings of the 1st Conference on Designing Interactive Systems: processes, practices, methods and techniques. ACM Press. pp. 219-226. 1995.
2. Campos, J.C. and Harrison, M.D. Systematic analysis of control panel interfaces using formal tools. In N. Graham and P. Palanque, editors, Interactive systems: Design, Specification and Verification, DSVIS'08, volume 5136 of Springer Lecture Notes in Computer Science, pages 72–85. Springer-Verlag, 2008.
3. Campos, J.C. and Harrison, M.D. Interaction engineering using the IVY tool. In G. Calvary, T.C.N. Graham, and P. Gray, eds, Proc. ACM Symp on Engineering Interactive Computing Systems, pp 35-44. ACM Press, 2009.
4. Campos, J.C. and Harrison, M.D. Modelling and analysing the interactive behaviour of an infusion pump. Electronic Communications of the EASST, 5, 2011.
5. Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F. Pistore, M., Roveri, M., Sebastiani, R. and Tacchella, A. NuSMV 2: An Open Source Tool for Symbolic Model Checking. In K. G. Larsen and E. Brinksma, editors, Computer-Aided Verification (CAV '02), volume 2404 of Lecture Notes in Computer Science. Springer-Verlag, 2002.
6. Clarke, E.M., Grumberg, O and Peled, D.A. Model Checking. MIT Press, 1999.
7. Doherty, G.J., Campos, J.C. and Harrison, M.D. Representational Reasoning and Verification. Formal Aspects of Computing, 12(4):260-277. 2000.
8. Doherty, G., Campos, J.C. and Harrison, M.D. Resources for situated actions. In N. Graham and P. Palanque, editors, Interactive systems: Design, Specification and Verification, DSVIS'08, volume 5136 of Springer Lecture Notes in Computer Science, pages 194–207. Springer- Verlag, 2008.
9. Harrison M.D. and Massink M. Modelling interactive experience, function and performance in ubiquitous systems. In: Electronic Notes in Theoretical Computer Science: 4th International Workshop on the Practical Application of Stochastic Modelling (PASM). Imperial College, London: Elsevier. 2010.
10. Hillston, J.A. A compositional approach to performance modelling. Cambridge University Press, 1996.
11. John, B., Prevas, K., Salvucci, D., & Koedinger, K.. Predictive Human Performance Modeling Made Easy. Proceedings of CHI 2004. ACM, 2004.
12. Massink M, Latella D, Bracciali A, Harrison MD and Hillston J. Scalable context-dependent analysis of emergency egress models. Formal Aspects of Computing: Volume 24, Issue 2, Page 267-302. 2012
13. Silva, J.C., Silva, C., Goncalo, R., Saraiva, J. and Campos, J.C. The GUISurfer tool: towards a language independent approach to reverse engineering GUI code. In Proceedings of the 2nd ACM SIGCHI Symposium on Engineering interactive computing systems, pages 181-186. ACM. 2010.
14. Silva, J.L., Campos, J.C. and Harrison, M.D. Formal analysis of Ubiquitous Computing environments through the APEX framework. In ACM Symposium on Engineering Interactive Computing Systems (EICS2012). ACM. 2012. (accepted)
15. Steunebrink, B.R., Dastani, M. and Meyer, J-J Ch. A formal model of emotions: integrating qualitative and quantitative aspects. In M. Ghallab et al. (eds) ECAI 2008 IOS Press Pages 256-260. 2008.

Position Paper: Modelling Interactive Critical Systems using Interactive Cooperative Objects Formalism

David Navarre, Philippe Palanque
ICS - IRIT
University of Toulouse, France
{ navarre, palanque } @ irit.fr

ORIGIN AND UNDERLYING PRINCIPLES

Interactive systems are particular reactive systems where inputs and outputs are linked to user. This explains why the first approaches in formal modelling such systems tried to catch interactive systems dialog by describing the inner states of the systems and how events can impact inner states. Parnas [21] was one of the first to use formal methods based on finite state machines (FSMs) for the specification of human computer interaction issues.

State based formalisms provide a description of all of the states a system can be in and the transitions between these available states. By doing so, such formalisms allow to take into account the interaction context and are therefore more adequate for modelling interactive applications where user's actions play a central role.

FSMs or finite automaton are a mathematical formalism which offers a graphical representation of models. The models formally represent the behaviour of a system considered as the composition of states, transitions and actions. Extensions to this formalism have been made to deal with its constraints. For example, Recursive Transition Networks (RTN) [24] provides hierarchical relationships between state machines, Augmented Transition Networks (ATN) also defined by Woods [24] allow the use of preconditions via variables on arcs. Generalized Transition Networks (GTN) proposed by Kieras [14] allows representing preconditions, actions and states can be GTNs.

Statecharts is a formalism developed in 1987 by Harel [10] for the graphical modelling of complex systems in which interleaving of actions is important. Statecharts extend traditional FSMs with useful characteristics such as the modelling of super-states (providing hierarchy), and interleaved behaviour. They have become used widespread with a variant of them being part of the UML. Their mathematical foundation allows (at least partly) model validation.

Petri nets, on which our approach is based, have already been used for modelling interactive systems [13]. They have been initially introduced by C. A. Petri in 1962 [22] and have been extensively used for the modelling of discrete event systems. They are particularly relevant to describe multimodal interactive systems as they allow the description of true concurrency.

Modelling is a key issue when dealing with safety critical interactive systems, justifying the use of formal description techniques to provide means:

- To define in a complete and unambiguous way the behaviour of the input and output devices, the interaction techniques, the interactive system behaviour
- To reason about that models in order to be able to assess the behaviour of the interactive system (for instance, is the interactive system able to return to its initial state, are all the interactive system states rendered to the user?)
- To support via dedicated tools editing, verification, validation and interactive prototyping of the behaviours and to modify and adjust them according to user's requirements and global performance.

MODELED RELATIONSHIPS

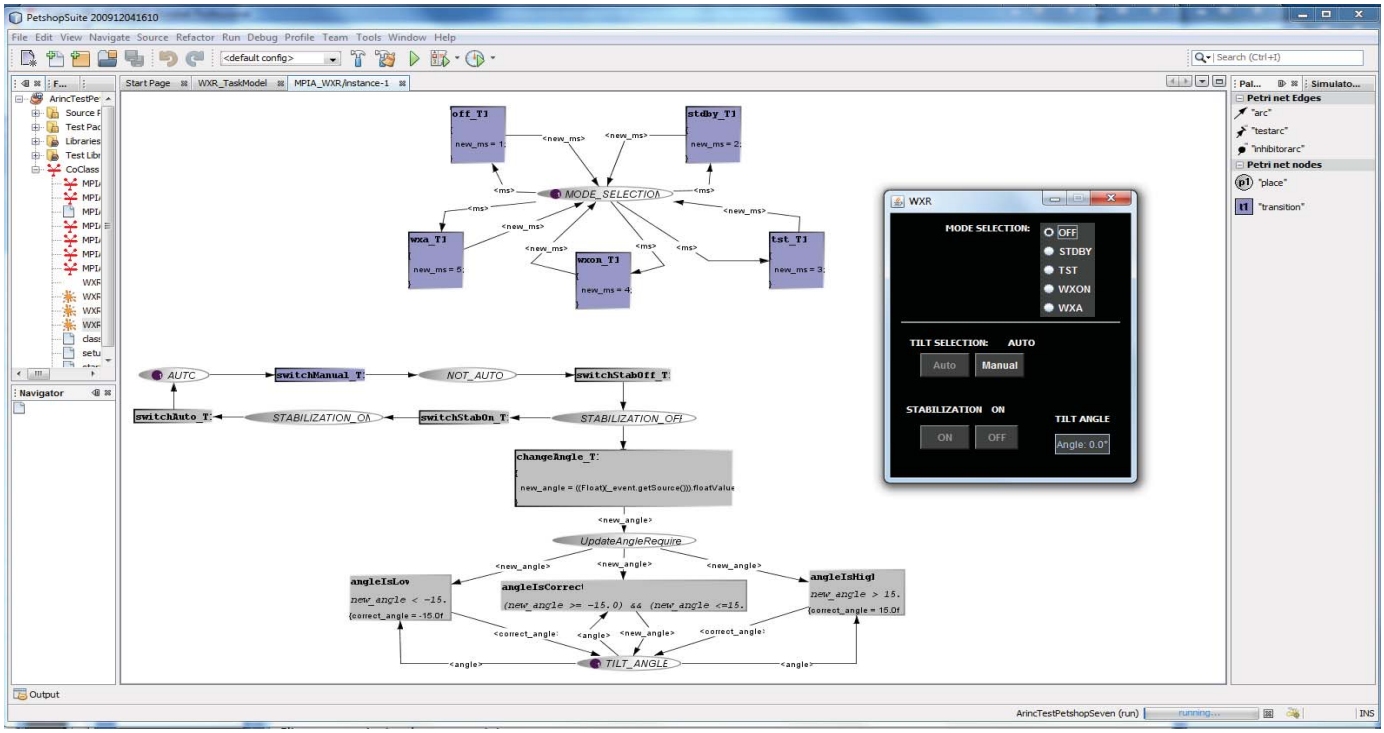
The ICO formalism is a formal description technique dedicated to the specification of interactive systems [17]. It uses concepts borrowed from the object-oriented approach (dynamic instantiation, classification, encapsulation, inheritance, client/server relationship) to describe the structural or static aspects of systems, and uses high-level Petri nets [9] to describe their dynamic or behavioural aspects.

ICOs are dedicated to the modelling and the implementation of event-driven interfaces, using several communicating objects to model the system, where both behaviour of objects and communication protocol between objects are described by the Petri net dialect called Cooperative Objects (CO). The ICO formalism has been applied to other domains than user interfaces as, for instance, CORBA services specification [4] and [5].

