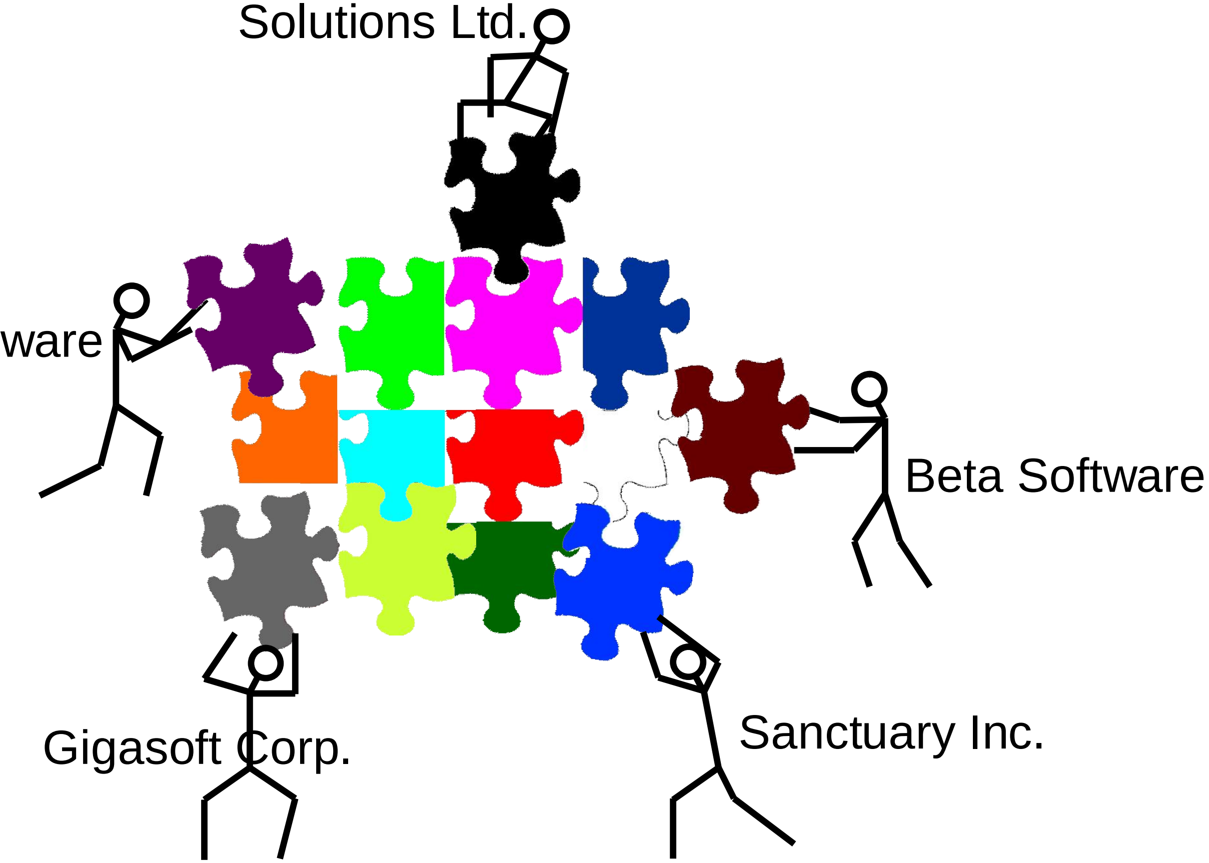
Formal Methods in Industrial Software Standards Enforcement

