Верификация компьютерных систем

Петренко Александр Константинович,
Институт системного программирования РАН (ИСП РАН)
www.ispras.ru
http://ispras.ru/ru/se

Юбилейная конференция ИСП РАН-20 лет
24 января 2014 года

Верификация (от лат. *verus* — «истинный» и *facere* — «делать»)

- **Верификация** — это проверка соответствия конечного продукта предопределённым эталонным требованиям (спецификациям)
- «Верификация в малом»:
  - Тройка Хоара P {C} Q
  - Метод Флойда
- Верификация модулей
  - Пред- и постусловия функций
  - Инварианты типов данных/объектов

Задача верификации функции f
pre_f (arg) => post_f (f(arg))

```
specification class AccountSpecification
{
specification void deposit(int sum)
{
  pre { return (0 < sum )&& (balance<=
                          Integer.MAX_VALUE-sum); }
  post { return balance == pre balance + sum; }
}
}
```

Теория vs. Практика

Источники сложности верификации реального программного модуля:

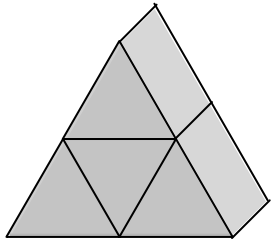
- Большой размер
- Наличие вызовов других модулей (размер замыкания модуля >> размера модуля)
- Неформализованность некоторых аспектов языков программирования
- Отсутствие инструментов
- Отсутствие подготовленных специалистов и др.

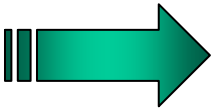
Практика

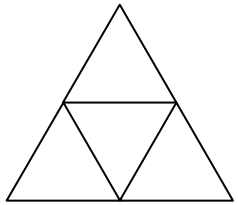
Сведение верификации к тестированию:

- Выбор конечного набора входных данных
  - вручную
  - (?) автоматическая генерация входных данных на основе анализа исходного кода
- Ручное вычисление «эталонных» результатов
- Оценка тестового покрытия по покрытию исходного текста программы

Генерация и выполнение теста в UniTESK (1994-2000)





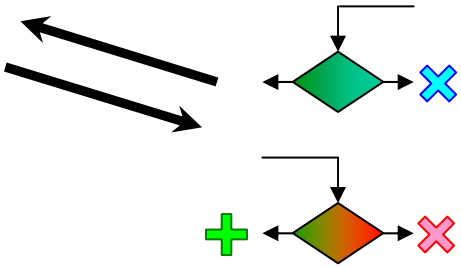


Модель поведения

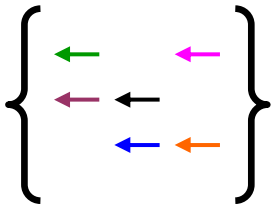
Модель теста

Модель полноты

Целевая систем



Тестовый оракул



Тестовые стимулы

Приложения UniTESK

Экспериментальные приложения технологии **1999-2005**:

