

Формализация интерфейсных стандартов и автоматическое построение тестов соответствия

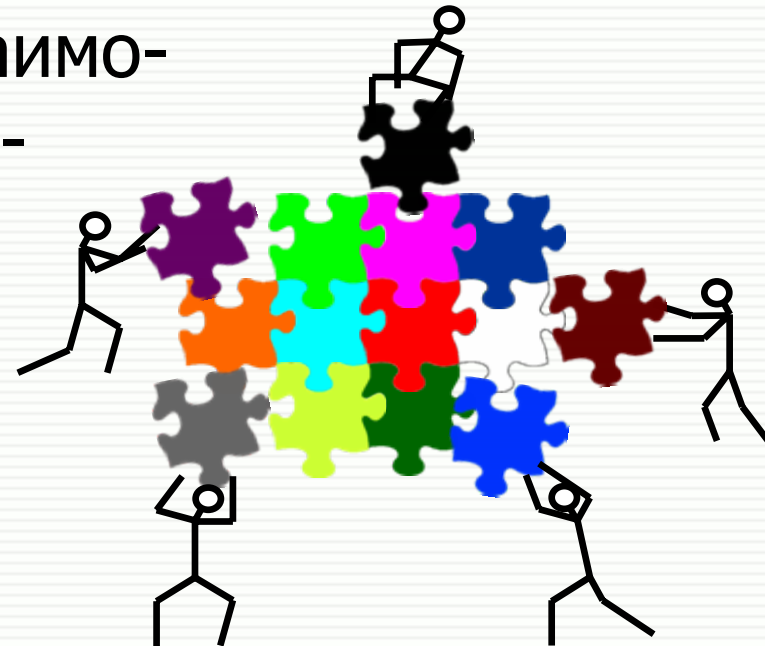
Владимир Рубанов

менеджер по проектам

ИСП РАН

Интерфейсные стандарты

- Фиксируют интерфейсы между различными программными системами
- Имеют важнейшее значение для разработки современного сложного ПО
- Обеспечивают взаимодействие компонентов от разных разработчиков и целых производителей

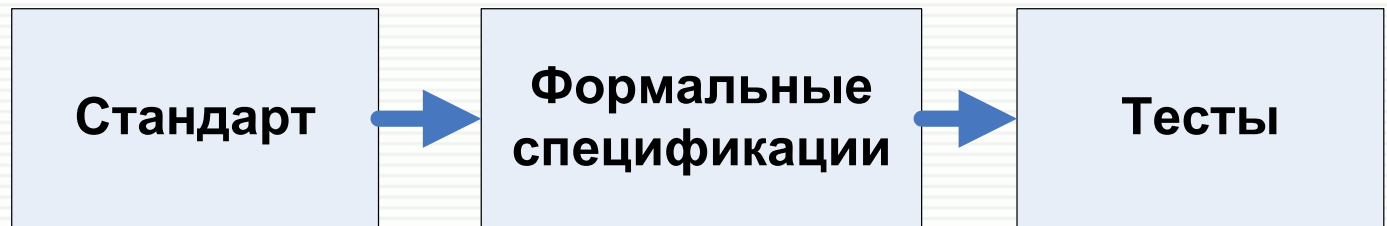


Основные проблемы

- Представленный на естественном языке текст стандарта сложно, а иногда невозможно «правильно» прочитать:
 - присутствуют неточности, допускающие двусмысленные трактовки;
 - некоторые аспекты описываются неполно или даже противоречиво.
- Отсутствуют средства, позволяющие легко проверять конкретные реализации на соответствие стандартам.

Методы решения

- **Формализация** - представление требований стандарта в строгом, машинно-читаемом виде на специализированном языке.
- **Тестирование** – разработка средств автоматического тестирования реализаций на предмет выполнения требований стандартов.



