PHILIPS

Design for Testability

experiences from the DVD domain

Paul Thijssen, Con Bracke
Philips Applied Technologies

SASG Meeting @ ASML Veldhoven
October 3, 2006

Presentation outline

- Introduction
- Test infrastructure
- Test requirements process
- Conclusions

Introduction

Philips Applied Technologies

- aka Apptech
- Merger of former CFT and PDSL
- Part of Corporate Technologies
- Customers
  - Philips BU’s (CE, DAP, Lighting, Medical, IP&S)
  - External customers (NXP, B&O, ASML, …)
- Activities
  - Standardisation (MPEG, MHP, DVB, DVD, SACD, BD, …)
  - First of a kind development (TV, STB, DVD, Motiva, …)
  - Consultancy

About the authors

- Paul Thijssen
  - System architect @ Apptech
  - Lead software architect for the DVD projects
- Con Bracke
  - Test architect @ Apptech

Test infrastructure

Test categories

- Component test
- Integration test
- System test
- Field test
- …

Test and debug requirements

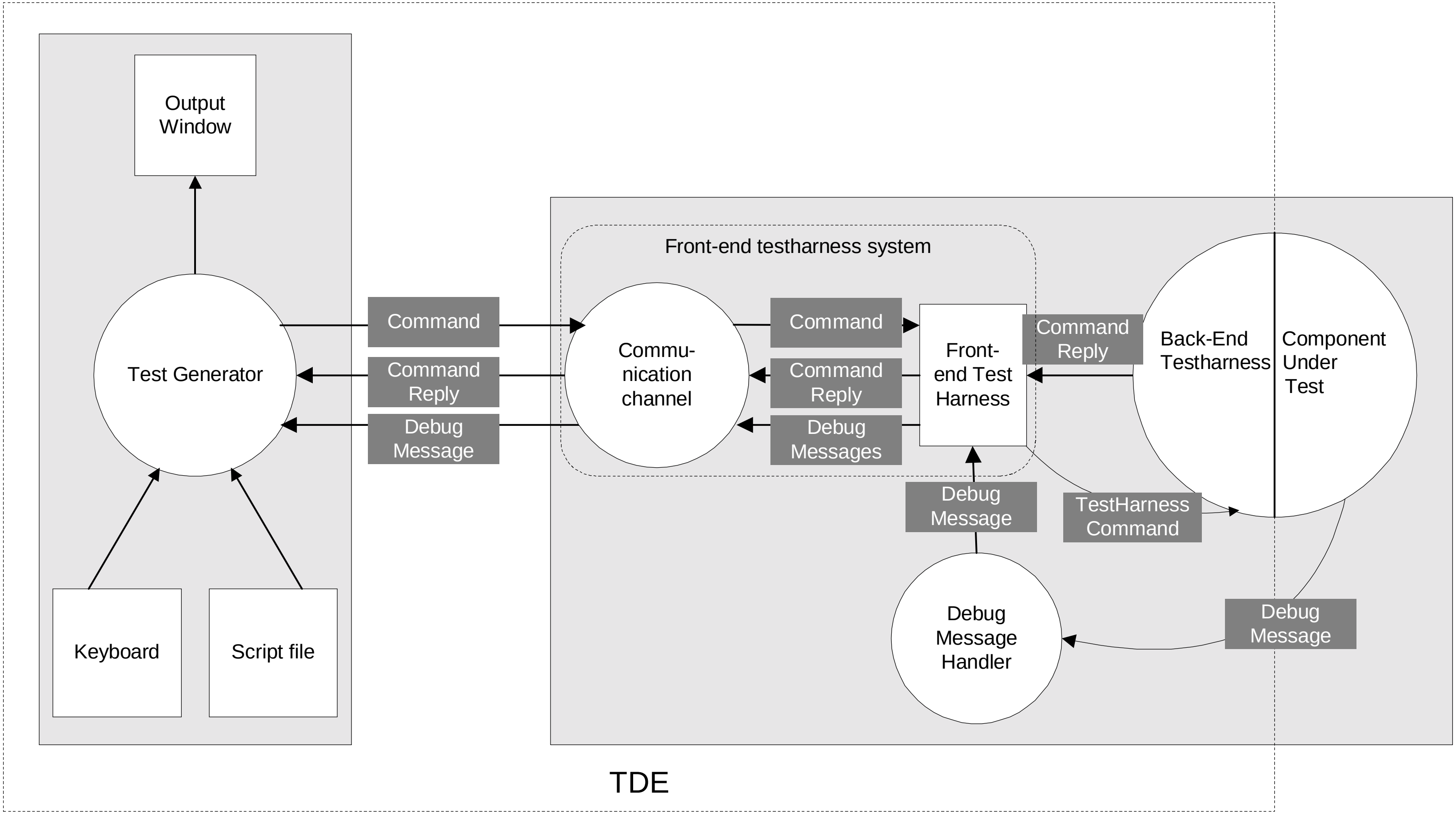
- Watch a program flow in a controlled way
- Intervene and influence the program flow
- Present things in a human readable form
- Automate the test process
  - Part of build process
  - Duration tests

