

***Verified Software  
Repository  
Report***

*Jim Woodcock  
University of York  
April 2006*

## ***Goals of the Panel***

- 1. To create a road map for building a repository of verified software, verification tools, benchmarks, challenge problems, models, specifications, and case studies.*
- 2. Coordinating the activities involved in setting up the repository and creating the processes for curating and maintaining the repository.*
- 3. Generating publicity to broaden the participation in the development of the repository.*

## **Goals of the Repository**

- 1. To accelerate the development of verification technology through the development of better tools, greater interoperability, and realistic benchmarks.*
- 2. To provide a focus for the verification community to ensure that the research results are relevant, replicable, complementary, and cumulative, and promote meaningful collaboration between complementary techniques.*
- 3. To provide open access to the latest results and educational material in areas relevant to verification research.*
- 4. To collect a significant body of verified code (specification, derivation, proofs, implementation) that addresses challenging applications.*

5. *Identify key metrics for evaluating the scale, efficiency, depth, amortization, and reusability of the technology.*
6. *Enumerate challenge problems and areas for verification, preferably ones that require multiple techniques.*
7. *Identifying (and eventually standardizing) formats for representing and exchanging specifications, programs, test cases, proofs, and benchmarks, to support tool interoperability and comparison.*
8. *Defining quality standards for the contents of the repository.*

## ***Funding***

- *NSF*
- *NASA EU (7th framework): Open platforms, complex systems*
- *DFG BMBF*
- *SFI*
- *Enterprise Ireland*
- *China*
- *EPSRC*

- *Commercial sponsors*
  - *Microsoft*
  - *Cisco*
  - *Intel*
  - *AMD*
  - *IBM*
  - *TCS*

*Commercial funding could be used to support Prize money, meetings, travel, fellowships.*

## People

*Jim Woodcock, Tiziana Margaria, Juan Bicarregui, Dave Naumann, John Rushby, Joe Kiniry, N. Shankar, Mark-Oliver Stehr, Steve Zdancewic, Jian Zhang, Kokichi Futatsugi, Masami Hagiya, Bernhard Steffen, Zhiming Liu, Mark Utting, Peter Lindsay, Peter O'Hearn, Cliff Jones, Jean-Raymond Abrial, Michael Butler, Wolfgang Paul, Tobias Nipkow, J Moore, Colin O'Halloran, Jim Grundy, John Harrison, Jacob Abraham, Augusto Sampaio, John Hatcliff, Jakob Rehof, Sanjit Seshia, Rustan Leino, Wolfram Schulte, Tom Reps, Brad Martin, Helen Gill, Ralph Wachter, Sol Greenspan, Karl Levitt, David Basin, Egon Boerger, Bertrand Meyer, Xavier Leroy, Ernie Cohen, Greg Morrisett, Byron Cook, Alessandro Cimatti, Cesare Tinelli, Silvio Ranise, Gilles Barthe, Sriram Rajamani, Paritosh Pandya, Virginie Weils, Hardi Hungar, Grigoire Rosu, Cesar Munoz, Klaus Havelund, Gerard Holzmann, Mike Hinchey, Guillaume Brat, Connie Heitmeyer,*

*Ramesh Bhardwaj, Carsten Schuermann, Mark Bickford, Marsha Chechik, Mark Lawford, Alan Hu, Lee Pike, William Farmer, Rance de Long, Josh Guttman, Michael Goldsmith, Markus Muller-Olm, Peter Sestoft, Rupak Majumdar, Pete Manolios, Orna Grumberg, Amir Pnueli, Clark Barrett, Aaron Stump, Gary Leavens, Chris Gill, Patrick Cousot, Anthony Hall, Paul Miner, Kevin Driscoll, Peter Mueller, Joakim von Wright, Henny Sipma, Luca de Alfaro, Andrew Appel, George Necula, Karl Crary, Mary Lou Soffa, Ed Brinksma, Vlad Rusu, Joseph Sifakis, Wolfgang Grieskamp, Alan Hartman, Eli Singerman, Alexandre Petrenko, Sergey Berezin, Dave Musser, Ofer Shtrichman, Natasha Sharygina, Daniel Kroening, Bart Jacobs, Scott Stoller, Alexander Pretschner.*

# Content

## 1. Tools

- *Integrated verification environments: Spec#, ESC/Java2.*
- *Language front-ends*
- *Static Analyzers*
- *Test case generators*
- *Theorem Provers*
- *Model Checkers*
- *Graphical User Interfaces*
- *Program Synthesizers*
- *Integrated Builds*

## 2. Benchmarks

### 3. Case Studies

- *Mondex Case Study Mondex is an electronic wallet.*
- *Electronic voting*

### 4. Interoperability

- *Interchange formats, mappings between models, logics, glue code.*
- *Models: state machines, automata, timed automata, hybrid automata, abstractions.*
- *Language syntax and semantics: Logics, specification languages, programming languages.*
- *Representations of proofs, test cases, counterexamples.*

### 5. Verified libraries: STL, openSSL, core Java, Bouncy castle, openPGP, glibc, GMP,

### 6. Tutorials

7. *Advanced search capability*
8. *Ontologies*
9. *Keywords*
10. *Forums*
11. *Challenge problems: File synchronization, File system, web server, kernel, TCP/IP, SSL, compression, theorem prover kernel, cache consistency, separation kernel, compilers, virtual machines, build tools, fault-tolerant architectures, model reduction for hybrid systems, aspect extraction, scale/parametricity.*
12. *Generic properties: Absence of runtime errors, data consistency, timing behavior, accuracy, type correctness, termination, translation validation, serializability, memory leaks, information hiding, representation independence, information flow.*

## *Planning and Road Map*

- **Years 1 to 5** *Benchmarking, static analysis, and specifications. Individual tool development with opportunistic integration. Small properties of big systems and big properties of small system. Logics for heaps, security, resources, modularity, weak memory models. Multiple code views. Engagement with early adopters.*
  - **Year 1** *One paradigmatic example of code from specs, assertions from code, assertions from test, verification condition generation, and theorem proving.*
  - *Initial case studies.*
  - *User community.*

- **Years 5 to 10** *Integrated tool development, medium-scale/medium-degree verification, loosely coupled to tightly coupled integration, workflows involving multiple tools. Engagement with industry.*
- **Years 10 to 15** *Large-scale applications, tool validation, embedded verification, novel models of programming. Developing a broad user base.*