ТЕСТИРОВАНИЕ ВСТРОЕННОГО ПО: АЛЬТЕРНАТИВА КЛАССИЧЕСКОМУ TDD

Dmitry Ovechkin
Director of Product Development at Innova Systems
Dmitry_ov@yahoo.com

INNOVA

ЧТО ТАКОЕ TEST DRIVEN DEVELOPMENT?

Сначала

ТЕСТ

```
result_t TestCalc(void)
{
    if (13 == Calc(5; 8))
      return SUCCESS;
    else
      return FAIL;
}
```

Потом

КОД

```
short Calc(int A, int B)
{
   return A + B;
}
```

ПЛЮСЫ TDD

Постоянная поддержка качества кода

Является частью автоматического тестирования

Являются частью автоматических билдов

Каждый компонент покрыт тестом

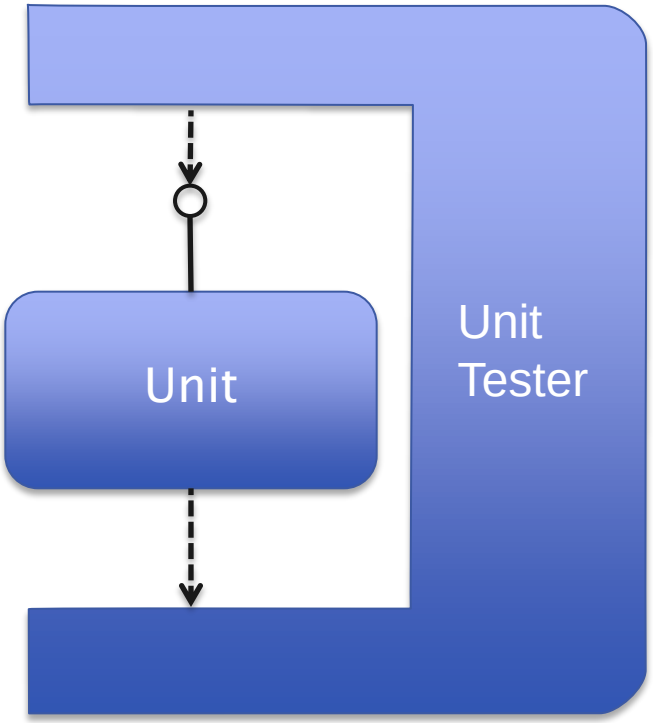


Схема Unit Test-a

МИНУСЫ TDD

- Сложно привыкнуть ментально
- Требуется больше времени на разработку и поддержку
- Рефакторинг приводит к переписыванию тестов
- Не применим к тестированию UI
- Сложно использовать для встроенного ПО (компилятор, ОС)
- Не тестирует временные зависимости
- Сомневаются в необходимости
- Жертвуют первой!

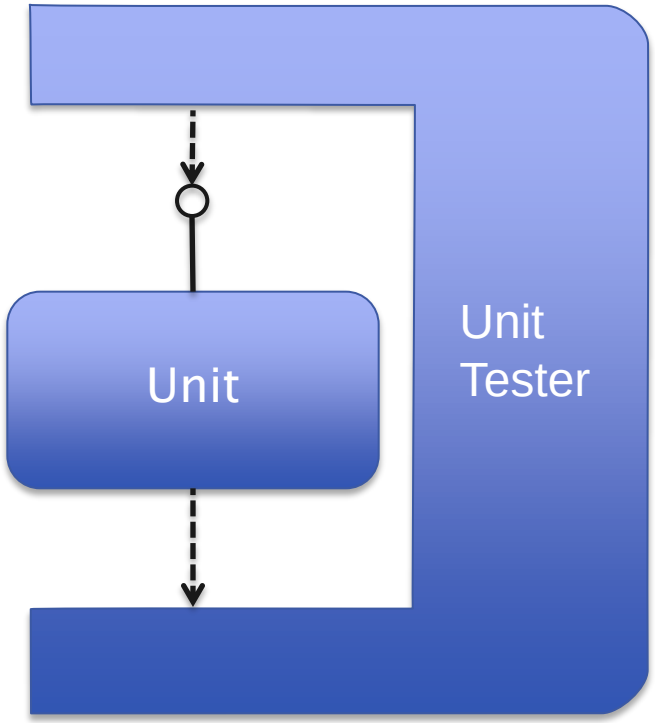


Схема Unit Test-a

ЧТО ТАКОЕ FEATURE TEST DRIVEN DEVELOPMENT?