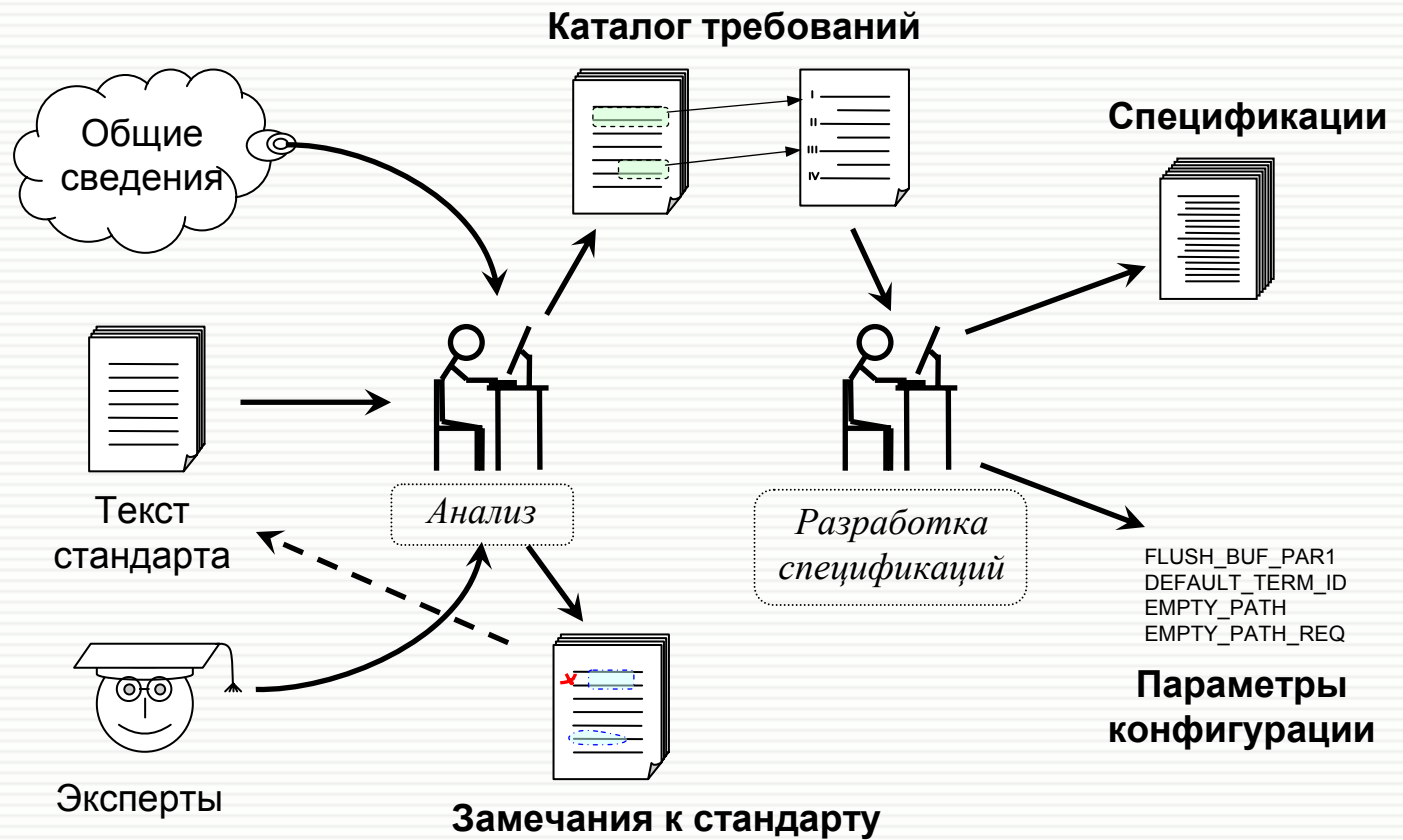
Основные требования

- Разделение работы.
- Инкрементальность процесса.
- Прослеживаемость требований.
- Механизмы синхронизации изменений.
- Управление конфигурациями стандарта.