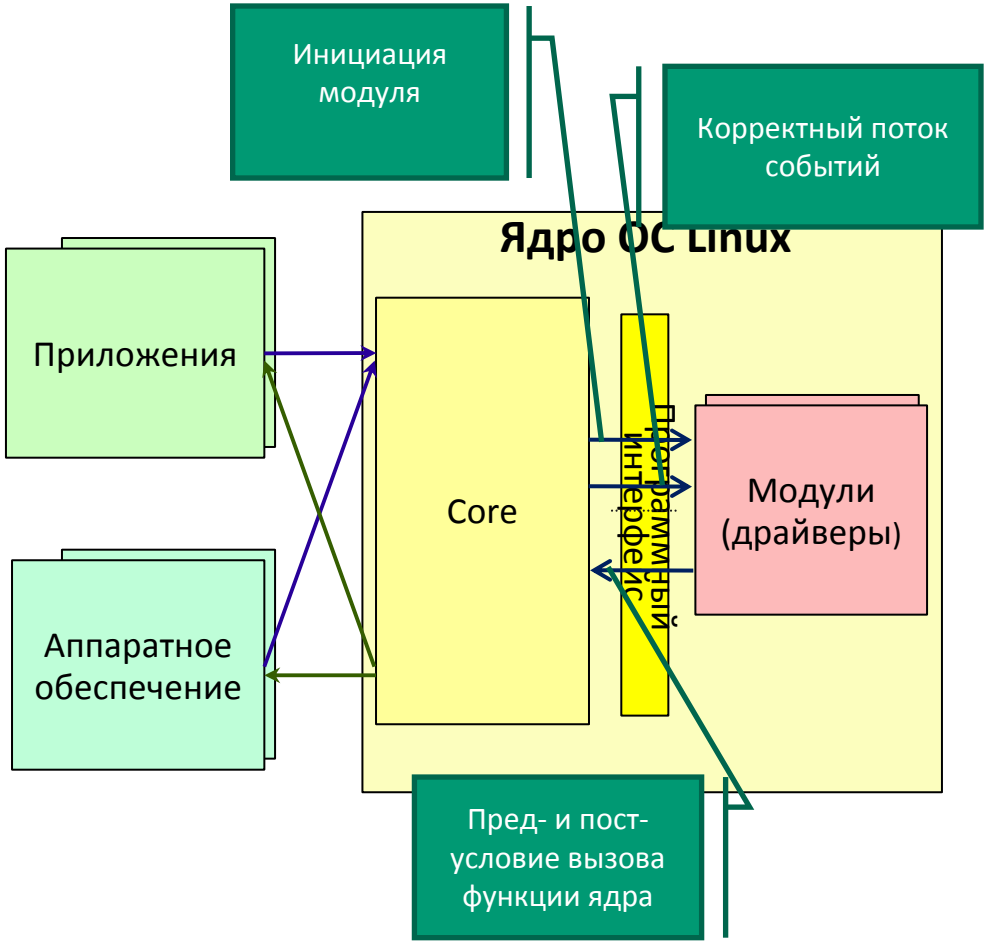
- **Microsoft Research ->** Телекоммуникационные протоколы
- **Luxoft ->** Банковские приложения
- **Intel+Daimler Chrysler ->** Компиляторы
- Микропроцессоры

Промышленные приложения технологии **2005-2013**:

- **Вымпелком ->** Распределенные системы (инфраструктура биллинга и информационных потоков предприятия в целом)
- **НИИСИ ->** Тестирование ОСРВ (ОС2000/3000, стандарты POSIX, ARINC-653)
- **НИИСИ+МЦСТ ->** Тестирование микропроцессоров
- **Минобрнауки+Linux Foundation+Nokia ->** Автоматизированная поддержка стандарта библиотек Linux-платформ (LSB)

Верификация модулей ядра ОС LInux

- Новый вид программного контракта
- ? Источники спецификации контракта
- ? Способ автоматической верификации

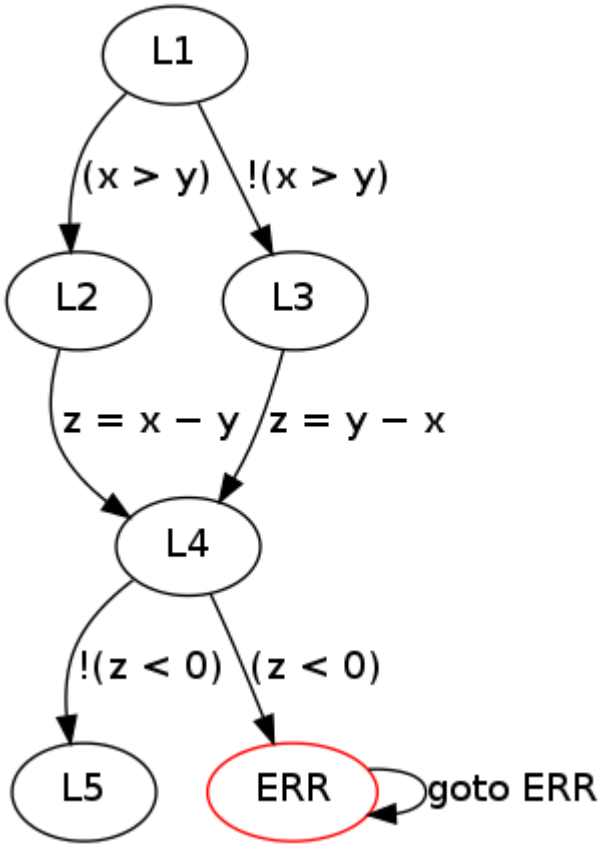


Постановка задачи верификации

```
L1: if (x > y)
L2:                    z = x – y;
          else
L3:                    z = y – x;

L4: if (!(z >= 0))
ERR: goto ERR;
L5:
```