A. Grinevich, A. Khoroshilov
V. Kuliamin, D. Markovtsev
A. Petrenko, V. Rubanov

ISP RAS, Moscow, Russia

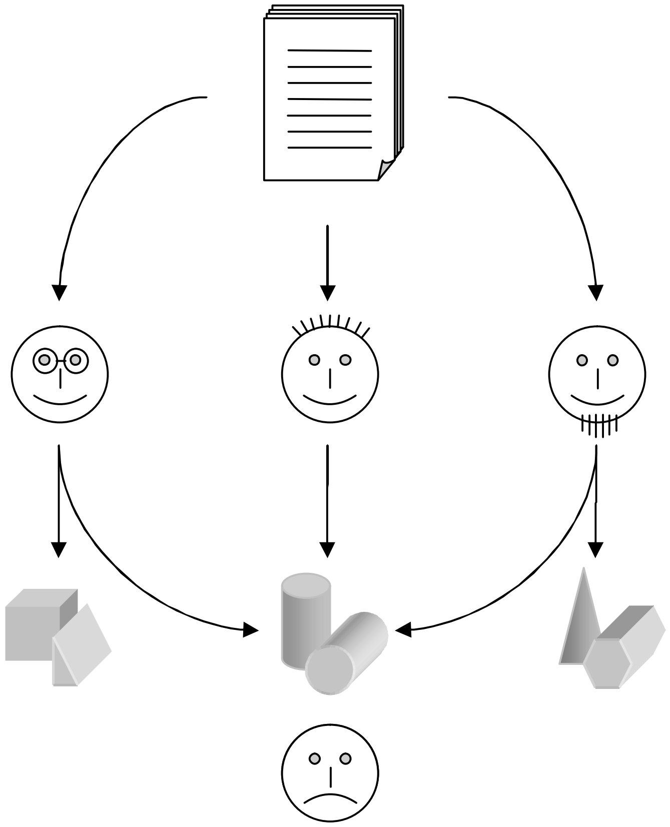
Modern Software Development



How to Make Software Robust?

Interface Standards

- Provide
  - Interoperability
- Require
  - Consistency
  - Completeness
  - Precision



Standard Formalization

- Helps to detect and remove inconsistency, incompleteness, ambiguity
- Conformance test suite

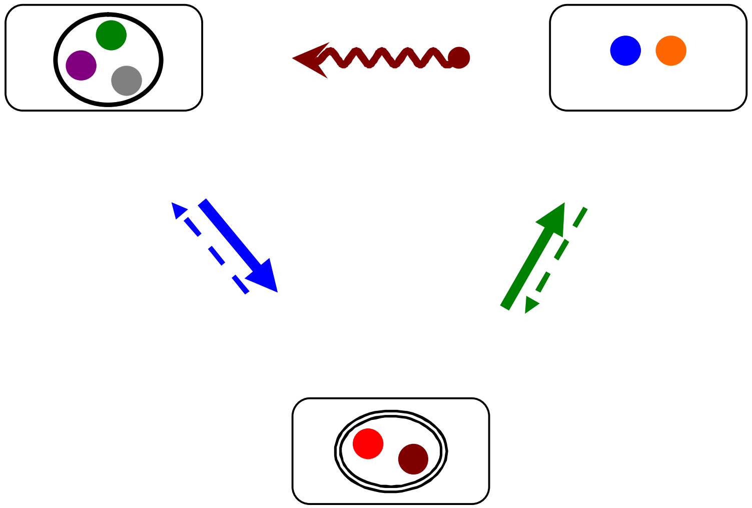
- Technical issues
  - Adequacy of formal models
  - Requirements traceability
  - Component-wise treatment of standard
  - Configurations
- Organizational issues
  - Coordination, skilled staff, etc.
  - Politics