Технология ИСП РАН

1. Декомпозиция стандарта
2. Выделение элементарных требований
3. Разработка формальных спецификаций
4. Разработка тестовых сценариев
5. Автоматическая генерация тестов

Формализация стандарта



Спецификация на SeC

```
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(".") ) );

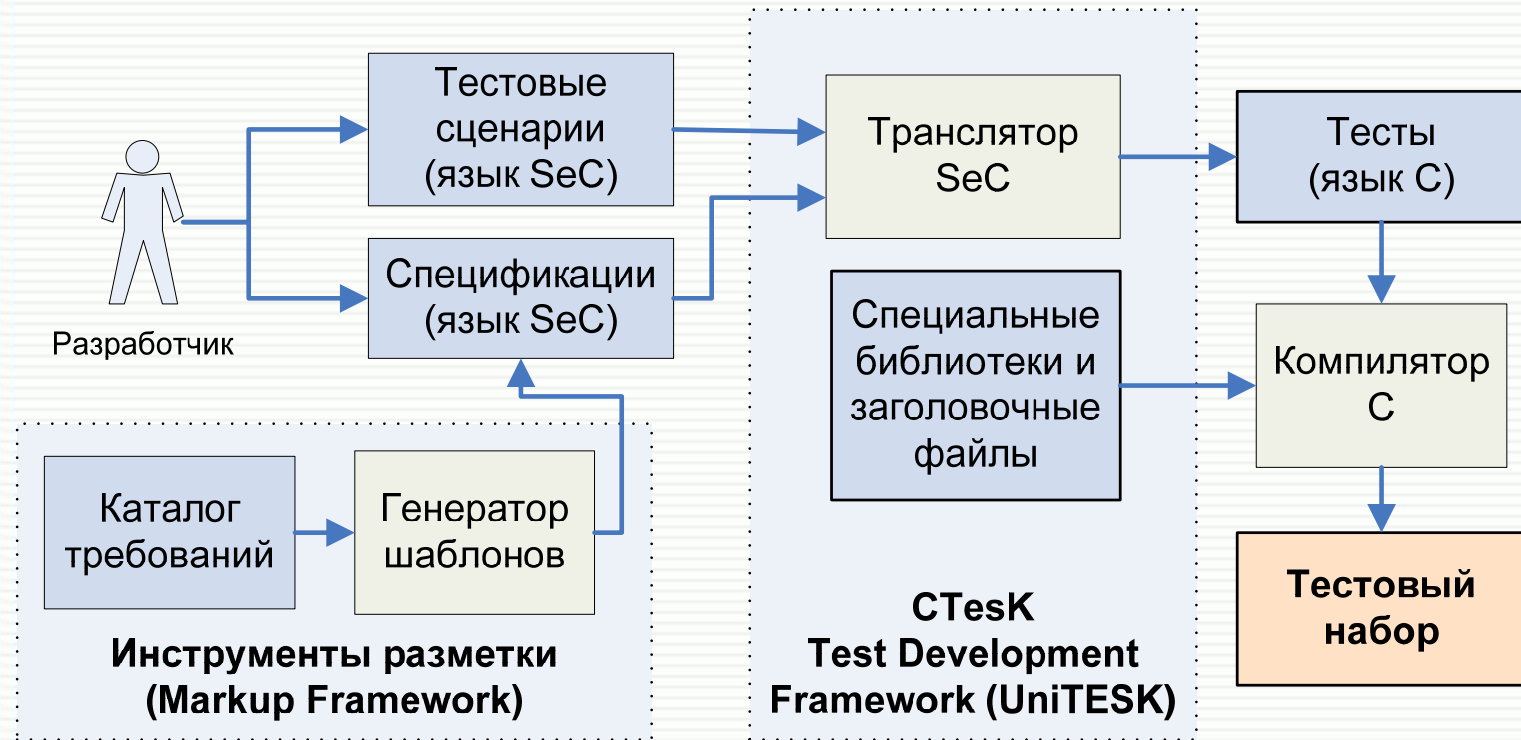
    if( equals ( @path, create_CString("") ) )
      REQ( "basename.05", "If path is empty string, basename() shall
          return \".\" ,
          equals( basename_spec, create_CString(".") ) );

    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 ) );
  }
}
```

