A Methodology for Analyzing Human-Automation Interactions in Flight Operations Using Formal Verification Techniques [1]

Denis Javaux
Symbio Concepts & Products
4690 Bassenge, Belgium
denis.javaux@symbio.pro

Bertram Wortelen
OFFIS Institute for Information Technology
26121 Oldenburg, Germany
bertram.wortelen@offis.de

Andreas Lüdtke
OFFIS Institute for Information Technology
26121 Oldenburg, Germany
luedtke@offis.de

Charles Pecheur
Université catholique de Louvain
1348 Louvain-La-Neuve, Belgium
charles.pecheur@uclouvain.be

Regina Peldszus
European Space Operations Centre (ESOC)
64293 Darmstadt, Germany
regina.peldszus@esa.int

Sonja Sievi
Astrium GmbH
28199 Bremen, Germany
sonja.sievi@astrium.eads.net

Yuri Yushtein
European Space Research and Technology Center (ESTEC)
2200AG Noordwijk, The Netherlands
yuri.yushtein@esa.int

Abstract

When designing and developing systems in safety critical or cost intensive environments it is important to identify as much potential risks as possible prior to operating the system. This includes aspects of the interaction between human and automation systems that are prone to issues. This work-in-progress paper describes a methodology that systematically derives relevant analysis questions for complex human-automation interaction systems. It demonstrates how formal models for all components of the human-automation system can be created. These models are used by model checking algorithms to verify the safety properties associated with the selected analysis questions. While this paper includes no evaluation of the methodology, an ongoing evaluation study is outlined based on the life support system (ECLS) of the European science laboratory Columbus, which is part of the International Space Station. Each step of the formal verification methodology is illustrated with the results obtained so far on the ECLS case study.

Objective

Development of critical systems mostly relies on the stringent Development Processes to ensure the targeted system quality and integrity levels. While approaching the sought after confidence level from the developers and the end users perspective, the process-based assurance cannot conclusively demonstrate the system correctness and consistency with the envisaged usage. For this reason rigorous formal techniques are finding their way into the critical systems development domain (MODUK 00-55, IEC 61508, ISO 26262, DO-178C/ED-12C, DO-333/ED-218). These approaches focus on the system design aspects, such as correctness and consistency of the system being developed with respect to the requirements specifications.

However, critical systems that are meant to be actively operated by humans include system operators in the global scope of the deployed system. This implies additional perspectives for the notions of system operational correctness and consistency: namely, the behavior of the operator(s), their interaction with the systems and its automation, impact success of the systems operation in the context of operational objectives. This is especially so in the cases of shared authority over systems operation between the operator(s) and the systems automation elements. This necessitates including the human operators and their interaction with the automated systems into the scope of application of the formal verification techniques. Next to system modeling, this requires the creation of human operator behavioral models, models of the automated systems external control, as well as models of the human-automation interaction, including possible human errors and the facts attributing to them (Dix 2013).

Related human-automation co-operation issues include the mismatch between the systems design assumptions regarding the operators' performance; operator interfaces inadequacy with respect to the systems status

[1] This work was carried out under a programme of and funded by the European Space Agency. The view expressed herein can in no way be taken to reflect the official opinion of the European Space Agency.

comprehension or expected operator responses, which can lead to mode confusion and thus, to errors of commission or omission; elements of and factors in the operational environment affecting the operators performance of the corresponding tasks (Sarter, Woods and Billings 1997).

In order to allow for a coherent inclusion of the above mentioned elements (models) in the overall human-automation system model, suitable formalisms are needed, which are, on the one hand, sufficiently expressive to cover the domain of relevance, and, on the other, amenable to formal verification techniques. Given the heterogeneous domains to be modeled and verified in a global system operation scope, the formal modeling and verification techniques shall be accompanied by a methodology that provides a cohesive framework for specification, modeling and analysis of the critical operator-in-the-loop systems.