Approach

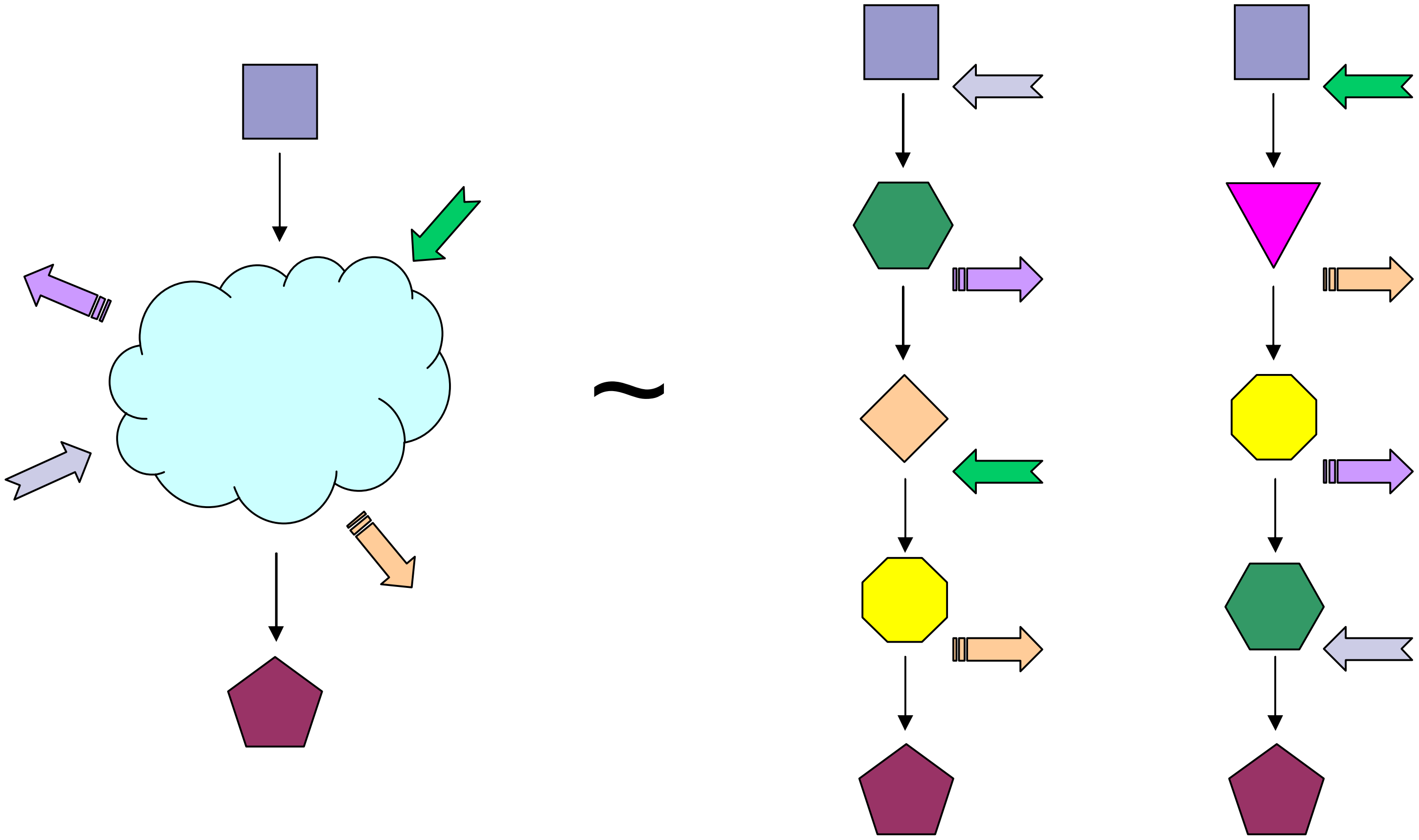
- Light-weight formal methods (model-based testing)
  - Formal specifications
    - Software contracts
    - Explicit links between specifications and standard text
  - Automated conformance test construction
    - Primary test goal – coverage of requirements
- Development process
  - Iterative development
  - Quality control
  - Training
- Propagation of results
  - Communications with standard committee
  - Participation in maintenance of standard

Software Contracts

- Components
- Internal states
  - Invariants
- Operations and ***events***
  - Preconditions
  - Postconditions