As illustrated by Figure 1, in the ICO formalism, an object is an entity featuring four components: a cooperative object which describes the behaviour of the object, a presentation part (i.e. the graphical interface), and two functions (the activation function and the rendering function) which make the link between the cooperative object and the presentation part.

An ICO specification fully describes the potential interactions that users may have with the application. The specification encompasses both the "input" aspects of the interaction (i.e. how user actions impact on the inner state of the application, and which actions are enabled at any given time) and its "output" aspects (i.e. when and how the application displays information relevant to the user).

Thanks to Petri nets, an ICO specification is fully executable, which makes possible to prototype and to test an application before it is fully implemented. The editing, execution and verification is done using the dedicated supporting tool called PetShop (illustrated by Figure 1) [19].



User Events	Event handler	Activation Rendering
asked_off	Off	setWXRModeSelectEnabled
asked_stdby	Stdby	setWXRModeSelectEnabled
asked_tst	Tst	setWXRModeSelectEnabled
asked_wxon	Wxon	setWXRModeSelectEnabled
asked_wxa	Wxa	setWXRModeSelectEnabled
asked_auto	switchAUTO	setWXRTiltSelectionEnabled
asked_stabilization	switchSTABILIZATION	setWXRTiltSelectionEnabled
asked_changeAngle	changeAngle	setWXRTiltSelectionEnabled

ObCS Node name	ObCS event	Rendering method
MODE_SELECTION	token_enter	showModeSelection
TILT_ANGLE	token_enter	showTiltAngle
AUTO	marking_reset	showAuto
AUTO	token_enter	showAuto
AUTO	token_remove	showAuto
STABILIZATION_ON	marking_reset	showStab
STABILIZATION_ON	token_enter	showStab
STABILIZATION_ON	token_remove	showStab

Figure 1. Example of an application modeled using ICO: The top part is a Petri net modeling the behavior of the corresponding interactive application (called WXR). It represents how the application current evolved when receiving user events. Two functions relate the Petri net to the graphical widow. The first one, called Activation function (table on the bottom-left part) describes how user events lead to transition firing and how transition availability makes user event available (for instance, the seventh line describes that when users press the button CTRL on the right of the label "TILT SELECTION" (triggering the high level event asked-auto), the corresponding transition "switchAUTO_T1" is triggered, making the application going to the state where the tilt selection is not automatic; if the corresponding transition is not available, the button would have been disabled). The second function, called Rendering function (table on the bottom-right part) describes hos state changes within the Petri net is translated into graphical rendering (for instance the first line describes that when a token enters the place MODE_SELECTION, the method showModeSelection is called, making a graphical change).

Several modelling challenges have been addressed using ICO:

1 Dealing with new input devices: modern user interfaces offer to the users more and more exotic input devices. As show in the virtual chess case study [18], ICOs are able to manage input devices such as the "Floc Of Birds" by poling at regular time interval the input device in order to capture motions performed by the users. More general multimodality aspects have been discussed in [15].

2 Dealing with visualization beyond standard 2D desktops: in a similar way as for input devices, output devices are likely to evolve and be more demanding for the description technique than the standard 2D screens. Manipulating languages dedicated to these new output devices is thus required. As shown with the chess case study [18], ICOs are able to handle exotic output devices as stereoscopic glasses.

3 Dealing with widgets: nowadays, most interaction between users and systems takes place by means of predefined standard interactive objects called widgets. Even though introduced more than 20 years ago in the IBM CUA standard [11] they are still largely used as they provide significant benefits both to the users (making easy to predict how they have to be manipulated and how they behave) and to the developers and they are usually available in the programming environment and thus require little effort to be integrated. In the case study of ARINC 661 specification [1][2] we have described using ICOs 21 widgets from the standard.

4 Dealing with large scale applications: most User Interface Description Languages have, so far, remained at the prototype level being only tested on case studies as a way to experiment their capabilities [16]. Being able to deal with large case studies and real world application will become a requirement for these languages if their authors want to see them going beyond the research laboratories. In the interactive cockpit application we have been partly addressing such problem. To deal with the large number of models executed simultaneously PetShop tool had to be restructured.

5 Dealing with the true concurrency requirement: mass market products such as the Wii game console [23] or the iPhone [12] feature a native multimodal interaction either by means of several input devices (two or more wiimotes for instance) or by means of a multimodal device as the multitouch tactile interaction on the iPhone. Being able to describe user interfaces for such new products will, for sure, become a necessity in the near future for next generation user interfaces description languages. While, thanks to their "true concurrency" semantics, Petri nets will remain (with other underlying formalisms such as Pi-Calculus) a good candidate to handle such behaviours, these new products will be more demanding in terms of number of concurrent threads, synchronisation between threads and response time. ICOs have been used in several domains for describing multimodal interfaces [15] dealing with several different interaction techniques and input devices (speech, gesture, bi-manual …) at a time.

6 Formal analysis of models: the use of formal description techniques can be greatly enhanced if they are supported by formal analysis techniques. In the area of Petri nets, such analysis techniques can be used for the detection of deadlocks (when no transitions are available/fireable in the Petri net model), the presence or absence of a terminating state, the boundedness of the model, the liveness of the model, the reversibility and home state or to verify requirements, i.e. to verify properties of the system model such that a certain state can never be reached for example or that a given interface component is always enabled. Furthermore, certain analysis techniques can be used to extract scenarios of events leading to a particular state; this is useful for tracing history and for connecting interactive systems behaviour to incidents or accidents [3]. With ICO the analysis capabilities are thus limited to analysis performed on the underlying net.

7 Exploiting models: in [19] we have already presented the use of our model-based approach as a way of supporting in an easy way rapid system prototyping. Executing the specifications (as when models are run within PetShop) has the immediate benefits of providing a prototype available for usability evaluation. In [7] we improve the idea of using model-based approaches as a support for usability evaluation using a marking graph to describe low level interaction scenarios. This approach improves classical usability evaluation methods in two directions: supporting usability evaluation using model-based scenarios and model-based prototyping and modifying models to accommodate changes due to usability evaluation results.

PROBLEMS ADDRESSED

The reason for focusing on the use and deployment of formal description techniques lies in the fact that they are the only means to provide both modelling in a precise and unambiguous way, all of the components of an interactive application (presentation, dialogue and functional core) and to propose techniques for reasoning about (and also verifying) the models. Applying formal description techniques can be beneficial during the various phases of the development process from the early phases (requirements analysis and elicitation, user interface prototyping …) to the later ones including validation (usability testing, functional testing …).

According to the recurring desire of increasing the bandwidth between the interactive system and the users, more sophisticated interaction techniques are continuously being proposed. Such proposals are usually presented in conferences such as ACM CHI (Human Factors in Computing

Systems) or UIST (User Interfaces Software and Technology) with a focus on the innovation and on the usability evaluation of interactive systems proposing such interaction techniques. While a significant effort is currently being undertaken by the CHI community in order to apply and extend current usability evaluation techniques to new kinds of interaction techniques very little has been done to improve the reliability of software offering these kinds of interaction techniques. As these new interaction techniques are currently more and more used in the field of command and control safety critical systems the potential for incidents and accidents increases. Similarly, the non-reliability of interactive software can jeopardize usability evaluation by presenting to the users unexpected or undesired behaviours.

Researchers and practitioners designing and developing those interaction techniques usually have to build their own user interface tools or twist programming languages [8] to be able to implement the interaction technique they want to design and evaluate. Once published, it remains a long way for these innovative interaction techniques to reach the maturity level required for dissemination in industry as several problems remains to be solved.

- The first problem is related to the scalability issue where research contributions must be able to go from demonstrational prototypes to real size applications.

- The second problem is related to the reliability issue i.e. to find ways to guarantee the correct functioning of the interactive system. Issues such as verification, validation and exhaustive testing are possible ways to address this problem.

- The third problem is related to the link between User Interface tools and the development process of interactive systems. The issue is here to be able to integrate such tools within development processes and other software development tools currently used in industry.

- The last but not least problem is related to the expressive power of the language i.e. how the language is able to cover the various elements that have to be addressed when user interfaces are described.

APPLICATIONS

Our formal description technique has been extended to deal with user interfaces such as multimodal [6] and 3D interfaces [18] and to address issues raised by various safety critical command and control systems such as Air Traffic Control workstations [19], Ground segments for satellite control [20] or civil [2] and military aircraft cockpits [6]. The expressive power of the notation makes it eligible for the description of new generation interfaces as challenges raised by multimodal interfaces are comparable with the ones they raise.

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

The proposed approach has already been used for modelling several aspects of interactive systems such as post-WIMP interaction, multimodality, real size case studies from different safety critical domains (such as Air traffic control, space operations, avionics…). Amongst the lesson learned we identified several interesting limitations in our approach such as:

- Improving our model analysis capability, to be able to investigate in a more precise way inner properties of safety critical interactive systems.

- Allowing model-based software testing dedicated to interactive systems to support the implementation process.

- Enhancing ICO with a presentation description language such as UsiXML Concrete User Interface language to allow an in-depth description of interactive applications.

To investigate these improvements, we have already started several projects with industrial partners in the domain of interactive cockpits or space ground systems.

REFERENCES

[1] Barboni, E., Conversy, S., Navarre, D. and Palanque, P., 2006. Model-Based Engineering of Widgets, User Applications and Servers Compliant with ARINC 661 Specification. In *proceedings of the 13th conference on Design Specification and Verification of Interactive Systems (DSVIS 2006),* Dublin, Ireland, July 2006, Lecture Notes in Computer Science, Springer Verlag. P 25-38

[2] Barboni, E., Navarre, D., Palanque, P. and Basnyat, S. 2006. Exploitation of Formal Specification Techniques for ARINC 661 Interactive Cockpit Applications. In *proceedings of HCI aero conference, (HCI Aero 2006)*, Seatle, USA.

