PVSio-web: mathematically based tool support for the design of interactive and interoperable medical systems

[Invited Paper]

Paolo Masci
Queen Mary University of London, United Kingdom
p.m.masci@qmul.ac.uk

Patrick Oladimeji
Swansea University
United Kingdom
p.oladimeji@swansea.ac.uk

Piergiuseppe Mallozzi
University of Pisa
Italy
piergiuseppe.mallozzi@gmail.com

Paul Curzon
Queen Mary University of London, United Kingdom
p.curzon@qmul.ac.uk

Harold Thimbleby
Swansea University
United Kingdom
harold@thimbleby.net

ABSTRACT

Use errors, where medical devices work to specification but lead to the clinicians making mistakes resulting in patient harm, is a critical problem. Manufacturers need tools to help them find such design flaws at an early stage and regulators need tools to help check devices are safe to approve for market. We have developed a prototyping tool, PVSio-web, to help check the safety of medical device interface and interaction design. It supports a model-based design process: that is, it is based on precise mathematical descriptions of the device's behaviour. This allows sophisticated proof and model checking technology to be used to verify that devices meet essential safety requirements. The architecture allows for the flexible addition of 'plug-in' modules to extend its functionality giving different views of the design that allow different stakeholders to work together. Working with the US regulator, the Food and Drug Administration (FDA), our tool has helped identify problems in a series of commercial medical devices. Hospitals have used it as part of training programmes highlighting safety-related design issues. In ongoing work we are developing plug-ins that support the verification and validation of interoperable medical systems.

Categories and Subject Descriptors

H.1.2 [**Information Systems**]: User/Machine Systems—*human factors*; J.3 [**Life and Medical Sciences**]: Medical Information Systems; K.4.1 [**Computers and Society**]: Public Policy Issues—*human safety*; K.6.3 [**Software Management**]: Software development; D.2.4 [**Software/Program Verification**]: Formal methods, Validation

General Terms

Design, Human Factors

Keywords

Safety, Medical Devices, Interaction Design, Verification, Formal Methods

1. INTRODUCTION

Tools that help manufacturers and regulators rigorously check both prototypes and final designs of devices are vital if use of those devices involve safety issues. Interaction design is an important but under-researched area in this respect. Poor interaction design of medical devices can lead to hazards due to clinicians making mistakes when setting up or using the devices. This is being taken increasingly seriously by regulators, due to the large number of incidents and subsequent recalls of devices.