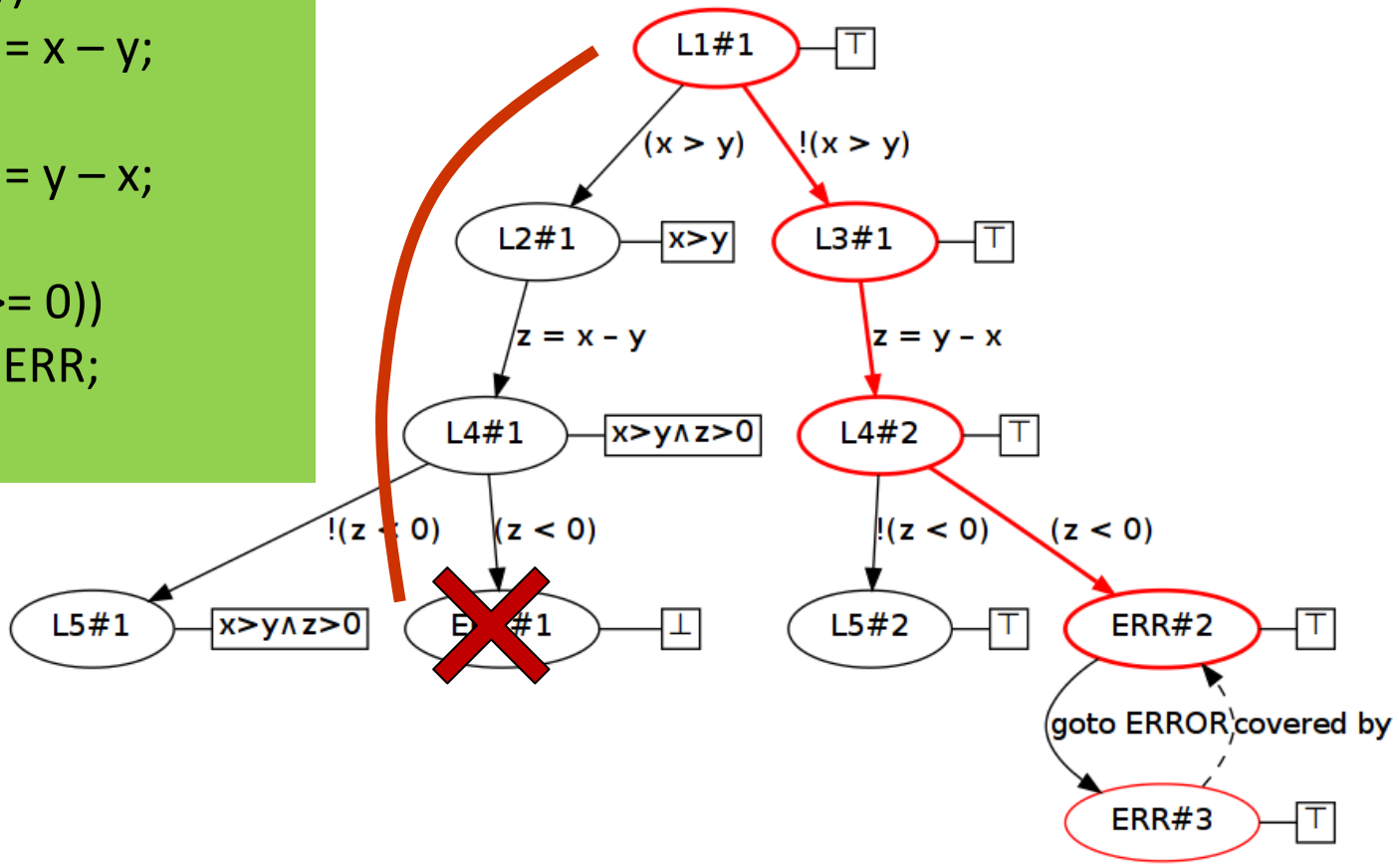
Граф потоков управления



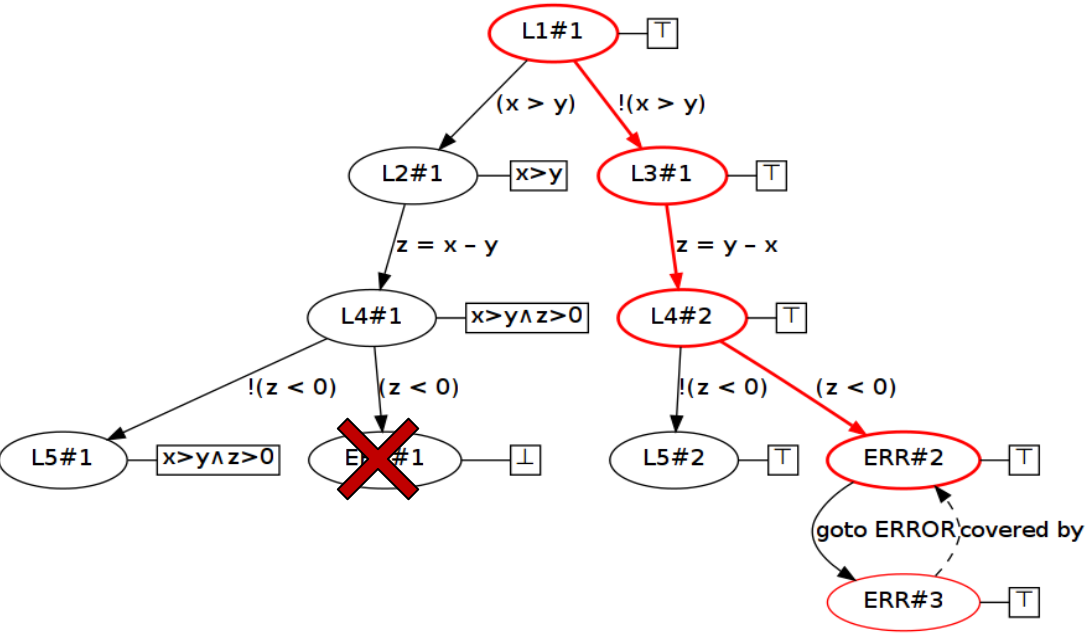
Решение задачи достижимости

```
L1: if (x > y)
L2:         z = x – y;