[3] Basnyat, S., Chozos, N. and Palanque, P., 2006. Multidisciplinary perspective on accident investigation. *Reliability Engineering & System Safety Volume 91, Issue 12* , December 2006, Pages 1502-1520.

[4] Bastide, R., Palanque, P., Sy, O., and Navarre, D. 2000. Formal specification of CORBA services: experience and lessons learned. In Proceedings of the 15th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (Minneapolis, Minnesota, United States). OOPSLA '00. ACM, New York, NY.

[5] Bastide, R. Sy, O., Navarre, D. and Palanque, P.. A formal specification of the CORBA event service. IFIP TC6/WG6.1 4th international conference on formal methods for open object-based distributed systems (FMOODS); Stanford univ., California, USA. Kluwer; 2000.

[6] Bastide, R., Navarre, D., Palanque, P., Schyn, A. and Dragicevic, P., 2004. A Model-Based Approach for Real-Time Embedded Multimodal Systems in Air-

crafts. In *proceedings of the Sixth International Conference on Multimodal Interfaces (ICMI'04) October 14-15,* 2004 Pennsylvania, USA

[7] Bernhaupt R., Navarre D., Palanque P. and Winckler M., 2007. Model-Based Evaluation: A New Way to Support Usability Evaluation of Multimodal Interactive Applications. In *Maturing Usability: Quality in Software, Interaction and Quality* Springer Verlag, HCI series, April 2007, Law E., Thora Hvannberg E., Cockton G. & Vanderdonckt J. (Eds.).

[8] Dragicevic P., 2004. Combining Crossing-Based and Paper-Based Interaction Paradigms for Dragging and Dropping Between Overlapping Windows. In *Proceedings of the 17th annual ACM Symposium on User Interface Software and Technology (UIST'04),* pages193-196. ACM Press.

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

[10] Harel, D., "Statecharts: A visual formalism for complex systems." Sci. Comput. Program., s.l. :Elsevier North-Holland, Inc., 1987, Issue 3, Vol. 8, pp. 231-274.

[11] IBM 1989. Common User Access: Advanced Interface Design Guide. IBM, SC26-4582-0

[12] Jobs, S. P. et al. 2008. Touch Screen Device, Method, and Graphical User Interface for Determining Commands by Applying Heuristics. United States Patent Application 20080122796. Kind Code A, May 29, 2008.

[13] Keh, H. C. and Lewis, T. G., 1991. Direct-manipulation user interface modeling with high-level Petri nets. In Proceedings of the 19th Annual Conference on Computer Science (San Antonio, Texas, United States). CSC '91. ACM, New York, NY, 487-495.

[14] Kieras, D. and Polson, P. G., 1983. A generalized transition network representation for interactive systems. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Boston, USA, December 12 - 15, 1983). A. Janda, Ed. CHI '83. ACM, New York, NY, 103-106.

[15] Ladry J-F., Navarre, D., Palanque, P. 2009. Formal Description Techniques to Support the Design, Construction and Evaluation of Fusion Engines for SURE (Safe, Usable, Reliable and Evolvable) Multimodal Interfaces. In *proceedings of the tenth International Conference on Multimodal Interfaces (ICMI'09) October,* 2009 Boston, USA.

[16] Navarre, D., Palanque, P., Ladry, JF, Barboni, E. ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. In ACM Transactions on Computer-Human Interaction (TOCHI) 16 (4), 18

[17] Navarre, D., Palanque P. & Bastide R. (2003). A Tool-Supported Design Framework for Safety Critical Interactive Systems in Interacting with computers, Elsevier, Vol. 15/3, pp 309-328

[18] Navarre D., Palanque P., Bastide R., Schyn A., Winckler M., Nedel L. & Freitas C. 2005. A Formal Description of Multimodal Interaction Techniques for Immersive Virtual Reality Applications. *Proceedings of INTERACT 2005*, Roma, Italy, September 2005, Lecture Notes in Computer Science, Springer Verlag.

[19] Navarre, D., Palanque, P., Bastide, R., Sy, O. 2001. A Model-Based Tool for Interactive Prototyping of Highly Interactive Applications. in 12th IEEE, *International Workshop on Rapid System Prototyping*, Monterey (USA), IEEE Press 2001

[20] Palanque P., Bernhaupt R., Navarre D., Ould M., Winckler M., 2006. Supporting Usability Evaluation of Multimodal Man-Machine Interfaces for Space Ground Segment Applications Using Petri net Based Formal Specification. *Ninth International Conference on Space Operations*, Rome, Italy, 2006.

[21] 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. 1969

[22] Petri, C. A., "Kommunikation mit Automaten." PhD Thesis. Technical University Darmstadt : s.n., 1962.

[23] Schlomer, T., Poppinga B., Henze N. & Boll S. 2008. Gesture Recognition with a Wii Controller, Proceedings of the 2nd international conference on Tangible and embedded interaction, ACM, 2008

[24] Woods, W. A., "Transition network grammars for natural language analysis." Commun. ACM, s.l. : ACM, 1970, Vol. 13.

Model Checking Human-automation Interaction with Enhanced Operator Function Model

Matthew L. Bolton
San José State University
Research Foundation
NASA Ames Research Center
Moffett Field, CA USA
matthew.l.bolton@nasa.gov

Ellen J. Bass
Department of Systems and Information Engineering
University of Virginia
Charlottesville, VA USA
ejb4n@virginia.edu

ORIGIN AND UNDERLYING PRINCIPLES

Engineers use task analytic behavior models to describe the normative human behaviors required to control a system [12]. These models represent the mental and physical activities operators use to achieve the goals that the system was designed to support. Enhanced Operator Function Model (EOFM) [9], an extension of the Operator Function Model [13], represents human behavior as an input-output system using an XML notation.

An instantiated EOFM describes inputs from external sources and how a human operator produces actions as part of a task. Tasks in EOFMs are hierarchical representations of goal-driven activities that decompose into lower level activities, and, finally, atomic actions. EOFMs express task knowledge by explicitly specifying the conditions under which human operator activities can execute (preconditions), when they can repeat (repeat conditions), and what must be true when they finish (completion conditions) in Boolean expressions. Any activity can decompose into one or more activities or actions (sub-acts). A decomposition operator specifies the temporal relationships between and the cardinality of the decomposed sub-acts (when they can execute relative to each other and how many can execute). EOFM supports all of the decomposition operators in Table 1. EOFM also supports a visual notation (see Figure 1 for an example).

EOFM has formal semantics that specify how an instantiated EOFM model executes [9]. Specifically, each activity or action can have one of three execution states: waiting to execute (*Ready*), executing (*Executing*), and done (*Done*). An activity or action transitions between each of these states based on its current state; the state of its immediate parent, its siblings (activities or actions contained in the same decomposition), and its immediate children in the hierarchy; and the decomposition operators that connect the activity to its parent and its children. This formal semantics allows an instantiated EOFM to be automatically translated into a formal model [9] capable of being evaluated by a model checker, the Symbolic Analysis Laboratory (SAL) [10] in our case.

Operator	Description
optor_seq	Zero or more of the sub-acts must execute in any order one at a time.
optor_par	Zero or more of the sub-acts must execute in any order and can execute in parallel.
or_seq	One or more of the sub-acts must execute in any order one at a time.
or_par	One or more of the sub-acts must execute in any order and can execute in parallel.
and_seq	All of the sub-acts must execute in any order one at a time.
and_par	All of the sub-acts must execute in any order and can execute in parallel.
xor	Exactly one sub-act must execute.
ord	All sub-acts must execute in the order they appear.
sync	All sub-acts must execute synchronously.

Table 1. EOFM Decomposition Operators

MODELED RELATIONSHIPS

EOFMs are typically used to represent normative human behavior. However, it is possible to generate potentially unanticipated erroneous human behavior in instantiated EOFMs and thus include it in formal system models.

Erroneous behavior can be produced in two different ways. In the first [5], each action in an EOFM task is replaced with a generative that allows for the performance of Hollnagel's [11] zero-order phenotypes of erroneous action. Through multiple performances of zero-order phenotypes, more complicated erroneous behaviors are possible. In the second erroneous behavior generation technique [7], the