Разработка тестов: UniTESK



Основные черты UniTESK

- **Контрактные спецификации:** предусловия и постусловия операций, инварианты данных.
- Спецификации описываются на **расширениях языков** программирования.
- Основной **критерий качества** тестирования – покрытие ветвей выполнения постусловий.
- Построение **цепочки тестовых вызовов** во время выполнения теста; логика построения - обход конечного автомата.
- Спецификации и конечный автомат рассматриваются как **модели реализации**.

Практические применения

- Десяток производственных проектов для закрытых спецификаций заказчика.
- Проекты по открытым стандартам:
 - Разработка тестов для стандарта IPv6
 - Формализация стандарта IPMP-2 (ISO/IEC 13818-11:2004)
 - Формализация и разработка тестов для стандарта LSB Core 3.1 (ISO/IEC 23360)

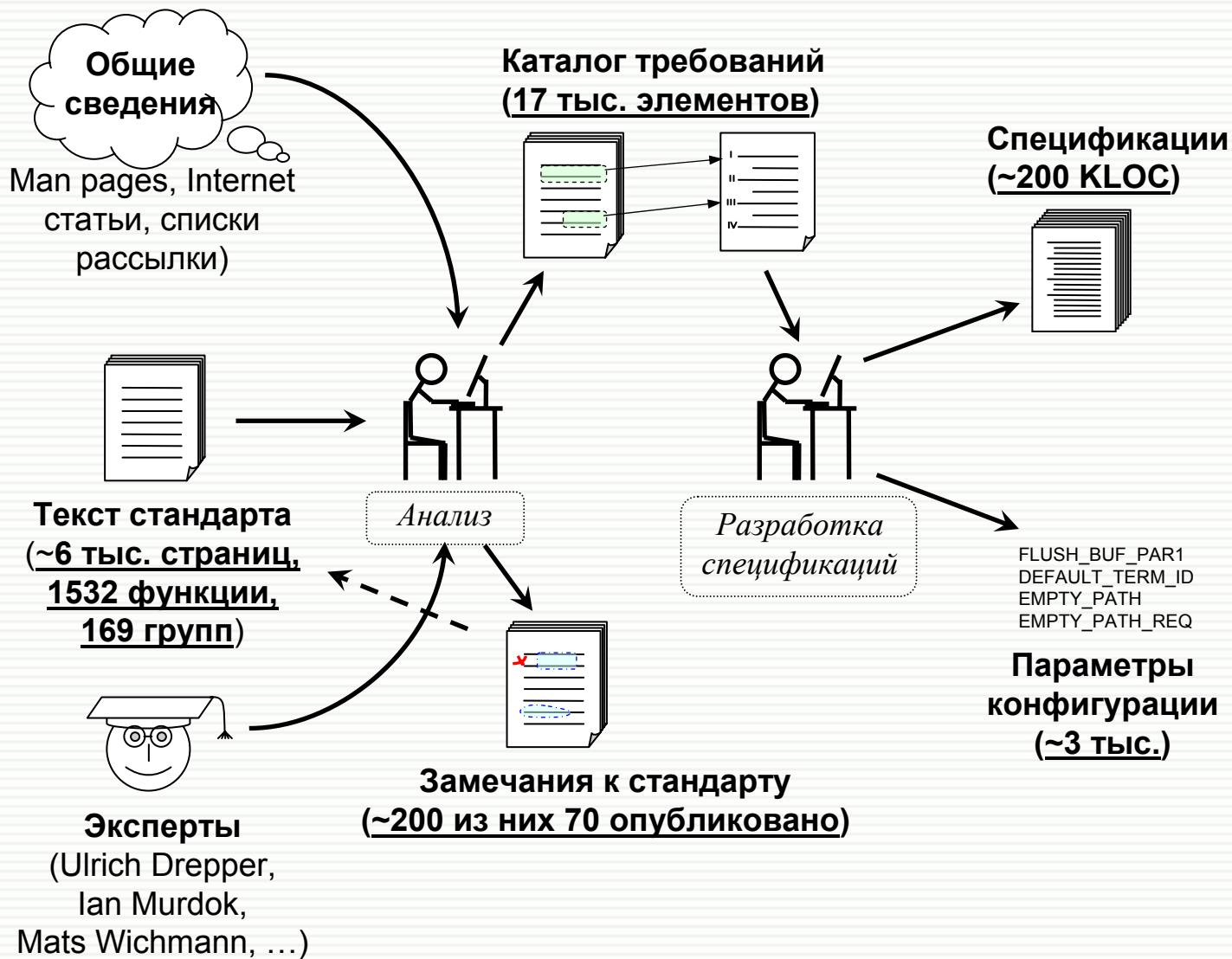
Проект OLVER

- Заказчик: Федеральное Агентство по Науке и Инновациям.
- Объем – 15 млн. руб (~\$550 тыс.).
- OLVER (Open Linux VERification) - формализация стандарта LSB Core 3.1 в отношении поведения интерфейсов основных системных библиотек Linux и построение соответствующего тестового набора.

Linux Standard Base (LSB) Core 3.1

- Специфицирует поведение интерфейсов основных системных библиотек Linux (всего 1532 функции)
 - threads, inter process communication, timers, signals, sockets, RPC, memory management, terminals, file system, large file support, formatted input/output, string manipulation, locales, maths, и т.д.