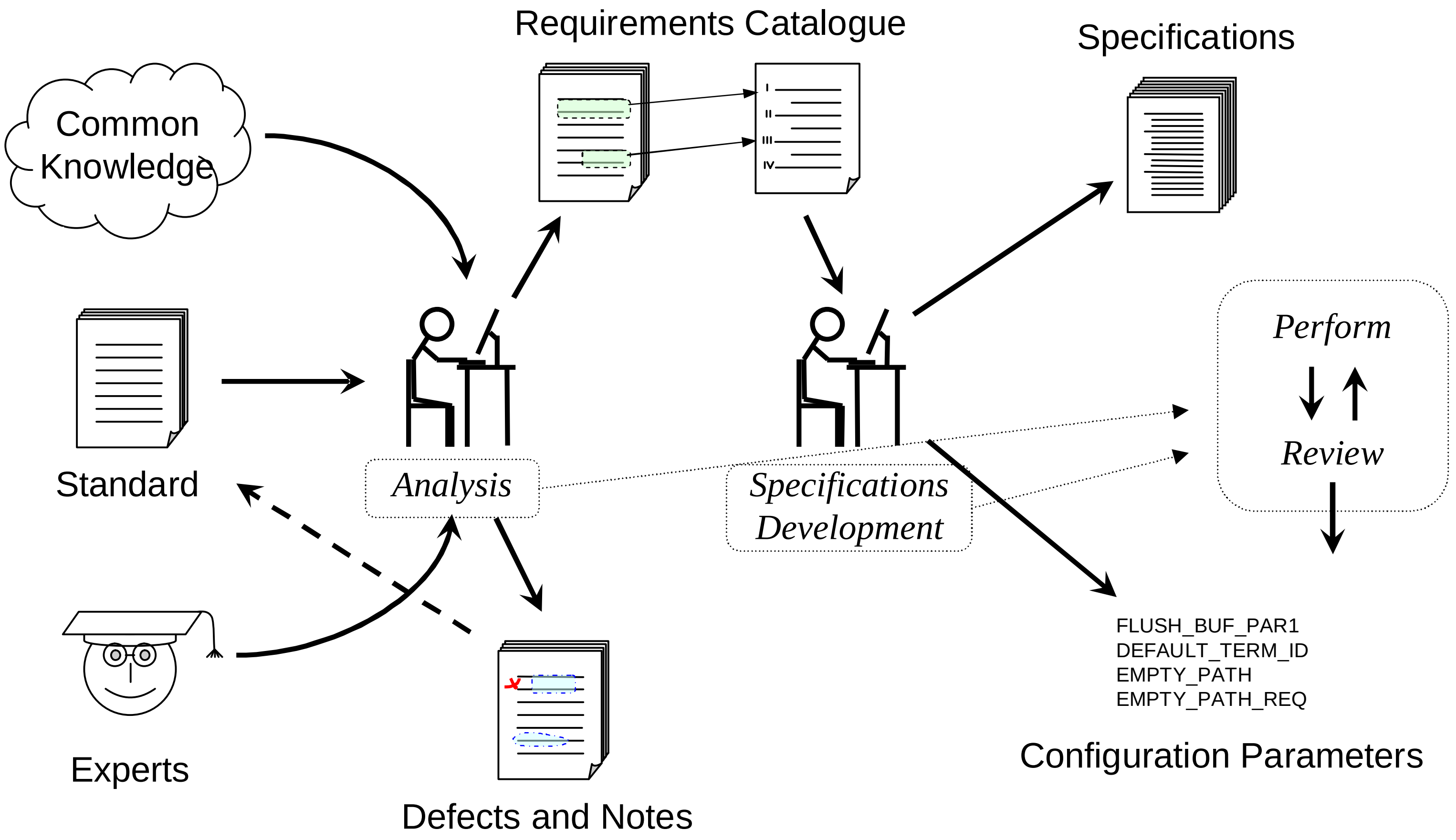
Concurrency Semantics



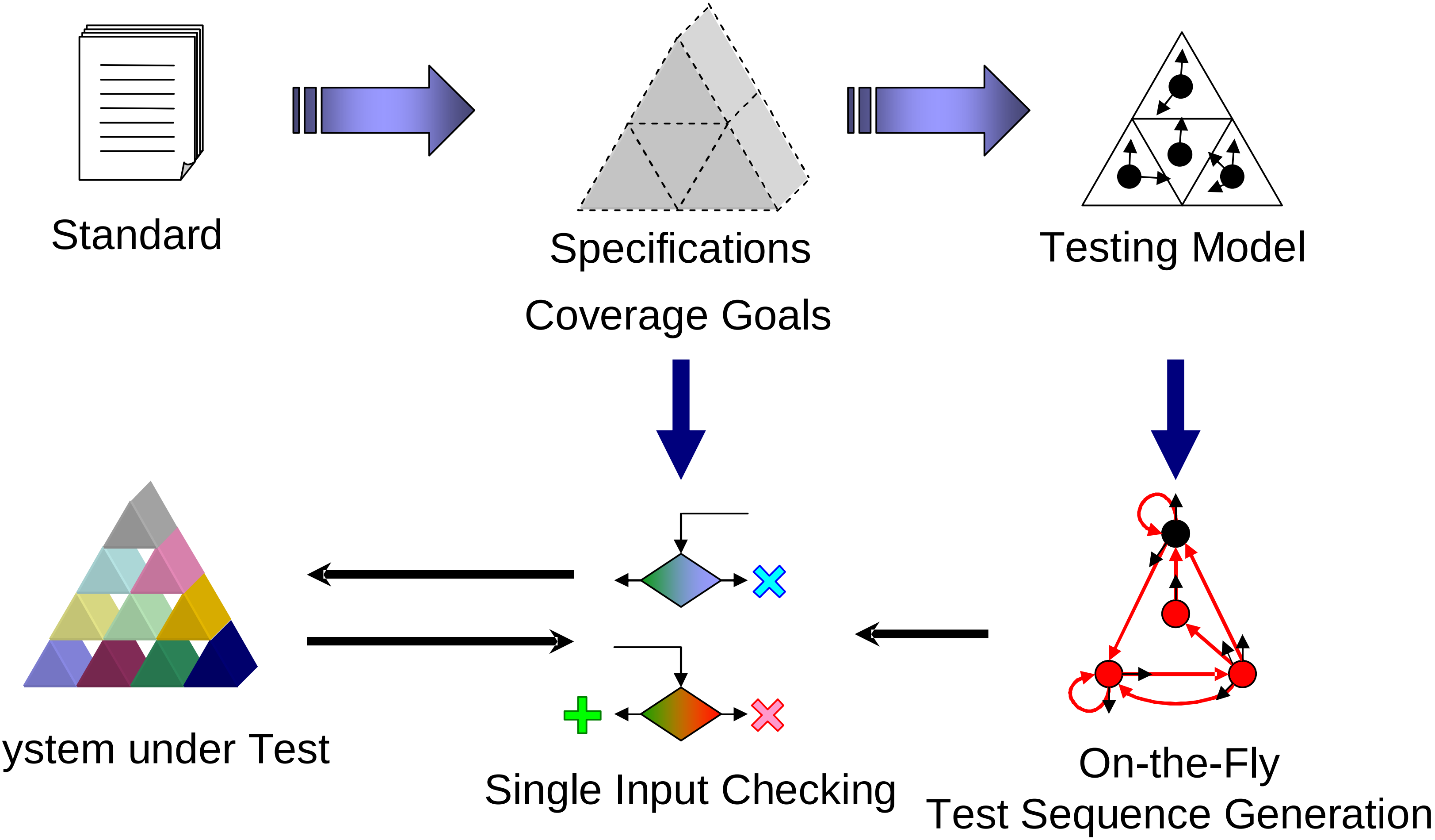
Requirements Traceability

```
specification CString* basename_spec( CString* path )  {
  post  {
    if( @path == NULL )
      REQ( "basename.04",  "If path is null, basename() shall return \".\"",
           equals( basename_spec, create_CString(".") )  );

    if( equals ( @path, create_CString("") ) )
      REQ( "basename.04", "If path is empty string, basename() shall return \".\"",
           equals( basename_spec, create_CString(".") )  );

    if( equals ( @path, create_CString("//") ) )
      REQ( "basename.03",  "If path is \"//\", basename() shall return \"//\" or \"/\"",
           (   equals( basename_spec, create_CString("/") )
            || equals( basename_spec, create_CString("//") ) )  );

    if( basename_all_slash(@path) )
      REQ( "basename.02", "If path contains only slashes, basename() shall return \"/\"",
           equals( basename_spec, create_CString("/") )  );

    CString* expected_basename = basename_model(path);
    REQ( "basename.01.01", "basename() shall return final component of path",
         equals( expected_basename, basename_spec )  );
  }
}
```