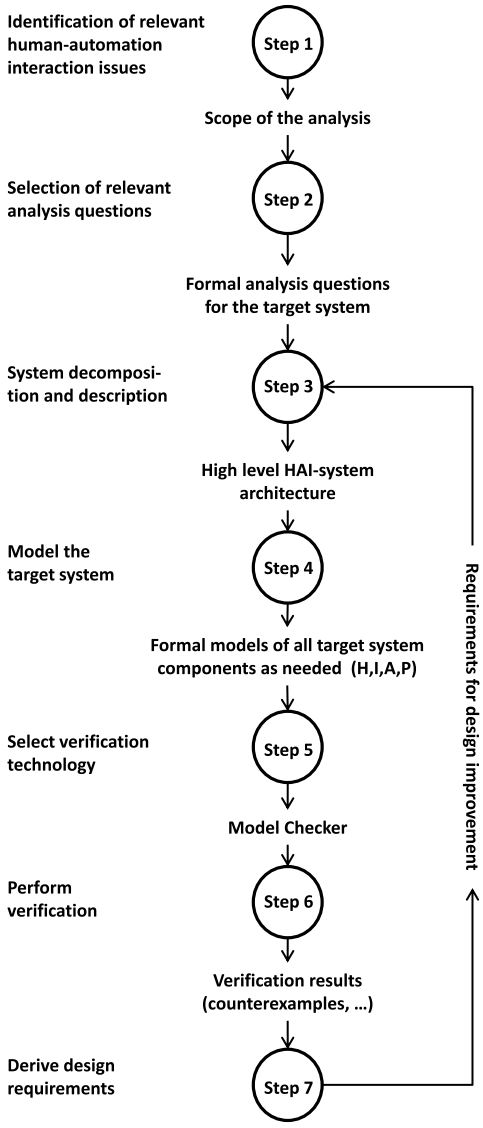

Figure 1: Steps of the Verification Methodology

Verification Methodology

The Verification Methodology comprises of seven constituent steps (see Figure 1), which will be described in detail in the following:

Step 1: Identification of relevant HAI issues
Step 2: Selection of relevant analysis questions
Step 3: System decomposition and description
Step 4: System modeling
Step 5: Selection of verification technology
Step 6: Verification
Step 7: Derivation of design requirements

Identification of Relevant Human-Automation Interaction (HAI) Issues

The methodology aims at addressing safety-related Human Factors issues in operational contexts. The first step therefore consists of identifying the main issues of interest for the Human-Automation Interaction (HAI) system at hand. Two main cases can be met:

a) The system already exists and operates in an operational context. Experience feedback is available, in the form of incidents or even accidents, for which the root causes need to be eliminated. The target issues to ad-dress in this case will be directly derived from that feedback (e.g., "Problems encountered when executing a procedure with unexpected results").
b) The system is new and no experience feedback is available. In this case, one should examine similar systems already in use (e.g., commercial airliner cockpit) in order to exploit known issues and available operational feedback (e.g., pilot out of the loop and degradation of situation awareness).

ECLS - Human-Automation Interaction Issues

We present a currently conducted case study to illustrate the methodology. The Environmental Control and Life Support (ECLS) system is a subsystem of the European science laboratory Columbus, which is part of the International Space Station (ISS). It has the important task of keeping the atmosphere inside the Columbus module in a healthy and habitable condition for human beings. Therefore it consists of four functional groups: (1) Air Condition, (2) Atmosphere Pressure Control, (3) Payload Supply, and the support of (4) Fire Detection & Suppression (FDS).

All functions in the ECLS System are safety-critical, ensuring crew health and vehicle integrity. The aspect of safety is covered by the detection of critical situations and the appropriate saving activities, which may interrupt any service. Saving mechanisms can be found on different levels. The lowest level is the equipment or functional level. If for instance the air flow falls below a

limited value and smoke detection cannot be performed automatically by the system any more, an event is raised and the crew will be informed to take over for smoke detection. Such an event must be evaluated in real-time operations until any critical or safety relevant root cause can be excluded.