    else
L3:         z = y – x;

L4: if (!(z >= 0))
ERR: goto ERR;
L5:
```

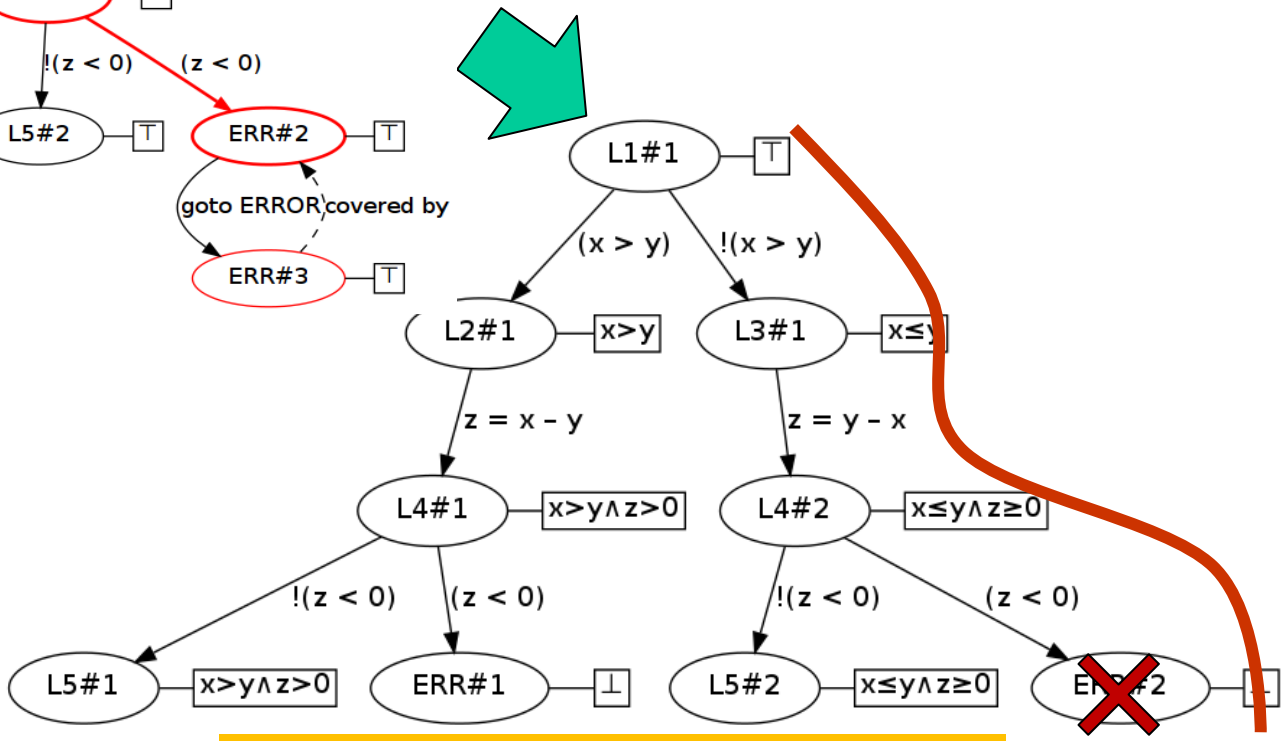


Уточнение абстракции



новые предикаты:

$(x \le y), (z \ge 0)$



$\neg(x_1 > y_1) \square (z_2 = y_1 - x_1) \square (z_2 < 0)$

Linux Driver Verification

Результаты верификации:

- Открытая система верификации: http://linuxtesting.org/result/ldv - найдено > 140 ошибок

Инструменты верификации:

- BLAST (UC Berkeley + ИСП РАН)
- CPAchecker (Univ. Passau + ИСП РАН)

Verification Tool Competition 2014: Победители в отдельных категориях

BitVectors

1. LLBMC
2. CBMC
3. CPAchecker

Concurrency

1. CSeq-Lazy
2. CSeq-MU
3. CBMC

ControlFlow

1. CPAchecker
2. FrankenBit
3. LLBMC

DeviceDrivers64

1. BLAST 2.7.2
2. UFO
3. FrankenBit

HeapManipulation

1. CBMC
2. Predator
3. LLBMC

MemorySafety

1. CPAchecker
2. LLBMC
3. Predator

Recursive

1. CBMC
2. Ultimate Automizer
3. Ultimate Kojak

Sequentialized

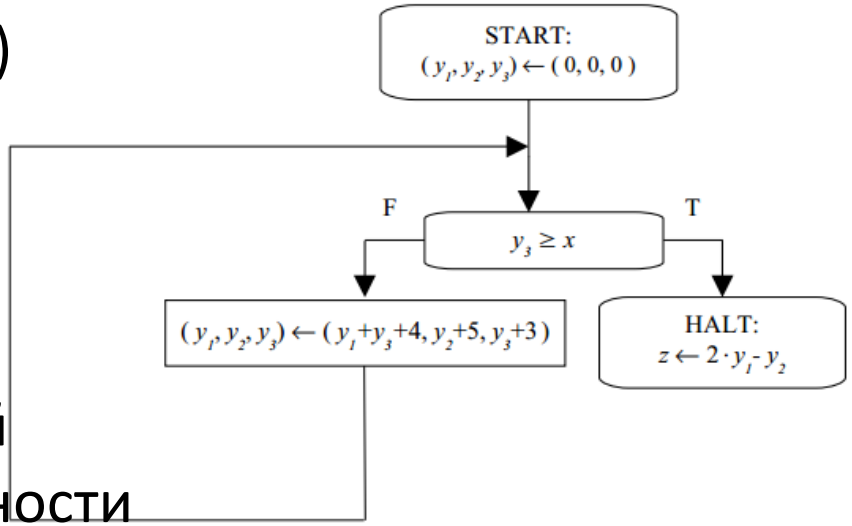
1. ESBMC 1.22
2. CBMC
3. LLBMC

Simple

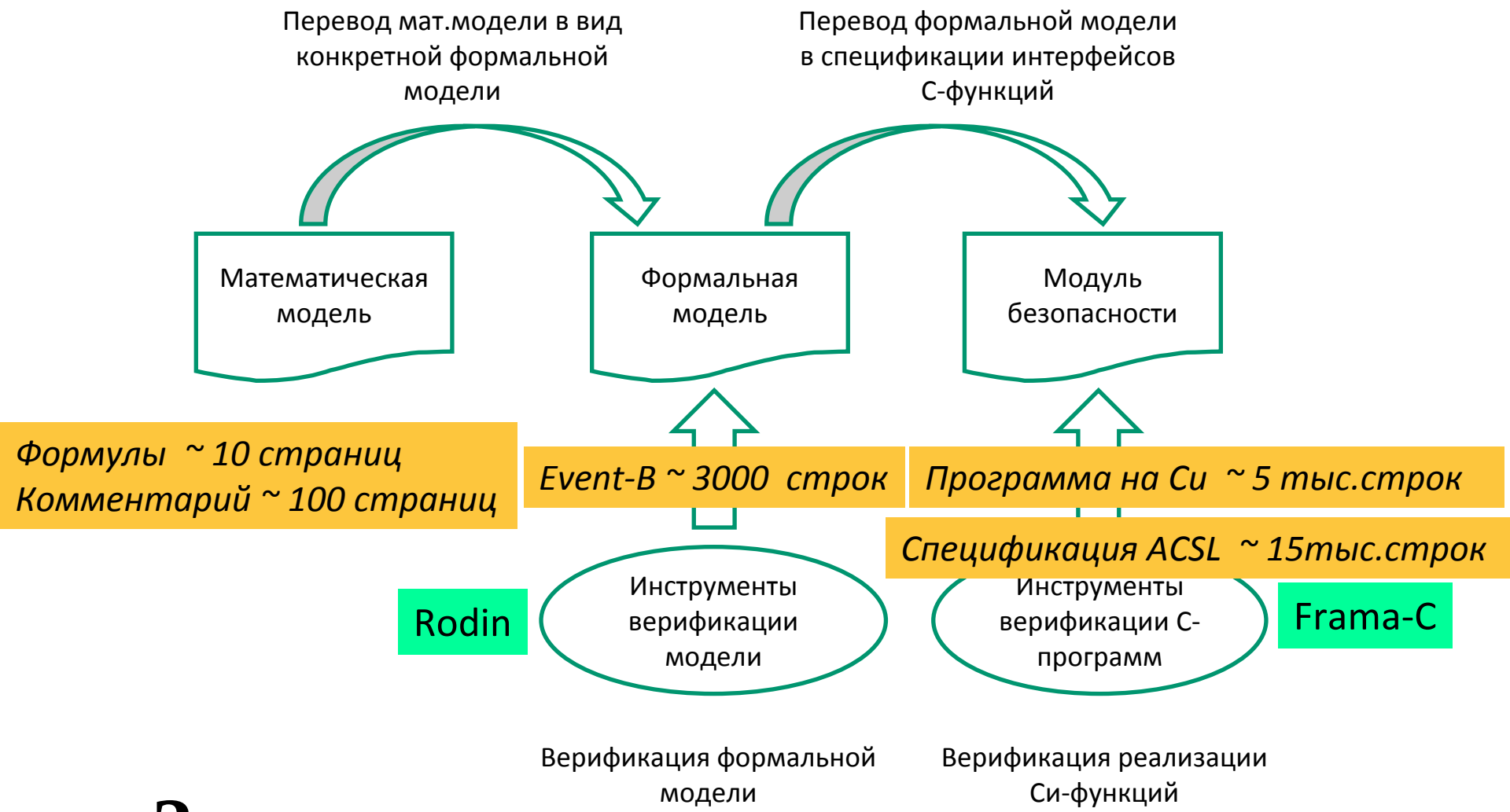
1. CPAchecker
2. UFO
3. CBMC

Верификация (от лат. *verus* — «истинный» и *facere* — «делать»)

- **Верификация** — это проверка соответствия конечного продукта предопределённым эталонным требованиям (спецификациям)
- «Верификация в малом»:
  - Тройка Хоара P {C} Q
  - Метод Флойда
- **РусБИТех:** Задача дедуктивной верификации модуля безопасности ОС Linux