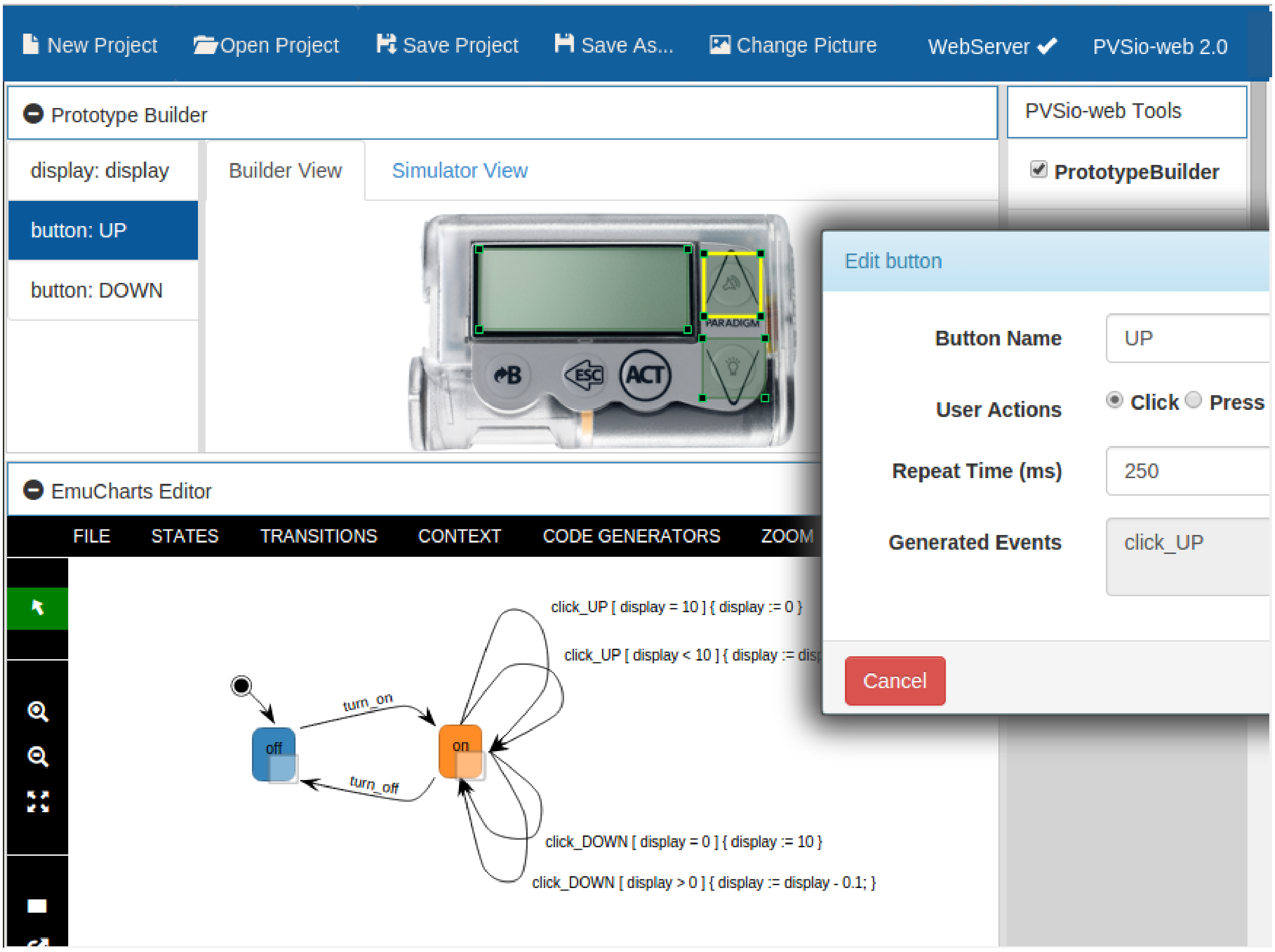
Regulators such as the Food and Drug Administration are promoting the use of model-based engineering techniques to explore design solutions and identify design defects in advance as a solution to these problems [14]. The process is based on the idea of creating mathematical models that describe the behaviour of the real system, and then analysing these models to gain confidence that the real system can operate safely. The advantage of this process is that developers can use it from the early stages of development (a full physical prototype of the device is not needed), and enables rapid and precise exploration of different alternative solutions and different scenarios. We have focussed on extending such techniques to the interaction and interface design of medical devices, and have developed a tool, PVSio-web [10] that supports this process.

2. RAPIDLY GENERATING PROTOTYPES

PVS is an industry standard tool used for verifying systems that need high levels of assurance. It is used within a model-based design process: a mathematical description of the way a device behaves (a model) is checked against similar descriptions of what it should do, using powerful theorem proving and model checking technology. Such techniques re-

MOBIHEALTH 2015, October 14-16, London, Great Britain

DOI 10.4108/eai.14-10-2015.2261720



Button Name: UP

User Actions: ● Click ○ Press

Repeat Time (ms): 250

Generated Events: click_UP

Figure 1: Screenshot of the PVSio-web prototyping environment.

quire a great deal of mathematical expertise however. An issue is in how to make them more accessible and more naturally fit with design processes. PVSio-web is a sophisticated graphical front-end that extends PVS.

PVSio-web makes it easy to create and check realistic prototypes of a device, focussing on the interface that doctors, nurses or patients must interact with. The prototypes have the appearance and behaviour of the real system being analysed. We took a pragmatic approach to the rapid generation of prototypes. A picture of the real system, or if early in the development, a design mock up, is used to represent the prototype's appearance. The developer creates programmable areas over interactive parts, like the buttons and displays. The prototype's behaviour in these programmable areas is given by an underlying mathematical model. Its behaviour is demonstrated or explored by clicking on buttons in the picture, with the results of the interactions seen immediately on the display areas of the picture. PVSio-web, while making prototypes quick and easy to create, also supports model based-engineering of both stand-alone and interoperable medical systems.

The standard view provided by the tool of the realistic looking and interactive prototype interface is designed for domain specialists and end users. It allows them to explore the behaviour of the prototype as they would in the final design. PVSio-web is particularly suitable for presenting mathematical properties, as well as the results of checking them, to engineers and domain specialists in a way that is easy to understand. Through this view, traces of behaviour determined to be problematic can be demonstrated and explored directly on the interface. It is also good for checking assumptions made in the models before analysis. It can also be used in a user-centred design process, allowing early evaluation of prototypes with the people who will have to use the device.

