Invited paper, presented at Workshop on Open Systems Dependability (WOSD) 2017, Keio University, Japan, October 2017.

Automated Integration of Potentially Hazardous Open Systems

John Rushby
Computer Science Laboratory
SRI International, Menlo Park CA USA

***Abstract*—I speculate on the feasibility of open systems that self-assemble into integrated systems of systems using automation to identify and manage novel hazards.**

1. Automated Integration of Open Systems

One of the benefits expected of open systems is that they can be combined as systems of systems to deliver some integrated service beyond that provided by any of the constituents alone. Of course, we would like some assurance that the integrated service accomplishes what we require and, perhaps more importantly, that it maintains safety, or other critical properties. Furthermore, we can imagine that these integrated systems of systems are constructed dynamically (i.e., during operation), either under the control of some "integrating app," or spontaneously as the constituent system discover each other (e.g., when multiple medical devices are attached to a single patient). We then need the assurance of required and safe behavior to be constructed automatically and dynamically also.

One way to accomplish dynamic integration and assurance is for the individual systems to be supplied with models of their properties, assumptions and behavior, and an argument providing assurance that they deliver their required and critical properties. As systems integrate into systems of systems, they exchange their models and assurance case arguments and compose these into a larger model and argument for the integrated system. The first step (exchange and composition of models) is an emerging framework known as Models@Runtime (M@RT) [1] while the second step (exchange and synthesis of assurance arguments) is known as *Safety* Models@Runtime (SM@RT) [2].

Trapp and Schneider [2] distinguish four levels of sophistication and difficulty in SM@RT according to how ambitious is the integration, and note that only the first two are feasible at present. My interpretation of this four-level hierarchy, starting with the simplest case, is the following. The focus here is on safety, but the ideas can be generalized to other critical properties, or to conventional requirements.

Unconditionally safe integration. Here, the component systems guarantee their own safety, with no assumptions on their environment. It follows that when two or more such systems are integrated into a system of systems, the result is also unconditionally safe. Trapp and Schneider refer to this class of systems as "Safety Certificates at Runtime."

Conditionally safe integration. Here, the component systems guarantee their own safety, but do have assumptions on their environment. When two such systems are integrated into a system of systems, each becomes part of the environment of the other and it is necessary for them to exchange their models and assurance arguments and to prove that the assumptions of each are satisfied by the properties of the other. The resulting system will also be conditionally safe. Trapp and Schneider refer to this class of systems as "Safety Cases at Runtime."

Safely managed integration. This class is similar to the previous one except the component systems are not able to ensure each others assumptions. Hence one or both systems must be adapted in some way, generally by synthesizing a wrapper or runtime monitor that excludes the troublesome cases. For example, if one system delivers an unacceptable result, a runtime monitor can block it and signal failure to the other system. Or if one system cannot deliver the assumed behavior in some cases, a wrapper can block or transform its inputs to exclude those cases. Trapp and Schneider refer to this class of systems as "V&V at Runtime."

Safe integration despite hazards. In this class, it is possible that the integrated system has hazards (i.e., potentially unsafe circumstances) not present with either system individually. For example, a surgical laser may be safe and an anesthesia machine may be safe, but the combination possesses a new hazard that the laser can cause burning and fire in the enriched oxygen supplied by the anesthesia machine [3]. Once the hazards are known, this class can be transformed into the previous one (e.g., the laser can be disabled if the anesthesia machine is delivering enriched oxygen, or the anesthesia machine can be instructed not to use enriched oxygen if the laser is operating). Trapp and Schneider refer to this class of systems as "Hazard Analysis and Risk Assessment at Runtime."

The first class of integrated systems is straightforward; the second is seen in the Japanese DEOS project [4] and in the already cited work of Trapp and Schneider. The third class is anticipated in the NATO interoperation framework called SILF [5] and prototyped in an SRI project called ONISTT [6]. I outline these projects and discuss related ideas and prospects in a previous paper [7].

Here, I wish to focus on the fourth class, where the integrated system may possess new hazards, not present in any of its constituents.

2. Automated Hazard Analysis

It should be noted that each of the integration classes, other than the first, substitutes automation at integration time for activities that are traditionally performed by humans at design time. Furthermore, these are activities traditionally considered to require human expertise. For conditionally safe integration, the automation most likely employs mechanized deduction to prove the theorems that ensure the assumptions of one component system are satisfied by the properties of the other. Now, mechanized deduction is nothing but a very sophisticated search over the models and assurance arguments provided with the component systems. Success in this search depends in part on the power of the methods of deduction employed, and in part on the quality of the models and arguments supplied—and these are the products of human ingenuity and expertise. Thus, automated integration does not eliminate human expertise but relocates it from the act of design to the recording of design knowledge and understanding in the form of models and assurance arguments.