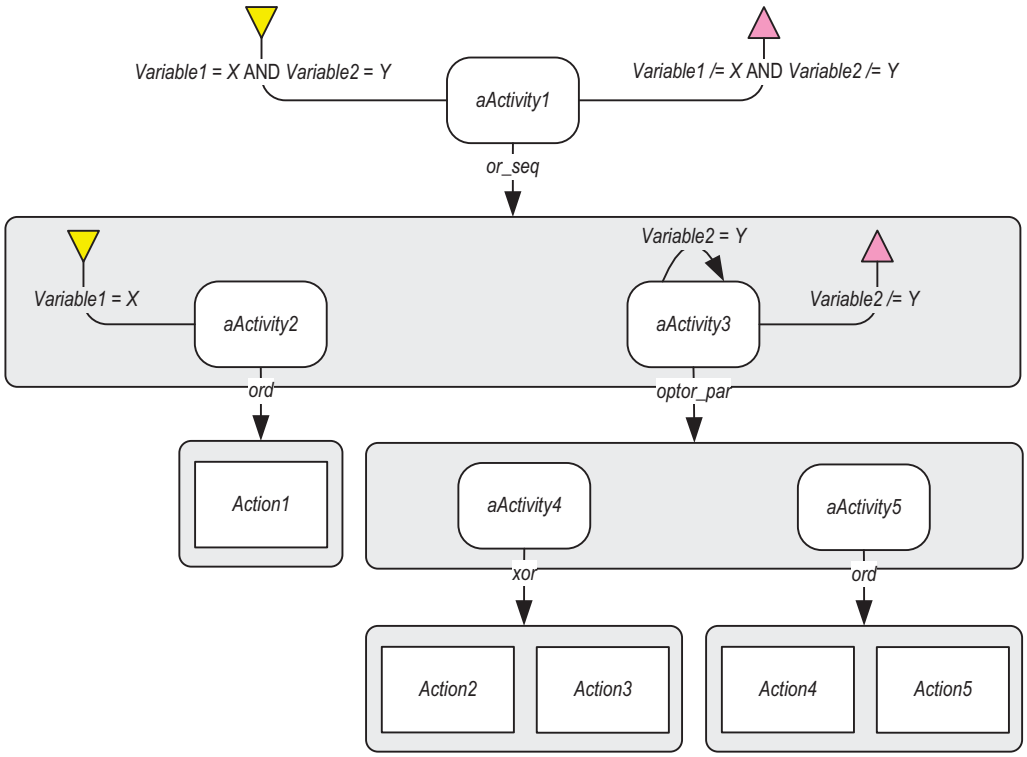


Figure 1. The visual representation of a task structure in an instantiation of an EOFM. EOFMs can be represented visually as tree-like graphs. Actions are rectangles and activities are rounded rectangles. An activity's decomposition is presented as an arrow, labeled with the decomposition operator, that points to a large rounded rectangle containing the decomposed activities or actions. Conditions on activities are represented as shapes or arrows (annotated with the logic) connected to the activity that they constrain. A precondition is a yellow, downward-pointing triangle; a completion condition is a magenta, upward-pointing triangle; and a repeat condition is an arrow recursively pointing to the top of the activity.

formal semantics of EOFM are extended in order to model human operator attention failures (Reason's [14] slips) that enable activities to be erroneously omitted, repeated, or committed. In both cases, a maximum is used to control the number of erroneous behaviors considered in a given evaluation. Both of these erroneous behavior generation techniques have been implemented as options in the EOFM to SAL translator.

Translated EOFM instances fit into a larger formal modeling architectural framework that supports concepts important to human-automation interaction. This encompasses models of human missions (i.e. goals), human task behavior (the translated EOFM instance), human-device interfaces (displays and controls available to the human operator), device automation (underlying device behavior), and the operational environment [4].

PROBLEMS ADDRESSED

The formal nature of the framework allows models created with it to be evaluated with a model checker. Model checking is an automated formal verification process that exhaustively searches a system's statespace to see if it can find a violation of specification properties. If no violation is found, the model checker has proven that the model adheres to the specification. Otherwise, the model checker produces a counterexample that illustrates how a violation occurred.

Thus, a model checker can be used on formal models that contain translated EOFM instances to prove whether or not the modeled human behavior, and the resulting human-automation interaction, will contribute to a violation of system safety (encoded in a specification property). This can include any generated (and thus potentially unanticipated) erroneous human behavior. System safety has been evaluated with normative human task behavior [9], normative human performance of checklist procedures [8], generated phenotypical erroneous human behavior [5], and generated erroneous human behavior caused by failures of attention [7]. It is also possible to use the visual notation of the EOFM to help analysts diagnose specification violations reported in counterexamples [6].

APPLICATIONS

EOFM has been used with model checking to find problems in, and explore design interventions for a number of applications. These include a Patient Controlled Analgesia Pump [4, 3, 7], an automobile with a simple cruise control [9], an aircraft instrument landing checklist procedure [8], and a radiation therapy machine [5].

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

There are a number of potential development opportunities for EOFM and its associated analyses. Firstly, analyses that utilize EOFMs, especially those with erroneous human behavior generation, do not scale well [2]. Thus current work is investigating ways of improving the EOFM to SAL translation in order to improve scalability. Secondly, the presented method has almost exclusively been used to evaluate single operator systems. However, ongoing research is investigating how to model human communication and coordination with EOFM [1]. Thirdly, in order to assist analysts in evaluating human-automation interaction with EOFM, work is currently investigating how specification properties indicative of good human-automation interaction can be generated automatically from instantiated EOFMs and used in formal verification analyses [3]. Finally, EOFM currently does not integrate with other infrastructures designed to evaluate human-automation interaction formally (see [2]). Future work should attempt to rectify this limitation.

ACKNOWLEDGEMENT

The project described was supported in part by Grant Number T15LM009462 from the National Library of Medicine (NLM), NASA Cooperative Agreement NCC1002043, and NASA award NNA10DE79C. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIA, NASA, the NLM, or the National Institutes of Health.

REFERENCES

1. Bass, E. J., Bolton, M. L., Feigh, K., Griffith, D., Gunter, E., Mansky, W., and Rushby, J. Toward a multi-method approach to formalizing human-automation interaction and human-human communications. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, IEEE (Piscataway, 2011), 1817–1824.
2. Bolton, M. L. *Using Task Analytic Behavior Modeling, Erroneous Human Behavior Generation, and Formal Methods to Evaluate the Role of Human-automation Interaction in System Failure*. PhD thesis, University of Virginia, Charlottesville, 2010.
3. Bolton, M. L. Validating human-device interfaces with model checking and temporal logic properties automatically generated from task analytic models. In *Proceedings of the 20th Behavior Representation in Modeling and Simulation Conference*, The BRIMS Society (Sundance, 2011), 130–137.
4. Bolton, M. L., and Bass, E. J. Formally verifying human-automation interaction as part of a system model: Limitations and tradeoffs. Innovations in *Systems and Software Engineering: A NASA Journal 6*, 3 (2010), 219–231.
5. Bolton, M. L., and Bass, E. J. Using task analytic models and phenotypes of erroneous human behavior to discover system failures using model checking. In *Proceedings of the 54th Annual Meeting of the Human Factors and Ergonomics Society*, Human Factors and Ergonomics Society (Santa Monica, 2010), 992–996.
6. Bolton, M. L., and Bass, E. J. Using task analytic models to visualize model checker counterexamples. In *Proceedings of the 2010 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE (Piscataway, 2010), 2069–2074.
7. Bolton, M. L., and Bass, E. J. Evaluating human-automation interaction using task analytic behavior models, strategic knowledge-based erroneous human behavior generation, and model checking. In *Proceedings of the IEEE International Conference on Systems Man and Cybernetics*, IEEE (Piscataway, 2011). in press.
8. Bolton, M. L., and Bass, E. J. Using model checking to explore checklist-guided pilot behavior. *International Journal of Aviation Psychology* (In Press).
9. Bolton, M. L., Siminiceanu, R. I., and Bass, E. J. A systematic approach to model checking human-automation interaction using task-analytic models. *IEEE Transactions on Systems, Man, and Cybernetics, Part A 41*, 5 (2011), 961–976.
10. De Moura, L., Owre, S., and Shankar, N. The SAL language manual. Tech. Rep. CSL-01-01, Computer Science Laboratory, SRI International, Menlo Park, 2003.
11. Hollnagel, E. The phenotype of erroneous actions. *International Journal of Man-Machine Studies 39*, 1 (1993), 1–32.
12. Kirwan, B., and Ainsworth, L. K. *A Guide to Task Analysis*. Taylor and Francis, London, 1992.
13. Mitchell, C. M., and Miller, R. A. A discrete control model of operator function: A methodology for information display design. *IEEE Transactions on Systems Man Cybernetics Part A: Systems and Humans 16*, 3 (1986), 343–357.
14. Reason, J. *Human Error*. Cambridge University Press, New York, 1990.

Position Paper: Modeling Human-Automation Interaction using Finite State Machines Formalism

Asaf Degani
General Motors R&D
Advanced Technical Center – Israel
asaf.degani@gm.com

ORIGIN AND UNDERLYING PRINCIPLES

The focus of the modeling framework is on a behavioral description of the system (i.e., its states, modes, and transitions), with special emphasis on user interaction and display feedback. The origin of the framework is Finite State Machine theory [18]. Parnas [13] was probably the first to use Finite State Machine models to describe user interactions with a computer terminal. This formalism enabled him to pinpoint several design errors such as "almost-alike" states, inconsistent ways to reach a state, and data entry problems. Foley and Wallace [5] also used this notation to describe their concept of a language of interaction between human and computer. Their state transition diagram describes the actions needed to make the transition from one state to another, as well as the system's response during the transition. Jacob [8] described several variants of the Finite State Machine, as well as other formalisms such as the Backus-Naur Form (BNF) [20] in the design of specifications for human-computer interaction with a communication system. He showed how such formalisms can be applied to a system with many commands. Since then, many researchers have used Finite State Machine theory to model user interactions [9,19] as well as a variety of newer extensions such as Petri Nets [12], OFM [11], and Statecharts [6].

A Finite State Machine model is a way of describing a system with its finite possible configurations— where the machine can be in any one of a finite number of configurations, or "states." The model has only finite input and output sets: it can respond only to the specified set of stimuli and produce only the specified set of behaviors [18, p. 232-235]. Finite State Machine theory captures the behavioral aspects of a system in a very precise and complete way; i.e., how it works and, in the context of human-machine interaction, how the system responds to user inputs and what information and feedback it provides to the user. The general structure of such a machine is described by state transitions of the following form: when event *alpha* occurs in state A, the system shifts to state B. The model can also be represented graphically as a state transition diagram that presents this behavioral information (states and transitions) as a perceptual code of nodes and arcs. This combination of theoretical and graphic formats has been used to represent human interaction with computer-based systems. For example, Figure 1 is a representation of the climate control system. Using statecharts language, a modern variant of the finite state machine formalism, it depicts the many (concurrent) components of the system such as the fan unit, compressor, air-source and air delivery units and zone control. As is in many human-machine systems, there is a clear hierarchy in the way the system modes and states are designed and this is also captured by the model. For example, there is an off/on mode but also manual, semi-manual, and fully manual modes to this system. The colored transitions, guards and labels are indicative of potential user interaction limitations and are there to alert designers to these problems [3].