Верификация модуля безопасности



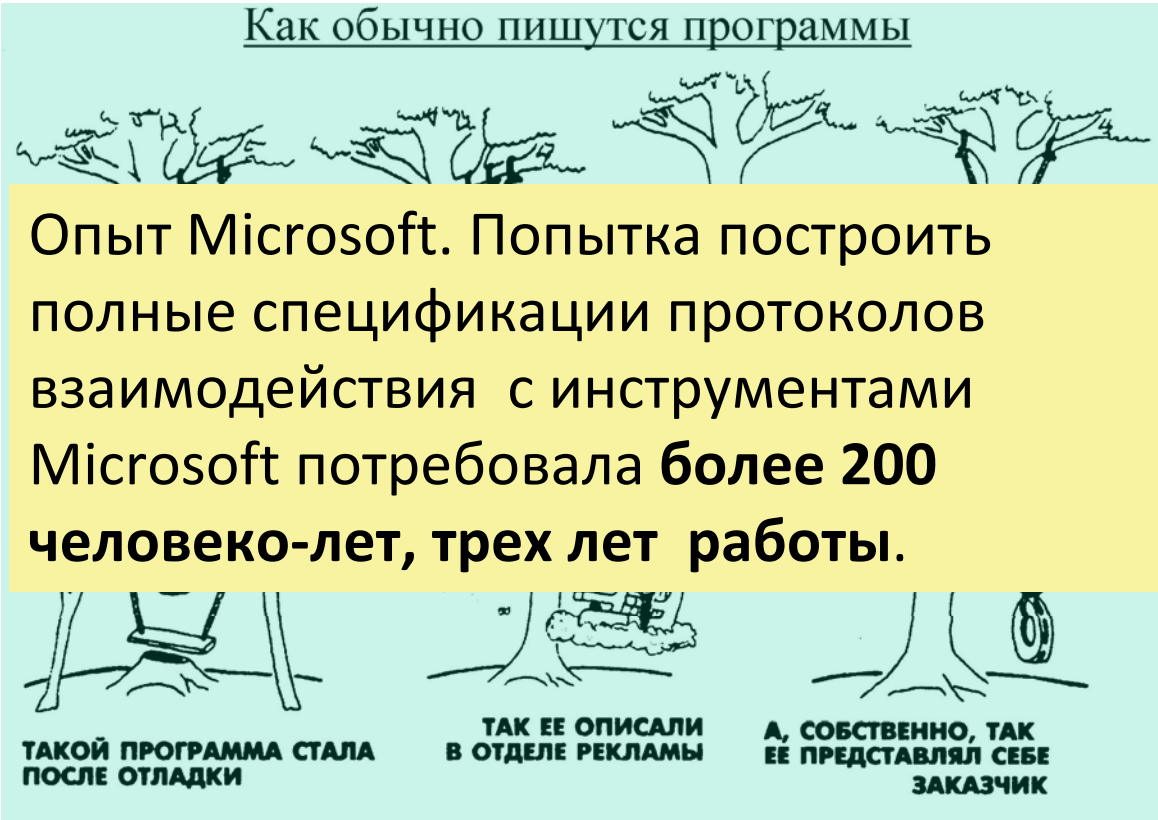
? Уровень автоматизации дедуктивного доказательства

Источники проблем верификации на практике

- Крайний дефицит качественных спецификаций и моделей