For safely managed integration, the automation must perform mechanized synthesis, which can be organized as a further search on top of mechanized deduction (guess a solution, try to verify its correctness, if that fails use counterexamples to help refine the guess and iterate). Again, human expertise is recorded in the models which, for synthesis, can usefully be in the form of templates whose parameters are instantiated by the search-based procedure.

I propose that hazard analysis can likewise be organized as a search over models. The challenge of hazard analysis is that the search space includes not just computational interactions among the component systems and between them and their environment, but interactions in many other dimensions. For example, one system may place an excessive computational load on another, causing it to overheat and potentially cause a fire. Consequently, hazard analysis is generally seen as quintessentially an activity requiring human experience and skill: "graybeard" experts can mentally sweep the vast space of possibilities to home in on the ones that matter, rather in the way that a chess grandmaster rapidly focuses on the most promising moves, without (apparently) doing an exhaustive search over all possibilities.

Even "graybeards" may overlook some possibilities, so there are systematic processes for hazard analysis that help guide the search in productive directions: Xu et al. provide a comprehensive summary [8]. Among the most effective methods are HAZOP and Leveson's STPA. The former models the system and its environment as "flows" of data and control and then asks "what happens if this value is —?" where — is selected from a catalog of "guidewords" such as "missing," "late," "small," etc. STPA models the system and its environment (including its development and regulatory environment) as a series of control systems and likewise contemplates the effects of errors and disturbances in these.

Catastrophic safety failures are sufficiently rare that most designers will not see one in their working lifetime and, probably for this reason, many analysts and designers discount harbinger events as "anomalies" and overlook or disregard hazard scenarios that seem to require very rare or improbable events. Ironically, these are a significant cause of real system failures, as documented in a NASA study [9].

I propose that we contemplate the construction of models that can support, say, HAZOP, and explore the possibility of hazard analysis by automated search over those models, including rare and improbable events. Until recently, such automation would have been infeasible as the models employed are too abstract and high-level (generally box and arrow diagrams) to support effective mechanized search. Nowadays, however, SMT solvers can support highly effective search over very abstract models using infinite bounded model checking with uninterpreted functions [10], [11]. I admit to having no experience of constructing such models and automating hazard analysis through mechanized HAZOP nor, as far as I know, has anyone else, but I claim it is now conceivable that it could be done. The required models will concern the component systems themselves, and their environment. I propose that we could start with some fairly restricted domain, such as medical devices, where the associated environment is itself relatively restricted—namely human physiology and its local physical environment. One might suppose that such a model would contain an element that a source of energy in conjunction with a "large" flow of oxygen triggers a potential "burn" or "fire" hazard. If the model of a laser notes that it is a source of energy and that of an anesthesia machine records the possibility that it can produce enhanced (i.e., "large") oxygen, then our search will reveal the potential "burn" hazard in the composed system of systems.

Clearly much research is required to identify suitable ontology, logic, structure, and automated deduction or search methods to make such modeling and analysis feasible and effective. But once feasibility is established, I suggest that development of environment models could become a collaborative enterprise, where a whole industry (such as medical devices) contributes to the construction and elaboration of an environmental hazard model that serves as a common resource. A model of this kind must deal with a wide range of phenomena (a critic might say the "whole world"), but it can be quite abstract (we do not need to calculate that fire will definitely break out in 6.5 seconds under such and such conditions, merely that it is possible). Furthermore, I suspect it will be compositional—that is, the modeling for "burn" will be disjoint from that for "overdose."

3. Conclusion

The vision that I propose for automated integration of open systems is that they are supplied with models that support calculation of their contribution to hazards, with other models that support calculation of their behavior and properties, and with an assurance case argument that justifies their safety (or other critical property). When two systems encounter each other, they exchange their hazard models and perform automated hazard analysis by a search over the composition of their individual models and that of their environment (which could be a standardized common resource). If new hazards are identified then these are added to the next stage of the process, which is synthesis of wrappers and monitors to exclude behaviors that can lead to hazards, and to the final stage, which is construction of a joint assurance argument (in effect, a proof [12], [13]) for safety.

Of the four classes in the integration hierarchy outlined in Section 1, Trapp and Schneider claimed that (only) the first two are feasible today [2]; I previously claimed that the third is also feasible [7], and this paper speculates that the fourth is at least conceivable.