Human-Automation Interaction Issues
The ECLS system has been in operational use for several years. Relevant issues that are addressed in the ECSL case study are based on operational experiences. The Columbus Flight Note System is a database that lists all anomalies and problems that occurred during the operation of Columbus. Amongst these are interesting cases where some procedure execution failed in various contexts. The objective of the study is to understand when exactly the procedures fail, why and how that could be corrected at the system level (e.g., re-design of the procedures, improvements to user interfaces).

Selection of Relevant Analysis Questions

The next step consists of selecting a set of analysis questions that capture the essence of the issues identified during the previous step, be they for a new or an existing Human-Automation Interaction system.

The methodology relies on a series of 38 predefined analysis questions, organized into eight categories.
The list below shows the different categories of questions (with the number of questions per category):

1) Information on automation states & behaviors (9)
2) Issuing commands towards automation (7)
3) Understanding automation: complexity issues (11)
4) Situation awareness and out of the loop problem (3)
5) Workload changes (2)
6) Vigilance (2)
7) Skill acquisition/degradation (3)
8) Trust (1)

The analysis questions are the result of an extensive analysis of the literature on Human-Automation Interaction issues. For example, based on Degani and Heymann (2002):

> *Is the operator informed when state transitions (e.g., mode transitions) occur?*

The analysis questions on which the next steps will focus are contained in that list, and match the key issues to be addressed.

ECLS - Analysis questions

A first analysis of the flight note database suggests the problems with the procedures may be due to the user applying the procedures in inappropriate internal automation modes. To investigate this hypothesis, we chose to address two analysis questions:

- *Is the operator informed on automation states?*
- *Is the operator informed when automation state transitions occur?*

We want to ensure the operator is always able to detect when the ECLS system changes its state. The operator should at all time know what is the internal state of the ECLS system.

System Decomposition

The methodology relies on detailed, formal models of the different constituents of Human-Automation Interaction systems. These formal models are needed for the verification techniques to be applicable.

The first step toward modeling is therefore to decompose the Human-Automation Interaction system into its components, with a particular focus on those relevant for the analysis questions previously selected. The methodology considers that such systems can be decomposed in the following types of components:

Agents: Agents are of two types, human and machine (automation). Both execute tasks or achieve functions using specific resources (e.g., information from sensors). Resources include the different types of simple, medium level and complex artifacts human agent utilizes in operational situations e.g., including logbooks, activity sheets, printers, smart whiteboards, simulations (cf. Hollnagel and Woods 2005, pp. 99-100).

Processes: These are acted upon by the agents who perform tasks and use resources.

Environments: Agents and processes are immersed in environments. Environments cannot be acted upon but can nevertheless influence agents and processes.

Interfaces: Any interface between the three previous types of components, e.g., user interfaces between human and machine agents.

In order to limit the description and modeling effort, only the components relevant for the selected analysis questions should be addressed, i.e. for a procedure applied by a human operator to set the state of a system, focus should be placed on the human agent, its tasks (the procedure) and the machine agent (system) upon which the procedure operates. To identify and describe these components for a given Human-Automation Interaction system we rely on Cognitive Work Analysis (Vicente 1999). Cognitive Work Analysis comprises of five phases to be used in combination or isolation, depending on the objectives of the study. They are relevant for our decomposition and description efforts:

Work domain analysis (Burns et al. 2009) structurally decomposes the Human-Automation Interaction system into its subcomponents. Human operators and automated systems are considered instances of the more general class of agents, human agents and machine agents respectively, who both contribute to the work domain.

Control task (or activity) analysis and strategy analysis are employed to determine the agents' tasks and functions, and how they are performed (activity and strategies).

Social organization and co-operation investigates how agents cooperate (co-operation), how they exchange information (social organization) and how they handle dynamic task distribution

Worker competencies analysis is extended in our ontology to cover the competencies of machine agents (e.g., in terms of the service they can provide).

This approach affords the description of all information needed on the agents involved in the system, their competencies, functions and tasks, including how they are dynamically associated, the resources used by these agents to perform the tasks, the processes upon which the tasks are executed, as well as the environments and constraints in and against which control is achieved.