MODELED RELATIONSHIPS

From a human factors and user interaction perspectives, the modeling framework is best suited to capture four types of relationships:

(1). The models describe and illustrate how external events trigger changes and reconfigurations in the system. For example, in an aircraft autopilot the angle of attack is a critical parameter that is constantly monitored. When the aircraft's angle of attack exceeds a given value, the aircraft may be entering stall. When this value is reached (around 15 degrees in most commercial airliners), the envelope protection system "kicks in" and will automatically advanced the throttles and/or reduce pitch attitude. In modeling autopilots systems, we pay utmost attention to such external events and their impact on the system [1, Ch. 15]. Along the same lines, we also describe important internal events such as timeouts (that may shut the system down), and outputs of internal computations (that may trigger a mode change). Naturally, we also focus on user initiated events such as global events (on/off switching), mode changes, reference values changes (e.g., aircraft altitude), and any other events that changes the state of the system and/or its interface.

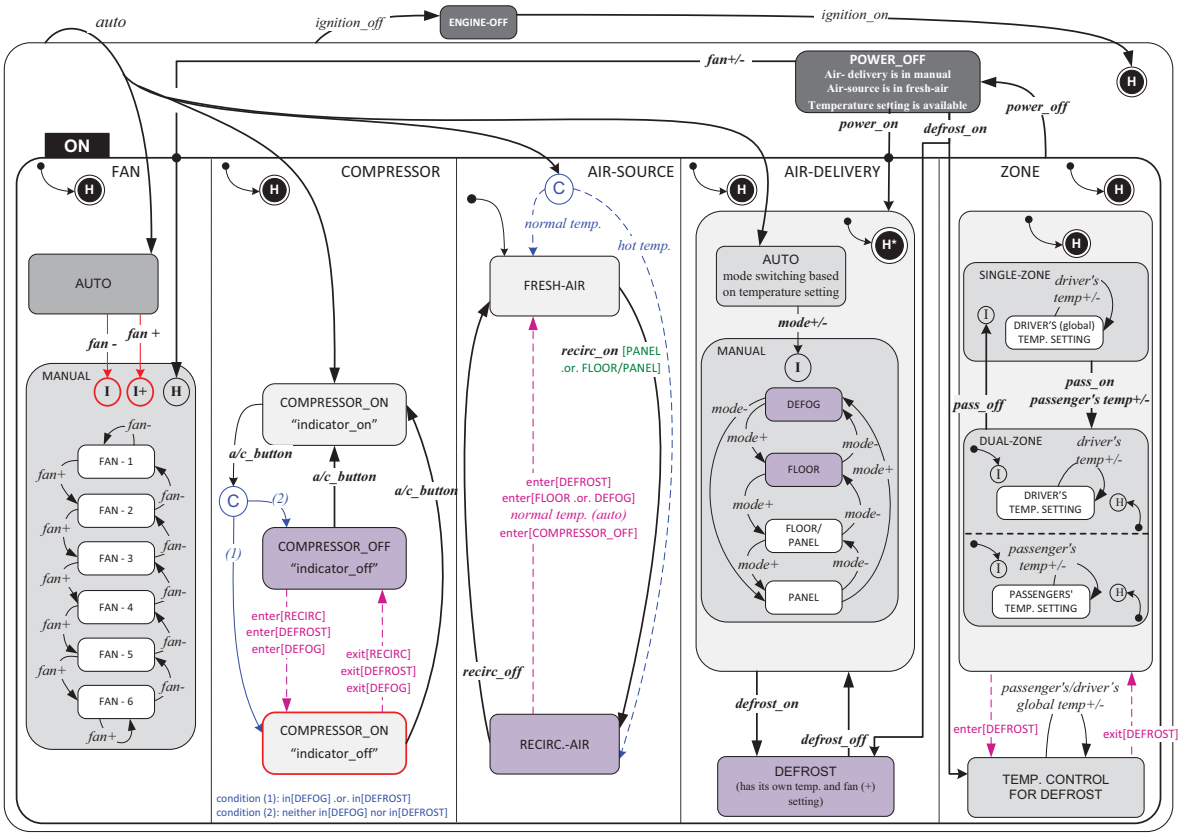


Figure 1: Statechart model of the climate control system. Broken lines (colored magenta) denote automatic transitions that are triggered either by internal dynamics or as side effects. Side effects occur when a given transition triggers another action (e.g., event) elsewhere in the system, or when the system enters or exits a specific state (also colored magenta). Conditional transitions are colored blue (as well as the condition itself). In situations where a transition is guarded (i.e., the transition only takes effect when the condition inside the block brackets is true,) the "guard" is colored green. Inconsistencies and discrepancies in the way the system behaves are outlined in red.

(2). We carefully consider relations between the different components of the system under consideration. For example, in Figure 1, consider the behavior of the AIR-SOURCE component. The model shows that there will be (an automatic – note the magenta broken line) state change from "recycled-air" to "fresh-air" when the driver switches the AIR DELIVERY mode to "defrost." Namely, a side effect here on the AIR-SOURCE due to changes that take place in another component (AIR DELIVERY). Generally speaking, we look for things like side effects, guards, and conditionals that are either within a given component or, in particular, those that affect other components in the system. Research on human-automation interaction had consistently shown that such coupling tends to confuse users [16,17] and are usually easily forgotten [4].

(3). The model helps the analyst to consider and evaluate the potential impact of external and internal events as well as side effect and guards on the user. (This, by the way, is the objective of the work and model described in Figure 1 – see Degani, Heymann, & Gellatly [3] for the full analysis). That is, we analyze the relations between system events and user understanding and performance. We can define different severity categories of such events on the user interaction and evaluate their potential impact on ease of use, elegance of interaction, and safety. We ask question like "will it cause confusion?" "Can and will users eventually understand system behavior?" And how much support is provided in the interface and in the user manual?

(4). Finally, we use to model to consider relations between machine model (that describes the behavior of the machine) and the interface model (a reduced and modified projection of the user model). Are all the important events in the machine model are projected to the user interface? Are there situations where the interface becomes non deterministic (error-states) or the interfaces blocks or (e.g., unnecessarily) augments the machine model [7]? In addition to the machine modes and interface model, it is possible to add also the user model where the user's "mental" model of the interface and machine is described [14]. Here we can account for situations where users forget (over time) how the machine works or simplifications and heuristics that people discover and employ (and evaluate their correctness).

PROBLEMS ADDRESSED

The idea is to describe the machine's behavior and all possible user interaction and display information as a way to help designers understand, evaluate, and formally specify the system. Another objective is to identify situations where the interface is incorrect [2], as well as situations where there are opportunities to improve (e.g., simplify) the interface [7]. There are a number of benefits from using a state-machine based formalism for this type of design approach: (1) it constitutes a clear description of the (interaction) design that enables review and discussion among multidisciplinary teams, (2) it articulates overarching design requirements as well as generic design patterns, (3) it uses a formal description for specifications, (4) it establishes a platform for analysis, heuristic or otherwise, of the design, and (5) it informs and supports the design of the graphical user interface (e.g., screen layout). Thus, the overall intent is to provide a formal approach to the design of human-machine interactions to improve not just the design but also the quality and rigor of the specifications. Quality means that the description is detailed and leaves nothing to interpretation or possible ambiguity. Rigor means that all system events and transitions are accounted for and described in the specifications [10].

APPLICATIONS

This approach has been used to discover and correct problems in avionics [1,2], automobile interfaces[2,7], and a number of consumer electronics[1].

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

First and foremost, the presented framework focuses on the system or machine under consideration. Therefore it does not address many human factors issues such as the cognition, decision making, perception, physical limitations. In terms of the modeling of the machine and user interaction, this modeling framework requires a thorough engineering understanding of the system as well as technical savvy in system analysis and verification techniques. The analysis of the model is only as good as the properties that are at the disposal of the analyst. As it stands now, we have only a limited set of properties to verify in a given system. Last but not least, like most modeling frameworks it is negatively affected by the level of abstraction used to describe the system. Naturally if the level of abstraction is high (overly simplified description) the results may be incomplete.

REFERENCES

1. Degani, A. *Taming HAL: Designing interfaces beyond 2001*. New York: Palgrave- MacMillan, 2004.
2. Degani, A., and Heymann, M. Formal verification of human-automation interaction. *Human Factors*, 44,1(2002), 28–43.
3. Degani, A., Heymann, M., and Gellatly, A. HMI aspects of automotive climate control systems. *Proceedings of the 2011 IEEE Systems, Men and Cybernetics Conference,* IEEE (2001).
4. Crow, J. Javaux, D. and Rushby, J.. *International Conference on Human-Computer Interaction in Aeronautics*, AAAI (2000).
5. Foley, J. D. and Wallace, V. L. The art of natural graphic man-machine conversation. *IEEE Transactions on Software Engineering*, *62* (1974), 462-471.
6. Harel, D. Statecharts: A visual formalism for complex systems. *Science of Computer Programming, 8* (1987), 231-274.
7. Heymann, M. and Degani, A. Formal analysis and automatic generation of user interfaces: Approach, methodology, and an algorithm. *Human Factors*, *49, 2*, (2007), 311-330.
8. Jacob, R. J. K. Using formal specifications in the design of human-computer interfaces. *Communications of the ACM*, *26, 4* (1983), 259-264.
9. Kieras, D.E., and Polson, P.G. An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies*, *22* (1985), 365-394.
10. Leveson, N. Safeware. *System Safety and Computers*. New York: Addison-Wesley, 1995.
11. Mitchell, C. M. and Miller, R. A. A Discrete Control Model of Operator Function: A Methodology for Information Display Design. *IEEE Transactions on Systems, Man and Cybernetics, 16, 3 (1986)*, 43-357.
12. Palanque, P., and Bastide, R. A design life-cycle for the formal design of user interface. In *Proceeding of the BCS-FACS Workshop on the Formal Aspects of Human Computer Interaction*, (Shefield, U.K., 1996)
13. Parnas, D. On the use of transition diagrams in the design of a user interface for an interactive computer system. In *Proceedings of the 24th Annual ACM Conference,* ACM (1969), 379-385.
14. Romera, M. *Using Finite Automata to Represent Mental Models*. Unpublished Masters Thesis. San Jose, California: San Jose State University (2000).
15. Rothrock, L., and Kirlik, A. Inferring rule-based strategies in dynamic judgment tasks: Toward a noncompensatory formulation of the lens model. *IEEE Transactions on Systems Man and Cybernetics, Part A-Systems and Humans*, *33,1* (2003), 58-72.
16. Sarter, N. B., & Woods, D. D. Pilot interaction with cockpit automation II: An experimental study of pilot's mental model and awareness of the flight management and guidance system. *International Journal of Aviation Psychology, 4, 1* (1994), 1-28.
17. Sarter, N. B., and Woods, D. D. How in the world did we ever get into that mode? Mode error and awareness in supervisory control. *Human Factors*, *37, 1* (1995), 5-20.