Furthermore, I claim that development of standardized environment hazard models will be a social good: they will become the repository of much accumulated knowledge and experience. As new hazards and hazardous circumstances are discovered, they can be added to the model so that all benefit from the new and perhaps painfully acquired knowledge. Looking forward, we should think of integration as an ongoing process rather than a one-time event: as systems and the environment change and evolve, so the customizations and arguments that ensure functionality and safety of an integration may need to be revisited periodically [4], and this could be enabled as automated *re*integration.

I have described integration as something that occurs between two or more constituent systems but, in the world of social media and universal connectivity, any system is immediately and implicitly integrated with many others (i.e., "the world") as soon as it is made available. Nuisance and embarrassment or worse are possible if the potential hazards of such integration are not considered and countered. For example, Microsoft's "Tay" was a Twitter bot that the company described as an experiment in "conversational understanding." The more you chat with Tay, said Microsoft, the smarter it gets, learning to engage people through "casual and playful conversation." Within less than a day of its release, it had been trained by a cadre of bad actors to behave as a racist mouthpiece and had to be shut down. We can speculate that in time a near-universal environment hazard model could be developed, so that the developers of systems such as Tay could easily discover the potential for abuse long before release. At the very least, the experience of Tay could be incorporated into the environment model to ensure that it is never repeated.

Thus, by encouraging the construction of standardized community-supported environment hazard models, automated hazard analysis could improve the quality of life by generalizing application of hazard analysis, and with it the ability to predict and avoid unpleasant consequences of thoughtless system design, from critical systems to those of everyday life.

Acknowledgments.. This work was partially funded by SRI International.

References

[1] Bencomo, N., France, R., Cheng, B.H., Assmann, U., eds.: Models@Run.Time: Foundations, Applications, and Roadmaps. Volume 8378 of Lecture Notes in Computer Science. Springer-Verlag (2014) 1, 3

[2] Trapp, M., Schneider, D.: Safety assurance of open adaptive systems—a survey. [1] 279–318 1, 3

[3] Whitehead, S.F., Goldman, J.M.: Getting connected for patient safety: How medical device "plug-and-play" interoperability can make a difference. Patient Safety and Quality Healthcare (2008) Available at http://www.psqh.com/janfeb08/connected.html. 1

[4] Tokoro, M.: Open Systems Dependability—Dependability Engineering for Ever-Changing Systems. CRC Press (2013) 1, 3

[5] NATO Science and Technology Organization, Neuilly-Sur-Seine, France: Framework for Semantic Interoperability. (2014) STO Technical Report TR-IST-094. 1

[6] Ford, R., Hanz, D., Elenius, D., Johnson, M.: Purpose-aware interoperability: The ONISTT ontologies and analyzer. In: Simulation Interoperability Workshop. Number 07F-SIW-088, Simulation Interoperability Standards Organization (2007) 1

[7] Rushby, J.: Trustworthy self-integrating systems. In Bjørner, N., Prasad, S., Parida, L., eds.: 12th International Conference on Distributed Computing and Internet Technology, ICDCIT 2016. Volume 9581 of Lecture Notes in Computer Science., Bhubaneswar, India, Springer-Verlag (2016) 19–29 1, 3

[8] Xu, X., Ulrey, M.L., Brown, J.A., Mast, J., Lapis, M.B.: Safety sufficiency for nextgen. NASA Contractor Report NASA/CR-2013-217801, NASA Langley Research Center (2013) 2

[9] Driscoll, K.: Real system failures. NASA Dashlink (2012) https://c3.nasa.gov/dashlink/resources/624/. 2

[10] Rushby, J.: Composing safe systems. In: 8th International Symposium on Formal Aspects of Component Software (FACS'11). Volume 7253 of Lecture Notes in Computer Science., Oslo, Norway, Springer-Verlag (2011) 2

[11] Rushby, J.: The versatile synchronous observer. In Iida, S., Meseguer, J., Ogata, K., eds.: Specification, Algebra, and Software, A Festschrift Symposium in Honor of Kokichi Futatsugi. Volume 8373 of Lecture Notes in Computer Science., Kanazawa, Japan, Springer-Verlag (2014) 110–128 2

[12] Rushby, J.: Assurance and assurance cases. In Pretschner, A., ed.: Lectures from Marktoberdorf Summer School 2016. IOS Press (2017) To appear. 3

[13] Rushby, J.: The indefeasibility criterion for assurance cases. In: Shonan Workshop on Implicit and Explicit Semantics Integration in Proof Based Developments of Discrete Systems, Kanagawa, Japan (2016) Postproceedings to be published in Springer LNCS. 3