- Опирается на стандарты Single UNIX Specification (POSIX), ISO C99, SVID и др.
- Более 6000 страниц текста

Формализация LSB Core 3.1



Тестовый набор OLVER

- Проверка требований стандарта для более, чем **1400** функций.
- Среда конфигурации и запуска.
- Различные отчеты по результатам тестирования:
 - о нарушенных требованиях;
 - о покрытии требований стандарта;
 - визуальное представление трассы тестирования для локализации ошибок.

Сравнение с другими тестами

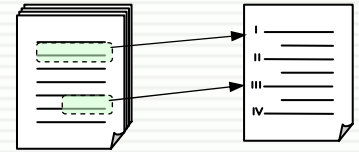
Подсистема	Всего ф-ций	GLIBC	LSB	LTP	OLVER
fs	54	39	33	42	51
io	128	78	92	73	116
locale	70	56	60	19	70
math	271	259	31	39	271
memory	18	12	14	17	18
ncurses	275	0	59	3	239
process	106	61	77	81	97
pthread	87	79	66	80	83
signal	37	25	24	30	34
socket	101	34	2	56	88
system	44	20	15	23	43
time	33	23	26	27	31
util	308	187	144	76	304
Всего	1532	873	643	566	1445

Контакты

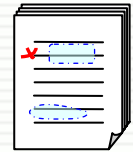
- Институт системного программирования РАН
<http://ispras.ru/>
- Центр верификации ОС Linux
<http://linuxtesting.org/>
<http://linuxtesting.ru/>
info-lvc@linuxtesting.org
- Владимир Рубанов
vrub@ispras.ru
+7-495-912-0754 доб. 4454

Результаты формализации

- Каталог элементарных требований стандарта.



- Замечания к тексту стандарта.



- Формальные спецификации требований.



- Конфигурационная система стандарта.

```
FLUSH_BUF_PAR1  
DEFAULT_TERM_ID  
EMPTY_PATH  
EMPTY_PATH_REQ
```