ECLS - System Decomposition

For the ECLS case study, the key components of the Human-Automation Interaction situation are:

- the human operator(s), including the procedures he/she has to apply,
- the user interface with the ECLS system,
- the ECLS system itself, modeled with a level of details and scope sufficient to support the execution of the target procedures by the human operator, through the ECLS user interface.

System Modeling

Models are the crucial part of the verification process. It is highly recommended to avoid having to create a model of an existing system just for verification purposes. Instead, the verification should rely on models already available during the design process (Varró and Pataricza 2003). A model-driven design process is therefore beneficial, if not required, for the application of verification techniques. Nevertheless, models should be as abstract as possible to ease the verification process, while being expressive enough to support the design activities and address the analysis question at hand. In the system modelling step, models for each component identified in the system decomposition step are created as state transition systems.

The methodology presented here focuses on how human agents interact in the context of automated system, yet it is not decoupled from the correctness analysis of the technical system. This analysis can provide abstract formal specifications of the often complex machine agents and their interaction with the process and the environment.

To address the analysis questions (Step 2) the component models must include several aspects, although not all are required for each analysis question.

Machine agent models should at least be detailed enough to describe the mode logic of the automation. Many problems that occur during the operation of automation systems and are associated to human automation issues can be traced back to the interaction between the human agent and the mode logic of the automation (Sarter, Woods and Billings 1997). Many questions (especially in categories 1 and 2) address issues related to automation modes.

Human agent models address different aspects. Nominal task models describe how the operator has to execute his/her tasks. Mental models describe the operator's view on the system. Cognitive models describe how information is cognitively processed and organised. In the present context especially the limitations of the cognitive system are of interest, because these are potential sources of human errors. In this context these processes may be called *error production mechanisms*, although their purpose is certainly not to produce errors.

Interface models describe the information exchanged between the different components of the decomposed system. Of special interest for several analysis questions are the interfaces to the user. The model of a simple interface might just be a listing of input and output variables of the machine agent or process models. For more complex ones, however, this should also include the navigation logic and display layouts.

Process and Environment models should be as simple as possible, though the required level of detail strongly depends on the application domain.

ECLS - System Modeling

A formal definition of the interface model for the ECLS system is provided on the basis of the Unified Synoptic System format (Brauer, Wolff, and van Leeuwen 2009; Nicklaussen 2003). It includes the graphical layout, as well as the navigation logic. It can be translated into a suitable verification model in a semi-automatic way.

The tasks that the human operators of the ECLS system have to perform are explicitly formulated in flight procedures using the Operations Data File (ODF) standard format (International Space Station Program 2007). It is available in human-readable form for the operators to use. Furthermore machine-readable versions are available as XML files. Thus the translation into a suitable verification model can also be supported by tools. However, the format is only semi-formal and contains several clear-text instructions and notes.

Furthermore the connection between flight procedure and user interface elements is sometimes ambiguous. Therefore the translation is only semi-automatic and requires several manual interactions.

Selection of Verification Technology

A great amount of verification tools are freely or commercially available with different features and for different purposes. There is not one tool that is the most suitable for the methodology presented here, and several aspects influence the selection of verification technology in Step 5.

The analysis questions selected in Step 2 are contained in a database that list suitable verification approaches for each analysis question. Many approaches are well known from the literature or represent modifications of known approaches, e.g., to mode confusion (Gow, Thimblebly, and Cairns 2005), (Degani and Heymann 2002), interface usability (Campos and Harrison 2008), or procedural issues (Curzon and Blandford 2001). The different approaches rely on different notations for the component models and may require special features like probabilistic model checking approaches or clocks. This restricts the choice of verification tools. The notation in which the property to be analyzed is specified (typically Linear Temporal Logic (LTL) or Computational Tree Logic (CTL)) may further restrict the tool selection. Finally the nature of the different component models, e.g., discrete, hybrid, also has an influence on the tool selection. Thus there is probably not one specific verification technology, which is suitable to address all analysis questions. The technology must be chosen thoroughly with all relevant analysis questions in mind.

