The Need for a Confidence View of CPS Support Environments (Fast Abstract)

Fredrik Asplund and Martin Törngren
KTH Royal Institute of Technology,
Brinellvägen 83, 10044 Stockholm, Sweden
Email: {fasplund, martint}@kth.se

Richard Hawkins and John A. McDermid
University of York
Deramore Lane, York, YO10 5GH, UK
Email: {richard.hawkins, john.mcdermid}@york.ac.uk

***Abstract*—Multi-View Modelling Integration Frameworks (MVMIFs) may help mitigate complexity associated with the development of CPS, but may also have implications on safety. Safety-related standards do not provide guidance to mitigate this problem. We therefore suggest that MVMIFs are extended with a confidence view to support the creation of an assurance case that covers issues related to risks in the support environment.**

I. Introduction

The complexity of the relationships between stakeholders, processes, models and tools during the development of *Cyber-Physical Systems* (CPS) is growing. The resulting *ad hoc* integration of models and tools threatens to impact development efficiency [1]. Multi-View Modelling Integration Frameworks (MVMIFs) may help to deal with these complex dependencies by handling multiple aspects of CPS, including safety, in a structured and coordinated fashion. Tools and tool integration do, however, have safety-related implications, a concern that several high-profile safety and safety-related standards for CPS development try to address (see e.g. ISO 26262 [2]). Unfortunately, no consistent approach exists that go beyond separate tools, i.e. that deal with tool chains and frameworks that provide integration between tools [3].

II. Multi-View Modelling Integration Frameworks

Supporting engineering through the use of tools has received much attention, since the number of interactions during CPS development is growing. Large research projects (have) focus(ed) on this with regard to e.g. requirements engineering [4], embedded systems [5] and systems of systems [6].

One of the approaches that has been put forward is that of *Multi-View Modelling* (MVM). In this approach multiple *views* are used to develop a product. Views are representations of a whole system from the perspective of a related set of concerns. These views deal with the complexity of modern development by "filtering" what different stakeholders see and structuring how they interact. The term MVM applies when models are developed based on several perspectives, typically using different modelling languages. Various discussed aspects of MVM include (a) proposals for specific formalisms and views that aid in integration of tools for MVM, such as dependency models [7]; (b) Proposals for technology that aid in the integration of tools for MVM, such as model-based tool integration [8]; and (c) processes and multi-view frameworks, such as architectural frameworks that define different viewpoints that together provide a description of a system [9].

Unfortunately, the relationships between the models are, despite the attention garnered by the MVM approach, often not systematically and cost-efficiently formalized. Disconnected tools lead to manual, inefficient, error-prone transfers of data; duplicated information lead to change management issues; improper alignment of concepts and assumptions lead to misunderstandings among stakeholders.

MVMIFs, essentially middlewares for defining, managing and automating part of the interactions between views, provide a way to overcome these problems. Hopefully MVMIFs can pave the way for benefiting from the MVM approach.

III. The Problem

MVMIFs are not the only reason for increasing the automation between tools during CPS development. More advanced tool integration is needed to relate different simulations [10], for collaborating across organizational boundaries / engineering disciplines (see [11], [12]), and so on. However, while most issues can be (inefficiently) supported through a slow migration using *ad hoc* tool integration, the introduction of a MVMIF will have a direct impact on engineering practices. This impact is tied to a significant increase in tool integration, especially with regard to low levels of *acquisition automation* and *analysis automation*[1]. If there are safety-related implications of such an increase in tool usage the net result might actually be negative [14].

Several high-profile safety / safety-related standards for CPS development already highlight that tool usage can have safety-related implications. Two primary perspectives can be identified. The first perspective establishes that trust in a tool used during safety-critical development is ensured by its development process constraints (see e.g. DO-178C [15]). Tool integration merely provides a tool context, important in so far as how to prepare the tool qualification for different tools. The second perspective stipulates that trust in tools is established through generic measures, such as securing thorough specifications (see e.g. IEC 61508:2010 [16]). The focus is

[1] In other words, automation related to the sensing / registration of input data and automation of working memory / inferential processes [13].

on the reliability of a specific set of tools. Tool integration is mostly important with regard to how it supports the quality of tool input. These two approaches do not preclude a top-down approach on tools and tool integration (high level guidelines in DO-178C and IEC 61508:2010 underline considerations such as consistency and complementarity of tools throughout the development life-cycle). However, neither approach provides any guidance on how to ensure trust throughout a whole tool chain. These perspective can be summarized as viewing automation as a "team member" (see [17] for a discussion of this type of perspective). In short, according to these standards automation can and should be evaluated based on its reliable execution of separate process steps independent of human operators. Automation that only supports the actions and decision-making of operators is relatively inconsequential, as long as the rest of the standards are adhered to successfully.

These bottom-up perspectives are therefore probably insufficient for dealing with all safety-related implications of introducing MVMIFs.