Сначала

FEATURE

ТЕСТ

Потом

КОД

```
result_t TestCalc(void)
{
    if (13 == Calc(5; 8))
      return SUCCESS;
    else
      return FAIL;
}
```

```
short Calc(int A, int B)
{
  return A + B;
}
```

FEATURE TEST DRIVEN DEVELOPMENT

- Тестирует фичи, а не компоненты
- Требует меньше времени по сравнению с Unit Test
- Тестирует временные зависимости
- Можно использовать скриптовые языки, а значит привлечь тестеров
- Можно использовать для встроенного ПО
- Не зависит от ОС и компилятора

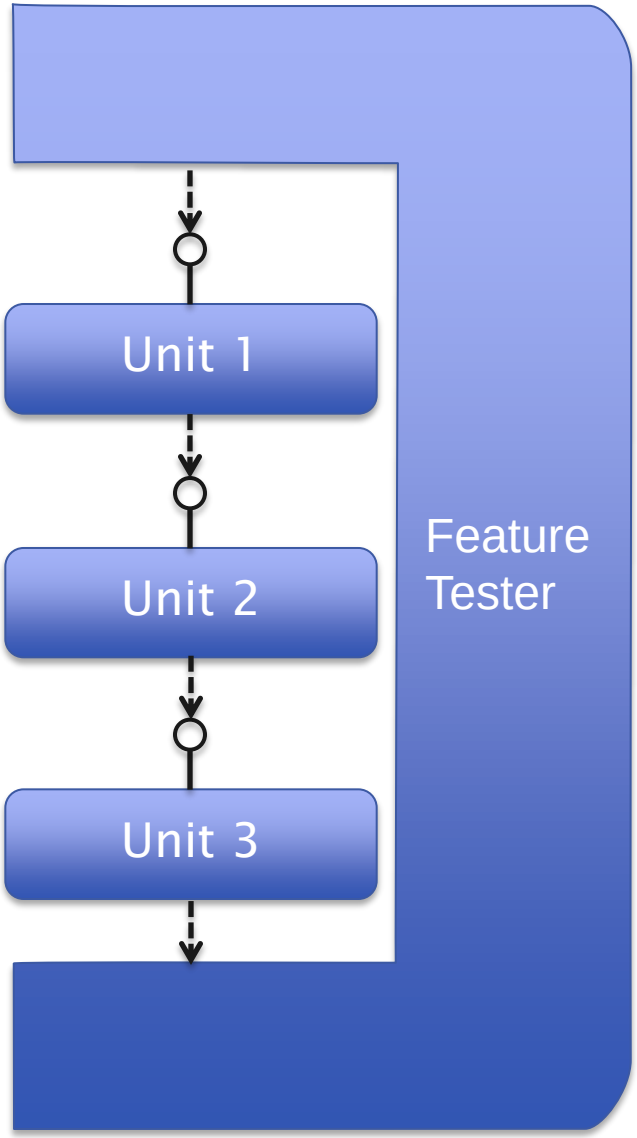
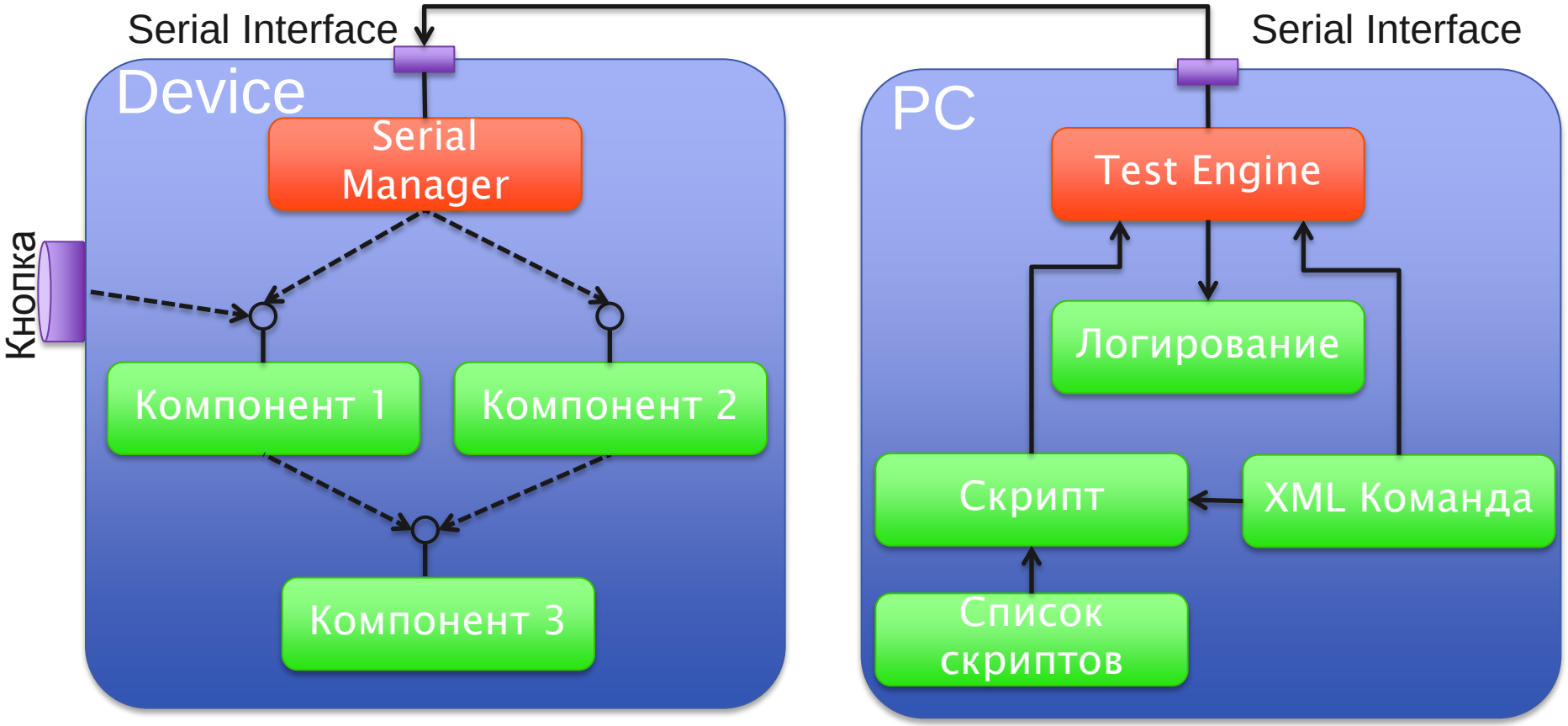