ECLS - Verification Technology

The ECLS system controls processes like airflow and atmosphere pressure, which are best described as continuous dynamic behaviors. For many of the relevant issues, e.g., from category 1) *Information about automation states and behaviors*, we expect that an abstract description with discrete behavior models is sufficient for the analysis. The specifications of these analysis questions are given as LTL formulas.

The translations of the interface and task models will only be implemented for one target verification tool, which we deem flexible enough to handle several analysis questions. We choose the HySAT tool (Fränzle, et al. 2007). It is a bounded model checker, which is able to test the LTL specifications and can handle non-linear, arithmetic constraints including transcendental functions over integer and float variables. However this is ongoing work and the choice might have to be reconsidered.

Verification

Finally, the verification is executed in Step 6. The system and property specifications have to be created in the format of the selected verification tool. In case of termination of the verification run and identification of property counter examples, the results have to be evaluated. The verification approaches for many analysis questions aim at identifying potential weak spots. However, not every counter example identified by the verification approach might be relevant. Thus the results need to be evaluated manually, in order to filter for the relevant traces.

The analysis questions are specified in order to identify weaknesses in the design of the human-automation system. Besides the identification of weak spots, it is of interest to ascertain how robust the system is against human errors. To analyze this, the models can be extended with alternative execution paths at locations which are expected to be prone to errors. The activation of these errors is specified as error scenarios, based on predictive models (e.g., Javaux 2002; Lüdtke 2005). Such models apply either to task models or human mental models. By injecting some errors into the models, the system's fault tolerance can be analyzed.

Derivation of Design Requirements

The main objective of the methodology is not only to verify how well it satisfies the target analysis questions, but also to suggest ways of improving the current design to afford effective resolution of potentially unsatisfying Human-Automation Interaction issues that may have been uncovered.

To reach this objective, the methodology specifies for each analysis question how to exploit the result of formal verification (e.g., traces) to improve the system and mitigate, reduce or completely eliminate the issue at stake.

Conclusion

With the Verification Methodology presented in this paper, we provide a framework that shall systematically support the use of formal verification methods for the analysis of complex human-automation systems. The focus of this work is not the development of new approaches to address each of the analysis questions in the database. Rather, we collect existing ones from literature in a structured database and extend this database incrementally.

The Verification Methodology consists of several steps, and implementing these in a system design process requires significant effort. Especially the model generation step can be laborious. We expect that a suitable model-driven design process reduces this effort. The design model specifications or even the implementation models can be used as a basis for a tool-supported abstraction and

translation process. This also facilitates avoiding the introduction of errors in the verification models during the translation process.

A common problem with methods for exhaustive verification is that they do not always scale well with the complexity of the system. If the system is too complex to be represented on a suitable abstraction level for certain analysis questions, other approaches can typically be used, like user studies or simulations. In these cases the methodology again benefits from integration in a model-based design process. For highly complex systems the most suitable analysis techniques are certainly simulations (Engell, Frehse, and Schnieder 2002). Indeed we do not expect to replace such methods with our Verification Methodology, but to complement them.

The benefit of our methodology is that knowledge about problems typically associated to human factor issues is formally and non-ambiguously defined in the analysis questions and the proposed approaches to address them. This allows a structured and tool-supported reuse of this knowledge. However, the methodology is only able to identify those issues that have already been defined in the analysis question database. Unexpected problems are hard to identify automatically, and instead require user studies.
The verification methodology will be evaluated in the case study outlined earlier. Most importantly, it will be tested whether this approach is able to identify potential problems in the Human-Automation Interaction system within a reasonable amount of effort. Furthermore the relevance of the identified issues will be assessed. Some approaches to the analysis questions might tend to identify many potential issues, but only few relevant ones. This introduces additional effort, because the identified issues must be checked and analyzed manually in order to derive design improvements in Step 7.

Finally, the usefulness of the verification results (e.g., counter example traces) for deriving improved system requirements shall be discussed on the basis of the case study.

References

Brauer, U., Wolff, M. & van Leeuwen, W. 2009. Enhanced Operations Efficiency by using the Unified Synoptic System. *Acta Astronautica*, 64:1150-1159.