18. Turing, A. M. On computable numbers with an application to the Entscheidungsproblem. *Proeedings of the London Math. Soiety.*, *42, 2* (1936), 230--265.

19. Wasserman, A.I. Extending state transition diagrams for the specification of human--computer interaction. *IEEE Transaction on Software Engineering. 8* (1985), 699-713.

20. Woods, W. A. Transition network grammars for natural language analysis, *Communications of the ACM*, *13, 10* (1970), 591-606.

Position Paper: Modeling Multiple Human-Automation Distributed Systems using Network-form Games

Guillaume Brat
NASA
Ames Research Center - USA
guillaume.p.brat@nasa.gov

ORIGIN AND UNDERLYING PRINCIPLES

The focus of the modeling framework is on interactions between human and automation agents in large, distributed agent networks/systems. This model combines Bayes nets with Game Theoretic methods to model human behavior and predict the behavior of a composite system involving humans and automation. In general, some of the nodes of the Bayes net will be set by the humans in the system, some will be set with known conditional distributions (e.g., noise models of sensors), and some might be "black boxes" provided by the proposer that simulate behavior of automated devices. Novel algorithms are required for sampling and prediction with this model.

Bayes nets have been widely studied to describe stochastic systems [1-3]. A Bayes net is a directed acyclic graph in which nodes represent random variables and edges represent conditional dependencies between these variables. The variables can be observable quantities or unknown quantities (or hypotheses). An edge between two nodes indicates that the random variables represented by the nodes are conditionally independent of each other. Each node is assigned a probability function that takes as inputs the random variables of the parent node and that gives the probability assigned by to the random variable associated with the node. There exist many algorithms to calculate the interference and learning in Bayesian networks.

Game Theory is also a well-know technique, which has been used to describe the behavior of interacting humans [4, 5]. It has been widely studied in economics contexts to represent human behaviors and study how decisions are made in auctions and negotiations for examples. The field first addressed zero sum games (so that gains and losses between participants are perfectly balanced), but it has evolved beyond that and can now study different models of equilibrium (Nash equilibrium, Quantal Response equilibrium, Quantal Level-K and Cognitive Hierarchy).

Wolpert has combined Bayes nets and Game Theory in a novel framework, called semi network-form games, to model systems in which humans interact with other humans and with automation. The semi network-form game is a specialization of the complete framework "network-form games" formally defined in [6, 7] by Wolpert. Currently it is relying on level-K equilibrium.

In a semi network-form game, a Bayes net is used to describe probabilistic interactions between agents (humans or automation) in a system using random variables. Automation (and physical sub-systems) is represented by a "chance" node while a human is represented by a "decision" node. The conditional probability distributions associated with "chance" nodes are pre-specified. The "decision" nodes also differ in the sense that they are associated with a utility function, which maps an instantiation of the net to a real number quantifying the player's utility. Utility functions are used to encode the goals of a player. In other words, it represents what a human tries to optimize during the game. A semi network-form game allows a player to control only one decision node while a complete network-form game make no such restriction allowing a player to control multiple decision nodes in the net. Network-form games bear a resemblance to Multi-Agent Influence diagrams [8], except that network-form games consider bounded rational agents and uses utility functions rather than utility nodes.

We illustrate the use of network-form games with the example of a 2-aircraft mid-air encounter (similar to the infamous Überlingen accident). The corresponding Bayes net is shown in Figure 2. At time t, the system is represented by a layer of observation (of the world state) nodes (for both pilots and TCAS boxes), a layer of TCAS nodes, a layer of pilot nodes, the world state as an input node and an outcome node. The state of the pilot node is influenced by both the pilot's observations and the TCAS outcome. The final outcome state is calculated by simulating the aircraft states forward in time using a model of the aircraft kinematics. The social welfare of the system is then calculated from the outcome state. The observational layer is necessary to model observational noise and incomplete information resulting from pilots and TCAS imperfectly observing the world state.

MODELED RELATIONSHIPS

The relationships modeled with network-form games are different from traditional techniques in human factors and user interaction perspectives. The modeling framework is best suited to capture the following types of relationships:

(1). The model strength resides in its ability to capture non-deterministic pilot behavior. For example, TCAS assumes that a pilot receiving an RA will delay for 5 seconds and accelerate at ¼ g to execute the RA maneuver. Despite their

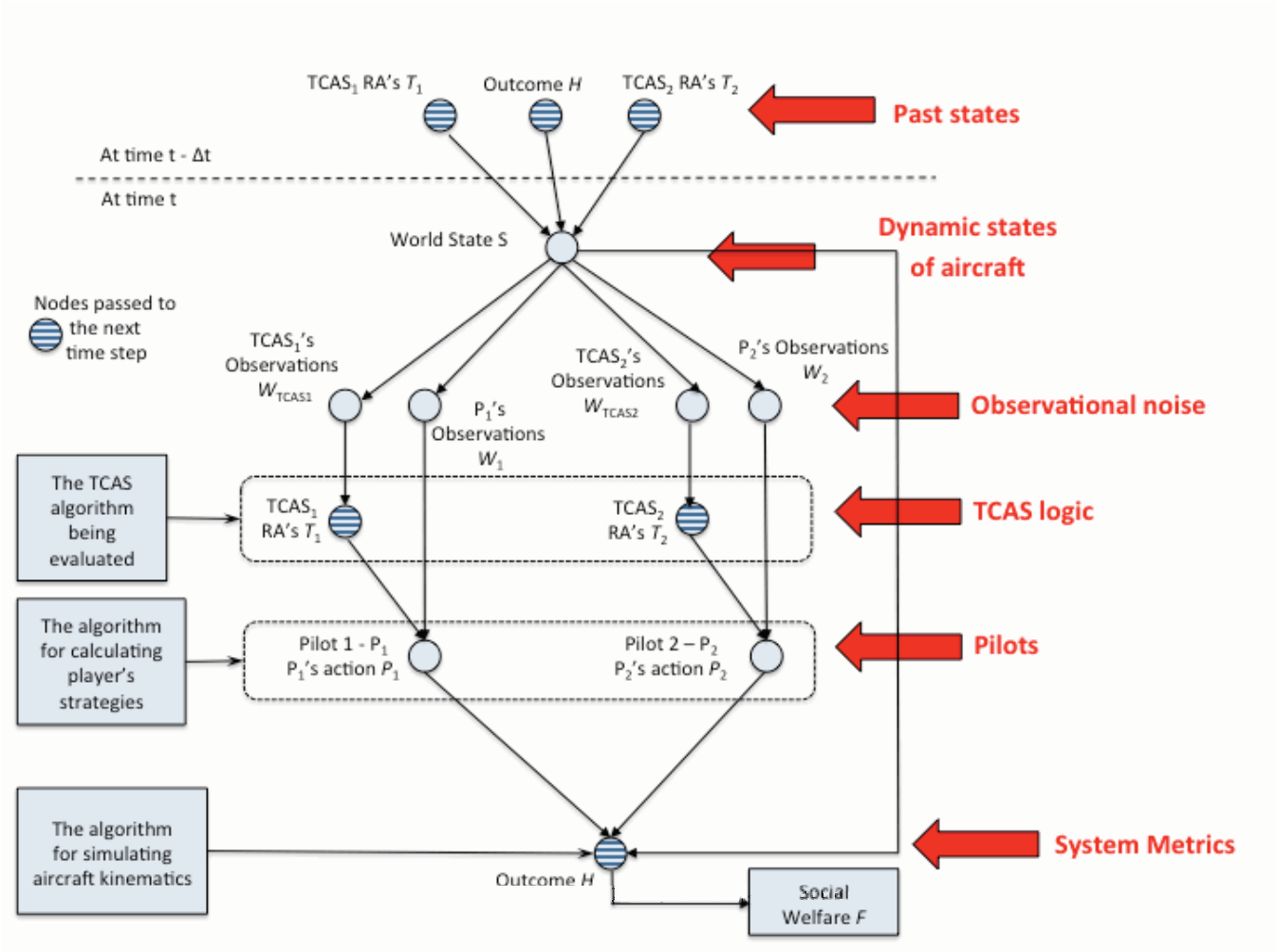


Figure 1. A Bayes net for a 2-aircraft mid-air collision example.

training, pilots actually have different reactions to TCAS RAs. A recent study [9] in the Boston area has found that

- 13% of RAs are obeyed in compliance
- 64% of RAs are obeyed in partial compliance, i.e., the aircraft is moved in the right direction but with an improper timing, and,
- 23% of RAs are ignored and the aircraft is moved in the opposite direction.