IV. A Way Forward

For safety-related CPS, it is common practice to develop an assurance case (safety case) to demonstrate through the provision of argument and evidence that the CPS is sufficiently safe to operate [18]. Whilst the core of any assurance case must be the technical risk argument (emphasizing product faults and failures) [19], inadequate attention has been given to how the assurance of the support environment relates to faults and failures in the product. A thorough handling of risks related to tools and tool integration during development would require the consideration of a more complete set of risks related to data integrity, locating/communicating complex data, traceability, differing understanding of data semantics/graphical notations, process notification/control, obsoleteness of data, automation, etc. (studied previously [20], but only recently given further attention [21], [22]). Our suggestion is to provide this through a confidence view of the CPS support environment to MVMIFs. This confidence view would facilitate the analysis of dependencies and failures in the support environment; the required confidence in the support environment could then be demonstrated through the an additional part of the assurance case, genereted from this confidence view.

Furthermore, assurance case patterns are an established technique for the documentation and reuse of argument structures [23]. Through developing assurance case patterns based on the dependencies between tools in this confidence view, although some per-project qualification will still be required, reassurance of tools across multiple tool chains should become easier. This approach would also maximise the utilisation of existing per-tool assurance information.

V. Conclusion

This paper states that the safety-related implications of MVMIFs may be mitigated by the creation of an additional confidence view of the support environment. We plan to both investigate the changes implied by MVMIFs with regard to safety standards and prototype a confidence view based on these investigations.

References

[1] iFEST Consortium. (2013) iFEST project final report. [Online]. Available: http://www.artemis-ifest.eu/sites/artemis-ifest.eu/files/iFEST_Final_Report_publishable_0.pdf

[2] *ISO/FDIS 26262:2010, Road vehicles - Functional safety*, International Organization for Standardization Std., 2010.

[3] F. Asplund, J. El-khoury, and M. Törngren, "Qualifying software tools, a systems approach," in *Computer Safety, Reliability, and Security, 31st Int. Conf., SAFECOMP 2012*, 2012.

[4] CESAR Consortium. (2012) CESAR. [Online]. Available: http://www.cesarproject.eu/

[5] iFEST Consortium. (2013) iFEST. [Online]. Available: http://www.artemis-ifest.eu/

[6] DANSE Consortium. (2014) DANSE. [Online]. Available: http://www.danse-ip.eu/

[7] M. Törngren, A. Qamar, M. Biehl, F. Loiret, and J. El-khoury, "Integrating viewpoints in the development of mechatronic products," *Mechatronics*, 2013.

[8] E. Kapsammer, T. Reiter, and W. Schwinger, "Model-based tool integration - state of the art and future perspectives," in *In the Proc. of the 3rd Int. Conf. on Cybernetics and Information Technologies, Systems and Applications*, 2006.

[9] R. Siegers, "The ABCs of AFs: Understanding architecture frameworks," in *Proc. of INCOSE Int. Symp.*, 2005.

[10] P. J. Mosterman and H. Vangheluwe, "Computer automated multiparadigm modeling: An introduction," *SIMULATION*, vol. 80, pp. 433–450, 2004.

[11] M. Broy, M. Feilkas, M. Herrmannsdoerfer, S. Merenda, and D. Ratiu, "Seamless model-based development: From isolated tools to integrated model engineering environments," *Proc. of the IEEE, Special Issue on Aerospace and Automotive Software*, vol. 98, no. 4, pp. 526–545, April 2010.

[12] A. Benveniste, B. Jonsson, G. Buttazzo, and L. Thiele, *Embedded Systems Design, The ARTIST Roadmap for Research and Development*, ser. Lecture Notes in Computer Science, B. Bouyssounouse and J. Sifakis, Eds. Springer-Verlag, 2005, vol. 3436.

[13] R. Parasuraman, T. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000.

[14] A. Galloway, J. A. McDermid, J. Murdoch, and D. Pumfrey, "Automation of system safety analysis: Possibilities and pitfalls," *Proceedings of ISSC*, 2002.

[15] *DO-178C, Software Considerations in Airborne Systems and Equipment Certification*, Special Committee 205 of RTCA, Inc. Std., 2011.

[16] *BS/IEC 61508:2010, Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*, International Electrotechnical Commission Std., 2010.

[17] A. R. Pritchett, "Aviation automation: General perspectives and specific guidance for the design of modes and alerts," *Reviews of Human Factors and Ergonomics*, vol. 5, pp. 82–113, 2009.

[18] R. Bloomfield and P. Bishop, "Safety and assurance cases: Past, present and possible future - an adelard perspective," in *Making Systems Safer*. Springer London, 2010.

[19] R. Hawkins, I. Habli, T. Kelly, and J. McDermid, "Assurance cases and prescriptive software safety certification: A comparative study," *Safety Science*, vol. 59, pp. 55–71, 2013.

[20] A. Hutcheon, D. Jordan, J. McDermid, R. Pierce, I. Wand, and B. Jepson, "High integrity software development: Process and tool issues," *Microprocessors and Microsystems*, vol. 19, no. 9, pp. 517 – 524, 1995.

[21] F. Asplund, M. Biehl, and F. Loiret, "Towards the automated qualification of tool chain design," in *Proc. of the SAFECOMP 2012 Workshops*, 2012.

[22] F. Asplund, "Risks related to the use of software tools when developing cyber-physical systems," Ph.D. dissertation, KTH Royal Institute of Technology, 2014.

[23] R. Hawkins, K. Clegg, R. Alexander, and T. Kelly, "Using a software safety argument pattern catalogue: Two case studies," in *Proc. of the 30th International Conference on Computer Safety, Reliability and Security (SAFECOMP 2011)*, 2011.