Burns, C., Torenvliet, G., Chalmers, B. &Scott, S. 2009. Work Domain Analysis for Establishing Collaborative Work Requirements. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting,* San Antonio, Texas, October 19-23, 2009, 53(4): 314-318.

Campos, J. C. & Harrison, M. D. 2008. Systematic Analysis of Control Panel Interfaces Using Formal Tools. In T. N. Graham, and P. Palanque (Ed.), *Design, Specification and Verification of Interactive Systems (DSVIS 2008)*, 72-85. Berlin and Heidelberg Springer.

Curzon, P., & Blandford, A. 2001. Detecting Multiple Classes of User Errors. In *Proceedings of the 8th IFIP International Conference on Engineering for Human–Computer Interaction*, 57-72.

Degani, A. & Heymann, M. 2002. Formal Verification of Human-Automation Interaction. *Human Factors*, 44(1): 28-43.

Dix, A. J. 2013. Formal Methods. In M. Soegaard, & R. F. Dam (Eds.), *The Encyclopedia of Human-Computer Interaction* (2nd ed.). Aarhus, Denmark: The Interaction Design Foundation.

DO-178C/ED-12C, *Software Considerations in Airborne Systems and Equipment Certification*, RTCA/EUROCAE, 2012.

DO-333/ED-218: *Formal Methods supplement*, RTCA/ EUROCAE, 2012.

Engell, S., Frehse, G. & Schnieder, E. (Eds.) 2002. *Modelling, Analysis, and Design of Hybrid Systems*. Berlin and Heidelberg, Springer.

Fränzle, M., Herde, C., Teige, T., Ratschan, S. & Schubert, T. 2007. Efficient Solving of Large Non-linear Arithmetic Constraint Systems using Complex Boolean Structure. *Journal of Satisfiability, Boolean Modeling and Computation*, 1: 209-236.

Gow, J., Thimblebly, H. & Cairns, P. 2005. Automatic critiques of interface modes. In *Proceedings of 12th international conference on Interactive Systems: design, specification, and verification (DSVIS 2005)*. 201-212. Berlin and Heidelberg, Springer.

Hollnagel, E. & Woods, D. D. 2005. *Joint Cognitive Systems: Foundations of Cognitive Systems Engineering*. London CRC Press.

IEC 61508 Standard, *Functional Safety of Electrical/Electronic/ Programmable Electronic Safety-related Systems*, 2nd Ed., 2010.

ISO 26262 Standard, *Road vehicles – Functional safety*, ISO, 2011.

International Space Station Program, 2007. *Operations Data File Standards,* Houston, Texas: National Aeronautics and Space Administration.

Javaux, D. 2002. A method for predicting errors when interacting with finite state systems. How implicit learning shapes the user's knowledge of a system. *Reliability Engineering & System Safety*, 75(2): 147–165.

Lüdtke, A. 2005. *Kognitive Analyse formaler sicherheits-kritischer Steuerungssysteme auf Basis eines integrierten Mensch-Maschine-Modells*, Ph.D. diss., Department of Computer Science, University Oldenburg.

MODUK, Defence Standard 00-55, *Requirements for Safety Related Software in Defence Equipment*, Part 1: Requirements/ Part 2: Guidance, Issue 2, UK Department of Defense, MODUK: British Defence Standards 1997.

Nicklaussen, D. 2003. Unified Synoptic System. In *Proceedings of DASIA 2003,* Prague, Czech Republic.

Sarter, N. B., Woods, D. D. & Billings, C. E. 1997. Automation Surprises. In G. Salvendy, *Handbook of Human Factors & Ergonomics* (2nd Ed.). New York, John Wiley & Sons.

Varró, D., & Pataricza, A. 2003. A. Automated Formal Verification of Model Transformations. In *Proceedings of CSDUML 2003 Workshop: Critical Systems Development in UML*, San Francisco, USA, 63-77.

Vicente, K. J. 1999. *Cognitive Work Analysis: Towards Safe, Productive, and Healthy Computer-Based Work*. London, Lawrence Erlbaum Associates.