Схема Feature Test-a

СХЕМА FTDD ДЛЯ ВСТРОЕННОГО ПО



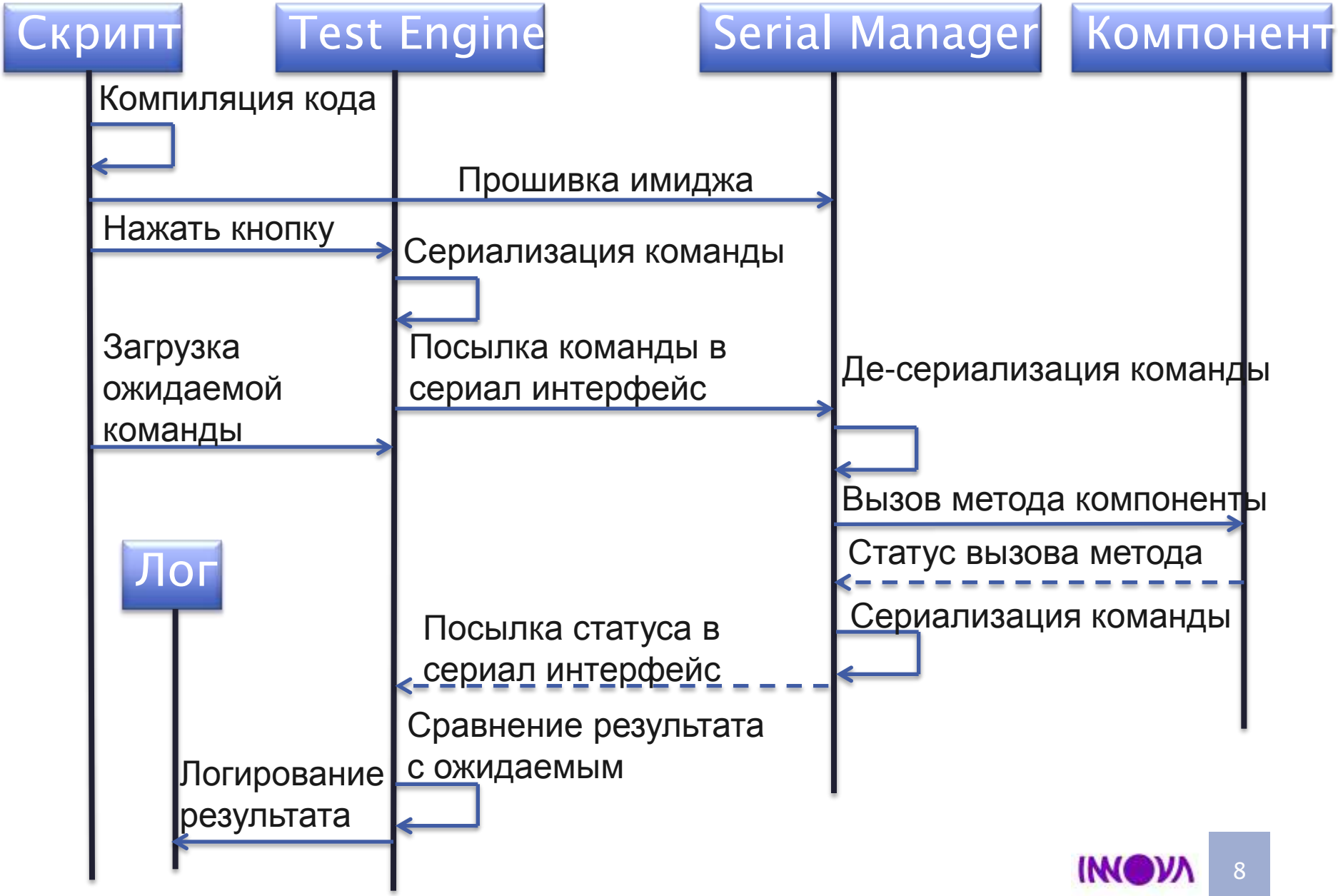
Test Engine – обрабатывает команды из скрипта, сериализует и логирует результат

Serial Manager – преобразовывает сериальные команды в вызовы методов

XML Команда – описывает формат сериальных команд

Скрипт – содержит описание тестового сценария и компилирует код

ДИАГРАММА ПОСЛЕДОВАТЕЛЬНОСТИ



ОПИСАНИЕ КОМАНДЫ – RESET_REQ.XML

```
<?xml version="1.0" ?>
<struct>
    <length type="uint8" />
    <const name="id" type="uint8">0x10</const>
  <list name="resetType" type="uint8">
    <alias value="0x00">Hard</alias>
    <alias value="0x01">Soft</alias>
  </list>
</struct>
```

ОПИСАНИЕ КОМАНДЫ – RESET_CONF.XML

```
<?xml version="1.0" ?>
<struct>
  <length type="uint8" />
  <const name="id" type="uint8">0x11</const>
  <list name="status" type="uint8">
    <alias value="0x00">Success</alias>
    <alias value="0x01">Invalid parameter</alias>
    <alias value="0x02">Unsupported attribute</alias>
    <alias value="0x03">Not Success</alias>
  </list>
</struct>
```

ПРОВЕРКА КОМАНДЫ - RESET.PY

```
"""
@parameters
  port - port (ports list) to use
  resetType - resetType
  status - expected status
"""

for port in ports:
  port.send('SysResetReq',
    resetType = resetType
  )

for port in ports:
  cmd, conf = port.receive()
  check(cmd == 'SysResetConf')
  check(conf['status'] == status)
```

ВЫЗОВ RESET КОМАНДЫ – TESTCASE1.PY

```
# Parameters list
port1='COM1'
port2='COM2'
cmdStatus='Success'
rType='Soft',

#calling reset
writeLog('1. Resetting nodes')
function('reset',
  port = [port1, port2],
  resetType=rType,
  status=cmdStatus
)
```

ЛОГИРОВАНИЕ

```
Case        : testcase1
Description: Resetting devices

1. Resetting nodes
   Send command RESET_REQ
     length 0x02
     id 0x10
     resetType 0x01 (Soft)
   Receive command RESET_CONF
     length 0x02
     id 0x11
     status 0x00 (Success)
```

Dmitry Ovechkin
Director of Product Development at Innova Systems
Dmitry_ov@yahoo.com