Clearly, pilots are not always responding in the same way to TCAS RAs. In fact, pilots make up their mind using more than just the TCAS information, taking into accounts other sources of information, including their own visual clues. The network-form game framework is able to capture this non-determinism by using probability distributions.

(2). The included game theory framework is also very useful to model the "gamesmanship", or guessing game, that may happening when humans interact with other humans. In a mid-air collision possibility, it is important to model the fact that a pilot is always wondering if the other pilot is going to react according to the training he received. Will the other person/pilot make the right move? What if he doesn't? What is my back-up strategy? When do I need to decide which strategy to follow? These types of questions are best answered in a game theory framework.

PROBLEMS ADDRESSED

The idea is to describe human and machine interactions within a large multi-agent system, e.g., airplane crew interacting with air traffic controllers and automation such as TCAS or ADS-B. From a safety point of view, problems can occur because of misunderstanding between

- humans, e.g., a pilot misunderstanding the orders issued by a controller, or, the pilots of two planes encroaching on their respective runways in order to optimize their on-time gate arrival time, or,
- humans and automation, e.g., a pilot doubting, or misunderstanding, the outputs of an automated box, or, a pilot following a TCAS advise when the controller is actually issuing a contradictory command.

The framework can be used at different level of granularity. It works for modeling human/machine interaction problems as well as new air traffic concepts of operation.

An interesting aspect of network-form games is its ability to model the fact that a human might reason about what another human is thinking of doing. Basically, the framework can explore how human reasons about the possible moves of an opponent. In the aeronautics case, one can model how pilots modify their actions based on the actions of another pilot on another plane, e.g., a plane on a collision course. One can model situations where the pilot is weighing his options based on his thinking that the pilot on the other plane is actually paying attention to a TCAS box or not. This can potentially affect his own reactions towards what his own TACS box is advising him to do.

In network-form games, this type of reasoning is captured by an equilibrium concept such as level-K thinking, which is defined recursively as follows. A level-K player plays as if all the other players are playing at level K-1. These players are playing in turn as if others at playing at level K-2, and so on until level 0 is reached, where the players play according to a known prior distribution. So, if we have two players A and B and K=2 and player A is a level 2 player, A plays as if Player B is a level 1 player that assumes that A plays a level 0 player. Note that those are assumptions made by player A. Player B might in reality be a level 2 player, not a level 1 who thinks A is level 0.

Now, this feature is also important when the automation is actually closer to autonomy than automation. Here we are using autonomy to describe situations in which control is not exercised by humans but by a computer of an algorithm with a certain degree of "intelligence". This is the case in Aeronautics when UAS (Un-piloted Aerial Systems) are operating autonomously in the National Airspace System and freely mix with piloted planes. The reasoning is not about another human, but about a system capable of fairly complex reasoning.

APPLICATIONS

This approach is also being used to discover and correct problems in cyber-security (cyber physical attack on smart power grid, denial of service).

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

The presented framework focuses on human factors issues such as decision making and (in some ways) perception. Issues such a cognition, and, physical limitations are hard to model. At this stage, the biggest limitation is that the analysis relies on having valid probability distributions for human behavior. The best solution would be to get access to results of high-fidelity, human-in-the-loop studies done by the FAA. However, getting access is difficult, and, the number of simulations in those studies does not lend to estimating statistically-valid probability distributions. This problem is being currently addressed by studying the possibility of using multi-fidelity simulations.

REFERENCES

1. Bishop, C.M., "Pattern recognition and machine learning," Springer 2006.
2. Darwiche, A., "Modeling and reasoning with Bayesian networks," Cambridge University Press, 2009.
3. Russel, S., Norvig, P., "Artificial intelligence a modern approach," 2nd ed. Pearson Education, 2003.
4. Crawford, V.P., "Introduction to experimental game theory, " Journal of Economic Theory, 104, pp. 1-15, 2002.
5. Crawford, V.P., "Modeling behavior in novel strategic situations via level-k thinking," In Third World Congress of Game Theory Society (GAMES), 2008.
6. Lee, R., Wolpert, D., "Game theoretic modeling of pilot behavior during mid-air encounters," Chapter 4 in Decision Making with Imperfect Decision Makers, ed. By T.V. Guy, M. Karny, and D. Wolpert, Springer, 2012.
7. Wolpert, D., Lee, R., "Network-form games: using Bayesian networks to represent non-cooperative games," NASA Ames Research Center working paper, Moffett Field, California.
8. Koller, D., Milch, B., "Multi-agent influence diagrams for representing and solving games," Games and Economic Behavior, 45(1), pp. 181-221, 2003.
9. Kushar, J.K., Drumm, A.C., "The Traffic and Collision Avoidance System," Lincoln Laboratory Journal, 16(2), 2007.

Decision Makers' Preference Capture in Human-Machine Interactions

Ignacy Kaliszewski

Warsaw School of Information Technology
ul. Newelska 6
01-447 Warszawa, Poland
+48 22 38 10 392
kaliszew@wit.edu.pl

Janusz Miroforidis

Warsaw School of Information Technology
ul. Newelska 6
01-447 Warszawa, Poland
Treeffect Co
Gdów 1028
32-420 Gdów, Poland
janusz.miroforidis@treeffect.com

ORIGIN AND UNDERLYING PRINCIPLES

In this paper, we present a method to capture decision maker's preferences in *multiobjective problems* and we discuss its use as a base for a decision maker – *multiobjective optimization problem* (model) interface.

In principle, all decision problems are multiobjective problems, i.e. each problem involves at least two objective. The usual approach to deal with multiple objective is to optimize the problem with respect to a selected single objective and observe the values of other objectives.

However, there exists a methodology which allows to address multiobjective optimization problems directly ([5,8,9,15,19], where other pertaining references are also given). Moreover, this methodology provides for an easy and intuitive capture of decision maker's preferences and allows in turn determining solutions which correspond to those preferences best. In other words, the methodology provides for an easy (to understand, command and implement) and intuitive interfacing multiobjective optimisation problems (models) to the decision maker.

MODELED RELATIONSHIPS

Let x denote a (decision) *variant* (solution), X a space of variants, X_0 a set of *feasible variants*, $X_0 \subseteq X$. Then the multiobjective optimisation problem is:

$$\text{''vmax''}\ f(x) \tag{7}$$

$$x \in X_0\ ,$$

where $f: X \to R^k$, $f = (f_1, \dots f_k)$, $f_i: X \to R$, $i = 1, \dots, k$, $k \geq 2$, are *objective functions* (*criteria*); ''vmax'' denotes the operator of deriving all *efficient* (as defined below) *variants* in X_0.

Variant $\bar{x}$ of X_0, is *efficient*, if $f_i(x) \geq f_i(\bar{x})$, $i = 1, \dots, k$, $x \in X$, implies $f(x) = f(\bar{x})$.

It is a well established result ([2,4,6]) that variant $\bar{x}$ is efficient[1] if and only if it solves the optimisation problem

$$min_{x \in X_0} max_i \lambda_i \left(y_i^* - f_i(x)\right), \tag{8}$$

where $\lambda_i > 0,\ i = 1, \dots, k$, and y^* is such that $y_i^* > f_i(x)$, $i = 1, \dots, k,\ x \in X_0$.

By the "only if" part of this result no efficient variant is excluded from being derived by solving an instance of optimisation problem (8). In contrast to that, maximisation of a weighted sum of objective functions over X_0 does not possess, in general (and especially in the case of problems with discrete variables), this property.

Besides the potential ability to derive each efficient variant, optimisation problem (8) provides for an easy and intuitive capture of decision maker's preferences. Observe that element $\hat{y}$, where $\hat{y}_i = max_{x \in X_0} f_i(x),\ i = 1, \dots, k$, represents maximal values of objective functions which can be attained if they are maximised separately.

To assists the decision maker in the search for *the most preferred variant* one can employ the optimisation problem (8). Here we assume the minimum of the decision maker rationality, namely we assume that the decision maker prefers an efficient variant to a non-efficient one.

Suppose that an element $x \in X_0$ such that $\hat{y} = f(x)$ does not exists which is rather a standard with conflicting criteria (if otherwise, x is the most preferred variant. Then, the decision maker knows that whatever efficient variant he (or she) selects he has to compromise on values of objective functions f_i with respect to values $\hat{y}_i$, $i = 1, \dots, k$. He can define his acceptable compromises on values $\hat{y}_i$, $i = 1, \dots, k$, and search for an efficient variant which corresponds to this compromise in three ways:

[1] Actually, variant $\bar{x}$ is *weakly efficient* but for the sake of conciseness we do not make this distinction here, for a formal treatment of this issue cf. [5,9,15].

1. providing a *vector of concessions* τ ,
2. providing a reference point y^{ref},
3. providing weights λ_i , $i = 1, \dots, k$.

Way 1. Components of a vector of concessions τ ($\tau \in R^k$) specify concessions the decision maker accepts to make with respect to $\hat{y}_i$, $i = 1, \dots, k$. Components of vector τ can be defined in absolute values ("the decision maker is willing to make a concession of z_i units on the value $\hat{y}_i$, $i = 1, \dots, k$") or in relative values ("the decision maker is willing to make a concession of z_i per cent on the value $\hat{y}_i$, $i = 1, \dots, k$").

Way 2. A reference point y^{ref} ($y^{ref} \in R^k$, $y_i^{ref} \leq \hat{y}_i$, $i = 1, \dots, k$), (it is irrelevant whether there exists an element $x \in X_0$ such that $y^{ref} = f(x)$ or not) specifies explicitly a compromise between values of objective functions f_i with respect to values $\hat{y}_i$, $i = 1, \dots, k$, which the decision maker regards as agreeable. A reference point specifies indirectly a vector of concessions:

$$\tau_i = \hat{y}_i - y_i^{ref}, \; i = 1, \dots, k \; . \tag{9}$$

Way 3. An experienced decision maker can define a vector of concessions τ in terms of weights $\lambda_i > 0$, $i = 1, \dots, k$, in optimisation problem (8). Vector of concessions τ and vector of weights λ are related by formula (10).