Formalization Process



Conformance testing – UniTESK



Standard

Specifications

Coverage Goals

Testing Model

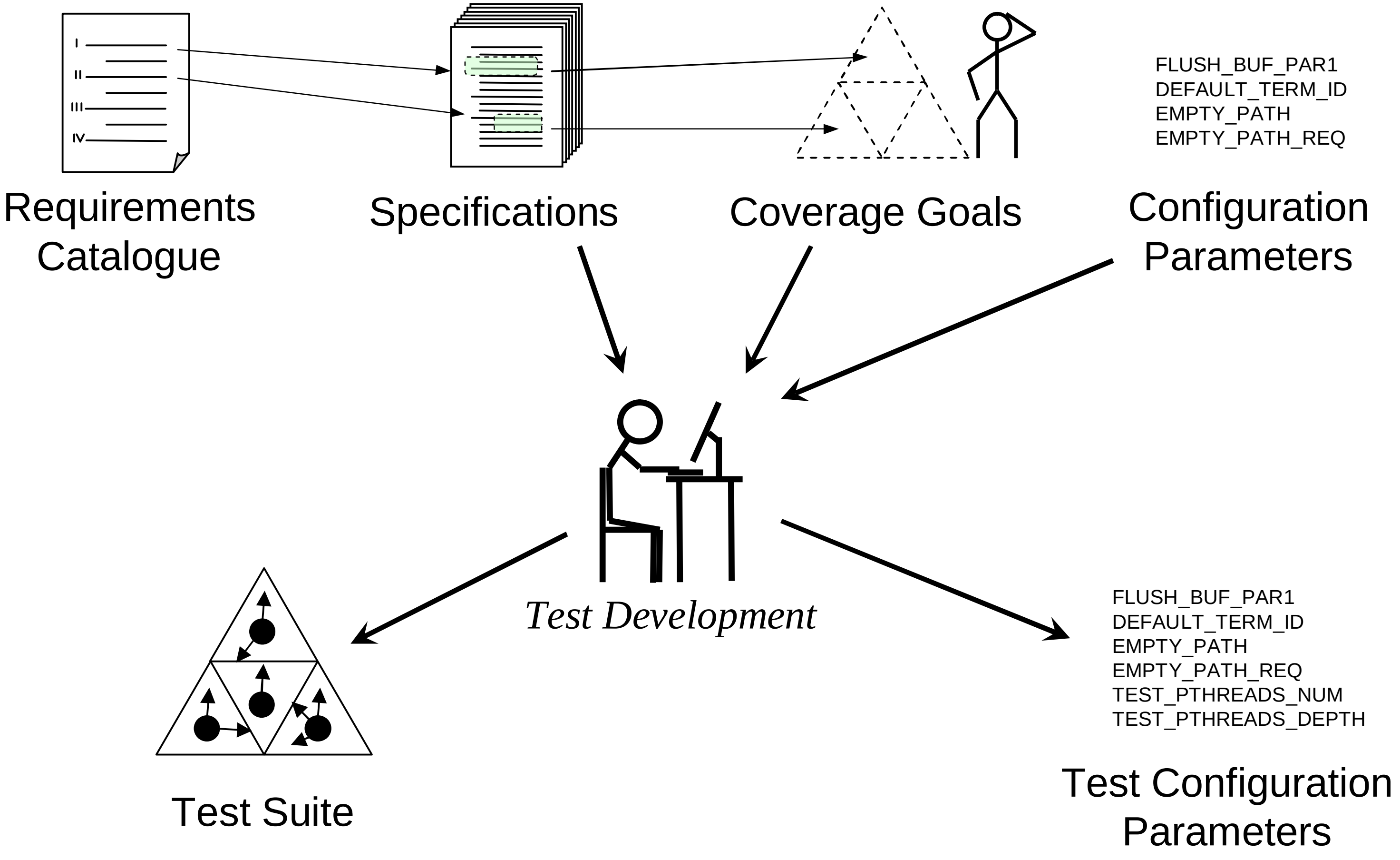
System under Test

Single Input Checking

On-the-Fly

Test Sequence Generation

Test Development Ins and Outs



Requirements Catalogue

Specifications

Coverage Goals

FLUSH_BUF_PAR1
DEFAULT_TERM_ID
EMPTY_PATH
EMPTY_PATH_REQ

Configuration Parameters

Test Development

Test Suite

FLUSH_BUF_PAR1
DEFAULT_TERM_ID
EMPTY_PATH
EMPTY_PATH_REQ
TEST_PTHREADS_NUM
TEST_PTHREADS_DEPTH

Test Configuration Parameters

Case Studies

- Test Development for IPv6 2001-2002
- Formalization of IPMP-2 (ISO/IEC 13818-11:2004) 2004
- Formalization and conformance test development for LSB 3.1 (**OLVER**) 2005-2006

OLVER Project

- **Customer** : Russian Federal Agency for Science and Innovations
- **Task** :