3. EXTENDING TOOL FUNCTIONALITY WITH PLUGINS

The architecture of PVSio-web is designed to be extensible and suitable for supporting stakeholders from wide-ranging backgrounds. It combines different views of the device designed for people with different roles and expertise. This allows a development team and their stakeholders to work together using a single underlying mathematical model because only those that need to see the model do so. The extensible nature means that it is easy to combine PVSio-web with the tools already used. For example an early plug-in [13] allows the interface model to be co-simulated with control software developed separately using traditional tools, such as MathWorks Simulink. Additional plugins enable mathematical analysis with different verification tools, such as Overture [7] and IVY/NuSMV [1].

3.1 Model editor

The behaviour of the prototypes developed using PVSio-web is specified using mathematical descriptions (i.e., models). The models drive the execution of prototypes as well as being the target of checking by the verification tools. A model editor allows formal methods experts to create and edit the underlying models that describe the device's behaviour and do basic sanity checks on it (correct use of types, coverage and disjointness of conditional statements used in function definitions).

3.2 Emucharts plugin and code generation

Developers on the whole do not currently possess the formal methods backgrounds to develop models directly. The models, however, can be created through a graphical editor. The designer works with a graphical notation they are used to and does not need to see the mathematical notation beneath. That notation can still be accessed by verification experts to check requirements mathematically and exhaustively. Developers normally create and edit designs using a graphical notation such as Statecharts. The tool therefore provides an editor for a notation based on Statecharts, called Emucharts. Using Emucharts, designers can declare variables, constants and states of an interactive system. They can also declare transitions between the states, as well as any conditions necessary for the transitions to occur, and how variables change when a transition occurs. Mathematical models are then created automatically from these design drawings in a variety of languages including formal languages like PVS or Ada, and general purpose programming languages like Javascript and C. This process of visually creating a state transition representation of an interactive system makes the model based design paradigm accessible to a variety of developers. Even those with no training in formal methods or computer science can reap its benefits.

4. INTEROPERABLE MEDICAL SYSTEMS

Medical devices have communication capabilities that can be exploited for improving the safety and effectiveness of healthcare systems. For example, consider a clinical situation where an infusion pump is infusing opioids to a patient. A patient monitor analysing the patient's conditions could alert the nurses if the patient enters respiratory depression, and at the same time immediately stop the infusion of the opioid, thus saving the patient's life.

The benefits of interoperable medical devices are clear. However, careful design decisions need to be taken to ensure safety of operation. For example, with reference to the previous example, what happens if the patient monitor is configured incorrectly, is operated incorrectly, or is malfunctioning? It is certainly desirable that the infusion pump operates as safely as in a situation when the pump is not connected to the patient monitor.

In [8], we have introduced a new mechanism in the tool that allows developers to install *virtual communication ports* on device prototypes developed using PVSio-web. Using these virtual ports, device prototypes can be connected to real communication networks, and thus use standard communication protocols to exchange data and commands with other devices connected to the same network. This mechanism therefore enables realistic prototypes of interoperable systems to be created that include both PVSio-web prototypes and real devices (e.g., physical prototypes, or final products). These prototypes are particularly suitable for exploring design requirements and regulatory issues of this new generation of medical systems.

5. APPLICATION AND IMPACT

We have successfully used PVSio-web in several different ways. In collaboration with the FDA, we developed Generic infusion pump prototypes [6] for exploring the definition of essential safety and usability requirements for infusion pumps.

We have also developed demonstrative prototypes for interoperable medical systems with infusion pumps and patient monitors [8]. Each interoperable device can be executed on a different physical machine, and exchange data and commands with the other prototypes using an open communication service for mobile devices. Future work on interoperable devices includes developing new demonstrative prototypes based on the Medical Application Platform [4] architecture. It was developed by the FDA in collaboration with other universities for the analysis of requirements for interoperable medical systems.

We have used PVSio-web to demonstrate previously undetected software defects in commercial medical devices that have safety implications [9, 12]. We have validated mathematical versions of a set of safety/usability requirements for infusion pumps [2, 3, 11]. We have also created training material [5] to help manufacturers, regulators, clinicians, and procurement staff identify design issues, before expensive design commitments are taken and/or before the final product is placed on the market. These demonstrations have been used as part of hospital training programmes to raise awareness about device design issues.

