Formalisation of a Component Platform

Matthew Fernandez (PhD student), Ihor Kuz, Gerwin Klein
NICTA and University of New South Wales, Sydney, Australia
firstname.lastname@nicta.com.au

Abstract

Developing and maintaining large safety- and security-critical software systems can be complex and error prone when based on a monolithic design. Techniques like formal verification can be used to gain a measure of confidence in the correctness of the system, but applying these to code bases at a scale of millions of lines of code remains infeasible [1]. Using component-based development to design a system from composable elements has the potential to lower the costs of both development and formal reasoning about the properties of the system. While there have been attempts in the past to apply formal methods to component systems, existing work assumes the correctness of the component platform itself [2, 3, 5]. This poster reports on ongoing work on the formal modelling and verification of a component platform for systems' development. There will be no demo with this poster.

The term *component platform* as used here encompasses the definitions of concepts used in a component system specification (*component*, *connection*, etc.), the so-called *glue code* to provide communication between components and other infrastructure required for hosting components at runtime. Assurance in a component system requires trust in three parts of the system: the critical components, the component platform and the underlying operating system. For this work we are modifying an existing component platform that targets the seL4 microkernel [4]. We have constructed formal definitions of the component system concepts and intend to provide a functional specification of the glue code and a machine-checked proof of correctness of the glue code specification.

Any formally established properties of a component platform are inevitably predicated on the correctness of the underlying operating system mechanisms used to implement its functionality. An incorrect assumption on operating system behaviour or an inconsistency between the operating system specification and its implementation is sufficient to invalidate the properties of the component platform. Previously the absence of a formal specification of an operating system has limited the value of a verified component platform. By building on the foundation of seL4, the properties that we assume of the underlying operating system are encapsulated in the kernel specification, which has been proven to correctly abstract the seL4 C implementation. Leveraging the seL4 proof and the outcomes of this work, it is anticipated that establishing an overall system guarantee will be significantly more cost-effective.

References

[1] J. Andronick, D. Greenaway, and K. Elphinstone. Towards proving security in the presence of large untrusted components. In *5th SSV*, Vancouver, Canada, October 2010.

[2] L. de Alfaro and T. A. Henzinger. Interface automata. In *9th FSE*, pages 109–120, Szeged, Hungary, September 2001.

[3] D. Giannakopoulou, C. S. Păsăreanu, and H. Barringer. Assumption generation for software component verification. In *17th ASE*, pages 3–12, Edinburgh, Scotland, UK, September 2002.

[4] G. Klein, K. Elphinstone, G. Heiser, J. Andronick, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood. seL4: Formal verification of an OS kernel. In *22nd SOSP*, pages 207–220, Big Sky, MT, USA, October 2009.

[5] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay, and R. Passerone. A modal interface theory for component-based design. *Fundamenta Informaticae*, 108(1–2):119–149, 2011.

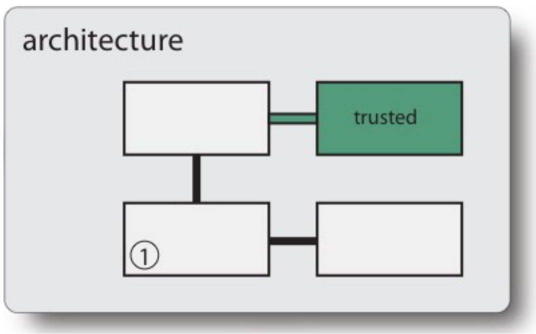
Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program

Formalisation of a Component Platform

Matthew Fernandez, Ihor Kuz, Gerwin Klein

Assurance in a component system requires trust in the underlying operating system, the component platform and the critical components. This work aims to provide correctness guarantees for the component platform.

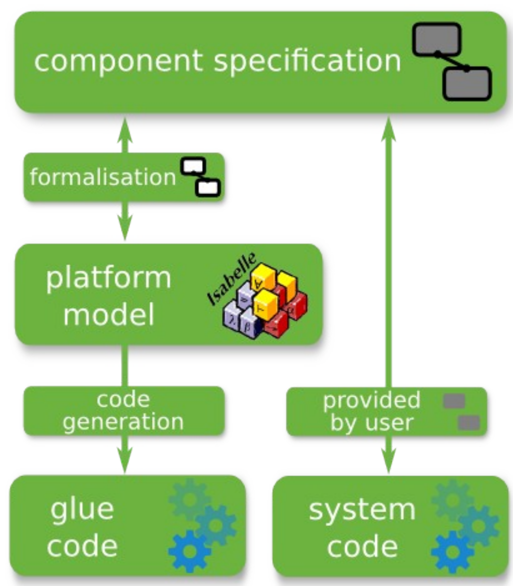


Component-based development

- Enables code reuse
- Facilitates reasoning in isolation
- Makes systems more tractable

A formal platform model

- Definitions and machine-checked proof of correctness in Isabelle/HOL
- Connection to the seL4 specification
- Support for reasoning about the behaviour of components



Current status

- Definitions of component concepts specified
- Definitions of what it means for system descriptions to be well-formed

Next steps

- Functional specification of glue code
- Statement and proof of correctness of glue code
- Glue code generation

From imagination to impact