PROTEST

ProTest
property based testing

European Commission
Information Society and Media

At a Glance

Project title
Property-based testing

Contact person
Prof. John Derrick
University of Sheffield
J.Derrick@dcs.shef.ac.uk

Website
www.protest-project.eu

Total cost / EC contribution
3,639,030 € / 2,709,821 €

Start date /end date
May 2008 / November 2011

Scope

Modern software systems are increasingly complex and interconnected, and a key challenge for the software industry is to deliver robust and reliable systems to customers. **Testing** is central to getting systems right, and in the **ProTest project** we deliver methods and tools for **property-based testing**, which shifts the emphasis from writing single tests to powerful properties that encapsulate aspects of system behaviour. These properties can then be tested in hundreds of randomly generated scenarios, rather than test-case-by-test-case.

The outcomes of the project allow developers to write more effective tests, more efficiently, and so to deliver higher quality software for a lower price, improving their competiveness and thus benefitting the European software industry. Using property-based testing we have advanced testing practice in the European automotive industry, and also found numerous bugs in well-tested industrial concurrent code that had defeated other attempts to find them.

Industrial consortium members are system builders Ericsson and LambdaStream, the SME Quviq, which has commercialised the QuickCheck tool, and consultants and trainers Erlang Solutions, and academic partners are Chalmers University, the Polytechnic University of Madrid and the Universities of Sheffield and Kent.

Advances

Property-driven testing and development is a powerful mechanism for gaining assurance of system reliability and functionality. Property-driven development can be used in a variety of programming languages and systems. The particular platform chosen for initial implementation of the project is Erlang/OTP (Open Telecom Platform), but a crucial aspect of our work was to provide for testing systems written in other languages such as C using the same testing framework.

Since current testing is based on sets of test cases embedded in test suites; we have built tools to aid software developers to extract properties from this data. Also, since many current specifications and models are informal: we have developed specialised property languages to ease the formalisation of specifications. Because all software systems are subject to change and evolution; we have built a refactoring tool to support the evolution of tests and properties in line with the evolution of the system itself.

Because not all properties can be tested in advance of systems being executed, and not all faults are found during testing, we have built a range of tools to support the post hoc examination of trace details for conformance to particular constraints.

Some of the most difficult bugs to track down are related to concurrency. While tracing and property based-testing find some of these, to complement them we have built a model checker that is able to systematically verify properties of concurrent systems.

Positioning in global context

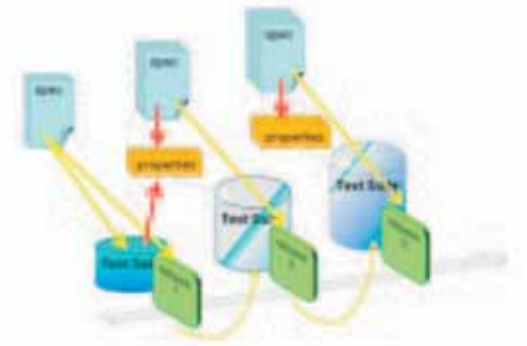
Property-based testing as developed in ProTest offers the promise of more effective testing, as prop-

SEVENTH FRAMEWORK PROGRAMME

erties are substantially more expressive of system requirements than sets of test cases. It promises a paradigm-shift in testing, allowing practitioners to isolate and fix bugs more effectively as well as earlier in the software development process.
ProTest consortium members lead the world in the development of properties for large-scale robust systems written not only in Erlang but also in C, and in sectors including automotive, telecoms, e-business, messaging and data storage.

Contribution to standardization and interoperability issues

There has been an overall project goal to extend the impact of the results outside the Erlang community, both to other programming languages, and also via standards. As one example, we have developed an extension that allows us to test C code with our testing tool QuickCheck. This has been applied in a case study based on AUTOSAR, an open and standardized automotive software architecture developed in the automotive industry. This resulted in car companies promoting QuickCheck testing as the new standard for testing automotive software and Quviq have partnered with SP Technical Research Institute of Sweden to enable software certification in this area.



Target users / sectors in business and society

Our results will help create new economic opportunities for software developers in a variety of sectors – including automotive, embedded, telecoms, messaging and e-business – using a variety of programming languages and platforms. Our property-based testing tool Quviq QuickCheck has had high impact within a number of companies including Ericsson, Gemini Mobile, T-Mobile, Motorola and Basho. Allowing reliable software to be delivered more rapidly and at lower cost will be of particular benefit to the European SME software developer community.

Overall Benefits for business and society

The project provides a development process and tools that ensure dependable quality of service through directly verifying properties of the systems. Our results will allow software developers to bring to market more reliable products on a shorter timescale. Thus, their profitability will be improved, and with that comes an increased ability to compete effectively in a global industry.
These innovations will help to nurture the European software service-provider sector, and help it to compete effectively on the global stage. Initial developments have been in the Erlang/OTP sector, but property-based testing in C is already supported, giving a wide impact to our results.

Examples of use cases

- A large company was faced with the challenge of testing a parser for their domain specific language used to manipulate SIP headers. LambdaStream and a Japanese customer of Quviq had similar challenges in testing parsers for protocol messages. Quviq developed a library that can automatically generate tests from grammar specifications; which simplifies testing of parsers up to a large extent.
- A French customer of Quviq, was faced with the challenge of testing Ejabberd, the worlds leading open source chat software. The difficulty was to specify that the order of certain events was not really fixed. Specifying this in Message Sequence Charts (MSCs) results in non-deterministic MSCs. A special specification language based upon MSCs was developed to enable the specification of messages for this particular scenario.

Achievements

Property-based testing has proved itself in practice: by equipping testers with the tools to track down faults that had proved elusive under other approaches, acknowledged bugs have been fixed.
The key results of our project include:

- The QuickCheck tool supports property-based testing of systems written in Erlang and C.
- McErlang supports model checking for Erlang programs, and is integrated with the PULSE scheduler to provide integrated testing of concurrent systems.
- Wrangler provides refactorings that allow the evolution of properties and tests as systems evolve, as well as assisting in property extraction from existing test suites.
- The Trace Tool Builder incorporates high-level offline log analysis, first available in Onviso. TTB is a part of the standard Erlang distribution.
- The ProTest tools are widely available: QuickCheck as a product from Quviq; Wrangler, McErlang and Trace Tool Builder are open source products.
- Through proof-of-concept work, the project has established the principles of property extraction from existing test suites and UML documents. This and other work is documented in the papers and case study reports written during the project.
- Training materials for the tools and techniques, including videos, presentations and courses.
- Links to all tools, papers and other deliverables are at the project website.