  - Поэтому верификации почти всегда предшествует сложная работа по сбору, анализу и систематизации требований
- Завышенные и одновременно заниженные ожидания пользователей от автоматизации верификации

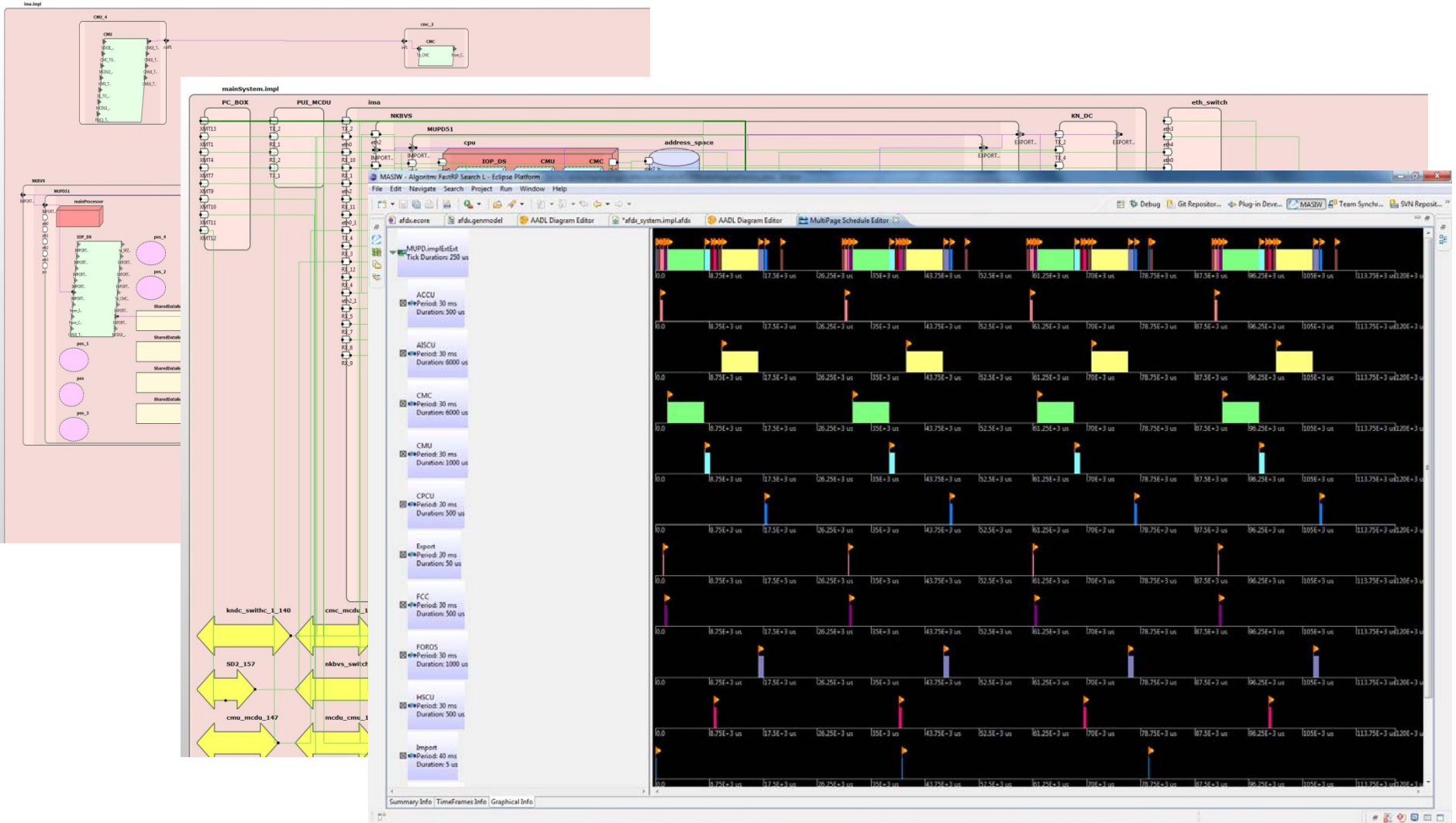
Карл Маркс. Капитал // Том 1, ИПЛ, М., 1969, стр. 189.