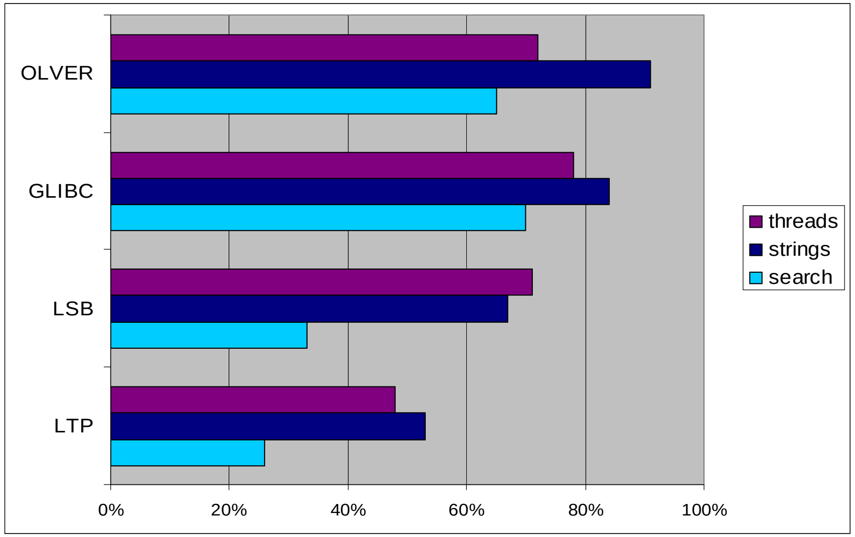
  Develop formal specification of standard reqs and conformance test suite
- **Standard** : Linux Standard Base (LSB) 3.1 Core (ISO/IEC 23360-1:2005)
  - Extensive references (~85%)
    - ISO/IEC 9945-1,2:2003 – POSIX
    - ISO/IEC 9899-1999 – C Language (Library)
    - X/Open Curses, System V Interface Definition, Large File Support
  - > 6000 pages of different standards
  - 1532 functions
  - threads, inter process communication, timers, signals, sockets, RPC, memory management, terminals, file system, large file support, formatted input/output, string manipulation, locales, maths, etc.

Project Progress

Current Results (01.06.2006)

- Standard text analysis
  - ~170 groups of functions
  - ~ 930 functions
  - ~ 10500 primary requirements
  - ~ 40 defects found
- Formalization & test development
  - ~ 740 functions
  - ~ 400 KLOC specifications & tests
- Test quality (code coverage)
  - Higher, than in analogous projects (LTP, LSB TS)
  - Roughly equivalent to implementation-based test suites
- http://www.linuxtesting.org

	LTP	LSB	GLIBC	**OLVER**
threads	48%	71%	78%	72%
strings	53%	67%	84%	91%
search	26%	33%	70%	65%



Application of results

- Active contacts with standard committee (FSG)
  - All defects in standard during last 3 months are reported by OLVER team
- Future integration with official LSB conformance test suite

Conclusion

- Long-history standards are stable enough to get significant and practically important benefits from formalization
- Light-weight formal methods are capable to manage with such huge tasks
- Most issues are common with generic huge projects (without formal methods)
  - Iterative development process
  - Adequate planning
  - Project repository
- Necessary skills can be trained

Contacts

- Linux Verification Center web site
  http://www.linuxtesting.org
- UniTesK projects web site
  http://www.unitesk.com
- Group leader
  Alexander K. Petrenko
  petrenko@ispras.ru

References

1. I. Bourdonov, A. Kossatchev, V. Kuliamin, and A. Petrenko. *UniTesK Test Suite Architecture.* Proc. of FME 2002. LNCS 2391, pp. 77-88, Springer-Verlag, 2002.
2. V. Kuliamin, A. Petrenko, N. Pakoulin, I. Bourdonov, and A. Kossatchev. *Integration of Functional and Timed Testing of Real-time and Concurrent Systems.* Proc. of PSI 2003. LNCS 2890, pp. 450-461, Springer-Verlag, 2003.
3. V. Kuliamin, A. Petrenko, A. Kossatchev, and I. Burdonov. The UniTesK Approach to Designing Test Suites. Programming and Computer Software, Vol. 29, No. 6 , 2003, pp. 310-322. (Translation from Russian)
4. V. Kuliamin, A. Petrenko. *Applying Model Based Testing in Different Contexts.* Proceedings of seminar on Perspectives of Model Based Testing, Dagstuhl, Germany, September 2004.
5. V. Kuliamin. *Model Based Testing of Large-scale Software: How Can Simple Models Help to Test Complex system*. Proc. ISOLA'2004, Pathos, Cyprus, 2004.
6. V. Kuliamin, N. Pakoulin, A. Petrenko. *Practical Approach to Specification and Conformance Testing of Distributed Network Applications*. In M. Malek, E. Nett, N. Suri, eds. Service Availability. LNCS 3694, pp. 68–83, Springer-Verlag, 2005.

Thank you!