The optimisation problem (8) if solved with

$$\lambda_i = (\tau_i)^{-1} , \; i = 1, \dots, k, \tag{10}$$

has the following property;

- it finds an efficient variant x such that $f(x)$ is on half line $y = \hat{y} - t\tau, \; t \geq 0$, whenever such a variant exists,
- otherwise, it finds an efficient variant x such that $max_i \lambda_i (\hat{y}_i - f_i(x)) = max_i \lambda_i (\hat{y}_i - \tilde{y}_i)$, where $\tilde{y}$ is on half line $y = \hat{y} - t\tau, \; t \geq 0$.

To avoid dividing by zero in formula (10), in formula (9) and in all pertaining considerations we are to replace $\hat{y}$ by y^*, but since in the definition of y^* ($y_i^* = \hat{y}_i + \varepsilon, \; \varepsilon > 0$, $i = 1, \dots, k$,) ε can be taken arbitrarily small, so the difference between $\hat{y}$ and y^* can be made insignificant.

PROBLEMS ADDRESSED AND APPLICATIONS

Any problem for which (7) is the underlying formal model can be treated along lines as outlined above.

Spectacular examples of topics framed as multiobjective optimisation problems are nuclear weapons [20], energy [6], medicine [14], finance [17], civil engineering [18], electromagnetic engineering [4], air transport [12].

LIMITATIONS AND DEVELOPMENT OPPORTUNITIES

The need to solve series of optimisation problems (one optimisation problem for each vector of concession τ) is a barrier for a broad use of the method for decision maker preference capture as outlined above.

An attempt to overcome this barrier is the concept, introduced by Kaliszewski ([7], [9], [10]), of deriving efficient variants by approximate calculations with controlled accuracy. This, with applications to practical multiobjective problems kept mind, is a rational approach. The necessity of solving optimisation problems in multiobjective problems to derive efficient variants in this concept is eliminated for the price of the necessity to derive a subset of efficient variants (still by solving optimisation problems but) prior to starting multiobjective problem solving. This allows multiobjective problem solving to be decomposed into the initial (technical) phase, which can be realised without involving the decision maker, and the (proper) decision phase, where there is no longer any need to solve optimisation problems.

The core of this concept is calculation – during the decision phase - of lower and upper bounds on component values of *outcome* $f(x)$ for efficient variant x (i.e. on values of objective functions $f_i(x)$, $i = 1, \dots, k$) selected from the set of feasible variants X_0 instead of deriving outcome $f(x)$ itself. For that purpose a number of efficient variants, forming so called *shell*, has to be derived, which as said before, can be done in the initial phase.

An efficient variant x is selected implicitly by the decision maker when selecting an instance of so called *vector of concessions* τ , which is a parameter of the bounds.

An instance of the vector of concessions defines the direction of searching for an efficient outcome, starting from the utopia point, i.e. a point in the outcome space, which dominates all outcomes of efficient variants from set X_0 .

With given shell S and selected instance of vector τ, this concept provides for assessments of values $f_i(x)$ in the form:

$$L_i(\tau, S) \leq f_i(x), \leq U_i(\tau, S), \; i = 1, \dots, k, \tag{3}$$

where x denotes an efficient variant corresponding to the preferences of the decision maker, represented by vector of concessions τ, and $L_i(\tau, S)$ and $U_i(\tau, S)$ are formulae for a lower bound and an upper bound on $f_i(x)$, respectively ([7],[9],[10]). With given shell S vector of concessions τ is the parameter of the bounds.

In the multiple criteria decision making process the decision maker evaluates variants x via its outcomes $f(x)$. The essence of the concept ([7],[9],[10]) has been to evaluate variants x selected by selecting instances of the vector of concessions τ via approximations of outcomes $f(x)$ in the form of lower and upper bounds on outcome component values, as in (3). Once the most preferred outcome is identified – and only then - the corresponding variant is made explicitly available.

The existence of formulae for lower and upper bounds on component values of outcomes $f(x)$ for efficient variants x allows to control the accuracy of approximations of those values. However, the necessity of using optimisation methods in the initial phase of the multiobjective decision processes is still a barrier for a broad popularization of multiple criteria decision making methodologies and decision support systems which implement such methodologies.

That barrier has been overcome by Miroforidis [16] (for further developments cf. also [11],[13]) by replacing shell S by a pair of shells: lower shell S_L (consisting of elements of set X_0) and upper shell S_U (consisting of elements of set $X \setminus X_0$), which can be derived by approximate computations in such a manner that upper and lower bounds on component values of outcomes can be still calculated.

The formulae for calculation of $L_i(\tau, S_L)$, and $U_i(\tau, S_U)$ (as by Miroforidis [16]) have the same degree of complexity as the formulae for calculation of $L_i(\tau, S)$, and $U_i(\tau, S)$ (as by Kaliszewski ([7],[9],[10]). It is worth noting that all those formulae consist only of arithmetic operations and operations of derivation of maximal values on finite sets of numbers, which allows, after deriving shell S or pair: lower shells S_L and upper shell S_U, for their easy implementation e.g. in a spreadsheet.

Replacing shells S by pairs of lower shells S_L and upper shells S_U and derivation of formulas similar in the nature to formulas for lower and upper bounds with shells S, has allowed elaboration of a new concept of employing evolutionary computations in multiobjective optimisation, namely the concept which relies on derivation of shells S_L and S_U by evolutionary algorithms ([16],[11],[13]). Within this new concept algorithms to derive pairs of lower shells S_L and upper shells S_U simultaneously have been proposed [16],[11],[13]).

Making use of approximate computations with controlled accuracy, as described above, allows to remove completely the necessity to employ exact (i.e. classical) optimisation algorithms in multiobjective optimisation and in consequence to remove the aforementioned barriers for wide popularization of decision support systems.

Potential applicability of evolutionary computations to multiobjective optimisation has been convincingly documented in the literature ([1],[2]) and practically verified on numerous applications. However, the idea of preference driven evolutionary computations in the multiple criteria decision making context is quite recent ([3],[13]).

REFERENCES

1. Chang, W.A. Advances in Evolutionary Algorithms. *Theory, Design and Practice*, Springer, 2006.
2. Coello Coello, C.A., Lament, G.A. and Van Veldhuizen D.A. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, 2007.
3. Deb, K. and Köksalan, M. (eds). Preference-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computations*, 14, 2010.
4. Di Barba, P. *Multiobjective Shape Design in Electricity and Magnetism*. Springer, 2010.
5. Ehrgott, M. *Multicriteria Optimization*. Springer, 2005.
6. Hobbs, B.F. and Meier, P. *Energy Decisions, and the Environment: A Guide to the Use of Multicriteria Methods*. Kluwer Academic Publishers, 2000.
7. Kaliszewski, I. Dynamic parametric bounds of efficient outcomes in interactive multiple criteria decision making problems. *European Journal of Operational Research*, 147, 2003, 94-107.
8. Kaliszewski, I. Out of the mist – towards decision-maker-friendly Multiple Criteria Decision Making support. *European Journal of Operational Research 158*, 2004, 93–307.
9. Kaliszewski, I. *Soft Computing for Complex Multiple Criteria Decision Making*, Springer, 2006.
10. Kaliszewski, I. A method of approximating Pareto sets for assessments of implicit Pareto set elements. *Control & Cybernetics*, 2007, 367-381.
11. Kaliszewski, I. and Miroforidis, J. Multiple Criteria Decision Making: efficient outcome assessments with evolutionary optimisation. *Communications in Computer and Information Science*, 35, 2009, 25-28.
12. Kaliszewski, I. and Miroforidis, J. On interfacing multiobjective optimisation models – the case of the airport gate assignment problem. The paper to be presented at ATACCS'2012, 29-31 May 2012, London, UK.
13. Kaliszewski, I., Miroforidis, J. and Podkopaev, D. Interactive Multiple Criteria Decision Making based on preference driven Evolutionary Multiobjective Optimisation with controllable accuracy. *European Journal of Operational Research*, 216, 2012, 188–199.
14. Küfer, KH., Scherrer, A., Monz, MP., Alonso, FV., Trinkaus, H., Bortfeld, TR. and Thieke, C. Intensity modulated radiotherapy - a large scale multicriteria programming problem. *OR Spectrum*, 25, 2003, 223-249.
15. Miettinen, K.M. *Nonlinear Multiobjective Optimisation*. Kluwer Academic Publishers, 1999.
16. Miroforidis, J. Computer aided decision making in department stores with multiobjective optimisation and approximate methods. PhD Thesis, Systems Research Institute, Polish Academy of Sciences, 2010.
17. Spronk, J., Steuer, R.E. and Zopounidis, C. Multicriteria decision aid/analysis in finance. In: *Muliple Criteria Decision Analysis: State of The Art Surveys*, J. Figurera, S. Greco, M. Ehrgott (eds), Springer, 2005, 799-857.
18. Yoon, M., Yun, Y. and Nakayama, H. *Sequential Approximate Multiobjective Optimization Using Computational Intelligence*. Ser. Vector Optimization, Springer, 2009.

19. Wierzbicki, A.P., Reference point approaches. In: *Multicriteria Decision Making - Advances in MCDM: Models, Algorithms, Theory and Applications*, Gal, T., Stewart, Th. and Hanne, Th. (eds). Kluwer Academic Publishers, 1999.

20. von Winterfeldt, D. and Schweitzer, E. An assessment of tritium supply alternatives in support of the U.S. nuclear weapons stockpile. I*nterfaces*, 22, 1998, 92-118.