International research groups are exploring applications of our tool in other application domains. For example Honeywell and NASA Langley are using it to check new flight decks, and next generation protocols for air traffic collision avoidance. Universities are using it to teach interactive and safety-critical systems, and explain formal methods technologies to students.

PVSio-web was downloaded over 1,600 times in 2014 alone, and over 1,200 times in the first six months of 2015. It is available for download with the main PVS distribution from SRI International, and from http://www.pvsioweb.org.

6. ACKNOWLEDGEMENTS.

This work was funded by EPSRC through the CHI+MED research project http://www.chi-med.ac.uk (research agreement EP/G059063/1).

7. REFERENCES

[1] J. Campos and M. Harrison. Interaction engineering using the IVY tool. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS09)*, pages 35–44. ACM, 2009.

[2] M. D. Harrison, J. C. Campos, and P. Masci. Reusing models and properties in the analysis of similar interactive devices. *Innovations in Systems and Software Engineering*, 11(2):95–111, 2015.

[3] M. D. Harrison, P. Masci, J. C. Campos, and P. Curzon. Demonstrating that medical devices satisfy user related safety requirements. In *4th International Symposium on Foundations of Healthcare Information Engineering and Systems (FHIES2014)*, 2014.

[4] B. Larson, J. Hatcliff, S. Procter, and P. Chalin. Requirements specification for apps in medical application platforms. In *Proceedings of the 4th International Workshop on Software Engineering in Health Care*, pages 26–32. IEEE Press, 2012.

[5] P. Masci. Design issues in medical user interfaces. `https://www.youtube.com/watch?v=T0QmUe0bwL8`.

[6] P. Masci, A. Ayoub, P. Curzon, I. Lee, O. Sokolsky, and H. Thimbleby. Model-Based Development of the Generic PCA Infusion Pump User Interface Prototype in PVS. In F. Bitsch, J. Guiochet, and M. KaÃćniche, editors, *Computer Safety, Reliability, and Security*, volume 8153 of *Lecture Notes in Computer Science*, pages 228–240. Springer Berlin Heidelberg, 2013.

[7] P. Masci, L. Couto, P. Larsen, and P. Curzon. Integrating the PVSio-web modelling and prototyping environment with Overture. In *13th Overture Workshop, satellite event of FM2015*, 2015.

[8] P. Masci, P. Mallozzi, F. L. De Angelis, G. Di Marzo Serugendo, and P. Curzon. Using PVSio-web and SAPERE for rapid prototyping of user interfaces in Integrated Clinical Environments. In *Verisure2015, Workshop on Verification and Assurance, co-located with CAV2015*, 2015.

[9] P. Masci, P. Oladimeji, P. Curzon, and H. Thimbleby. Tool demo: Using PVSio-web to demonstrate software issues in medical user interfaces. In *4th International Symposium on Foundations of Healthcare Information Engineering and Systems (FHIES2014)*, 2014.

[10] P. Masci, P. Oladimeji, P. Curzon, and H. Thimbleby. PVSio-web 2.0: Joining PVS to Human-Computer Interaction. In *27th International Conference on Computer Aided Verification (CAV2015)*. Springer, 2015. Tool and application examples available at http://www.pvsioweb.org.

[11] P. Masci, R. Ruksenas, P. Oladimeji, A. Cauchi, A. Gimblett, Y. Li, P. Curzon, and H. Thimbleby. The benefits of formalising design guidelines: a case study on the predictability of drug infusion pumps. *Innovations in Systems and Software Engineering*, 11(2):73–93, 2015.

[12] P. Masci, Y. Zhang, P. Jones, P. Curzon, and H. Thimbleby. Formal verification of medical device user interfaces using pvs. In S. Gnesi and A. Rensink, editors, *Fundamental Approaches to Software Engineering*, volume 8411 of *Lecture Notes in Computer Science*, pages 200–214. Springer Berlin Heidelberg, 2014.

[13] P. Masci, Y. Zhang, P. Jones, P. Oladimeji, E. D'Urso, C. Bernardeschi, P. Curzon, and H. Thimbleby. Combining pvsio with stateflow. In *Proceedings of the 6th NASA Formal Methods Symposium (NFM2014)*, Berlin, Heidelberg, April-May 2014. Springer-Verlag.

[14] A. Ray, R. Jetley, P. L. Jones, and Y. Zhang. Model-based engineering for medical-device software. *Biomedical Instrumentation & Technology*, 44(6):507–518, 2010.