Но и самый плохой архитектор от наилучшей пчелы с самого начала отличается тем, что, прежде чем строить ячейку из воска, он уже построил её в своей голове.



Опыт Microsoft. Попытка построить полные спецификации протоколов взаимодействия с инструментами Microsoft потребовала **более 200 человеко-лет, трех лет работы**.

MASIW – Modular Avionic System Integrator Workplace



Значимые результаты и вехи

- Инструменты моделирования, тестирования, верификации и результаты их применения
- Открытые проекты
  - UniTESK, OLVER, LDV, BLAST, CPAchecker, MASIW
- Учебные курсы
  - ВМК МГУ, МФТИ, МГИУ (втуз ЗИЛ)
- Книги. 2 монографии, 1 учебник
  - И.Бурдонов, А.Косачев, В.Кулямин
- Диссертации
  - 2 докторские, 11 кандидатских

Что впереди? Методы и инструменты

Методы и подходы:

- Как строить и использовать спецификации и модели более экономно
- Как сводить, согласованно использовать спецификации/модели на разных уровнях программно-аппаратного стека
- Алгоритмы верификации

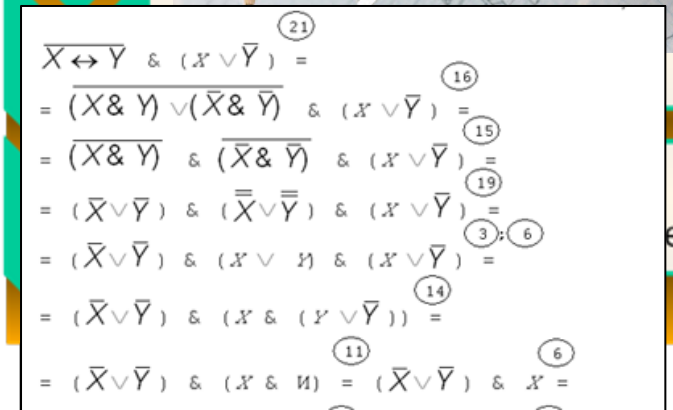
Что впереди?
Методы и инструменты

Три измерения интеграции инструментов:

- Поддержка всего жизненного цикла, от анализа требований для сертификации
- Интеграция разных техник моделирования и верификации (от ручного тестирования до аналитической верификации)
- Поддержка верификации на всем стеке компонентов компьютерных систем от приложений, до уровня ядра ОС, гипервизора и микро-архитектурного, TLM и RTL уровней описания логики микропроцессоров

Инструменты верификации:

- Генерация тестов, верификация моделей программ, дедуктивная верификация



Спасибо!