Tooling

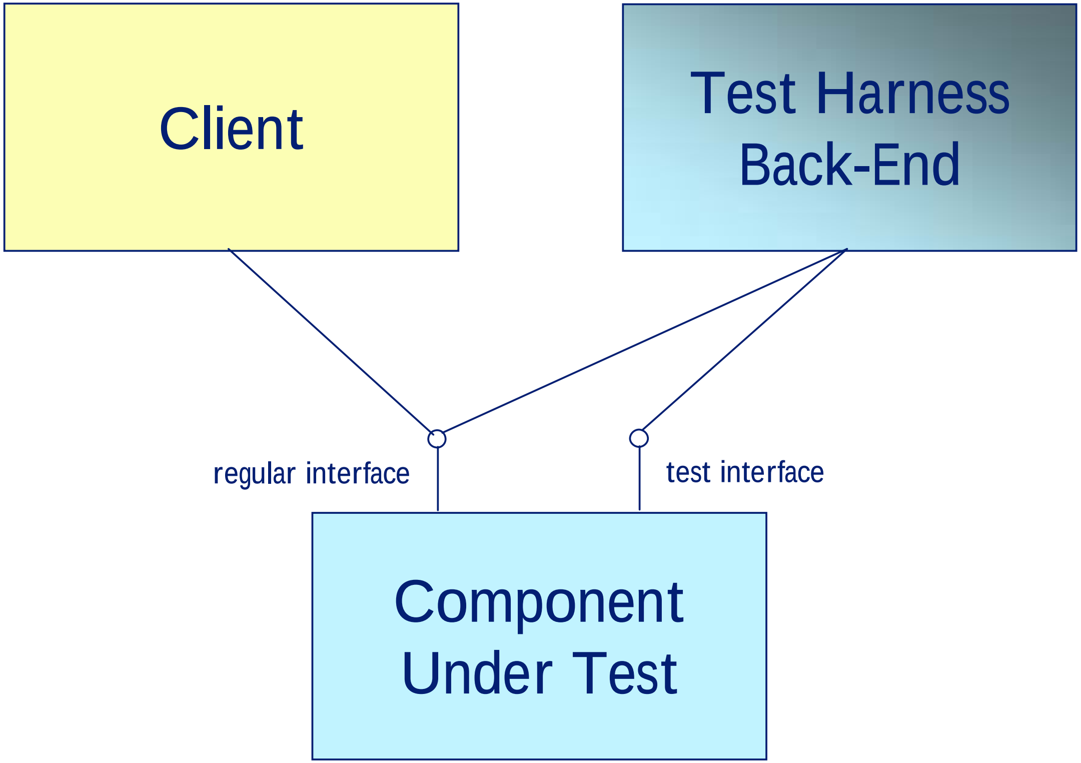
- Source level debugger
- Test (and debug) infrastructure

PHILIPS

DVD test infrastructure



Component under test



DVD debug output

- Debug levels
  - interface, state, error, warning, info, special, eventin, eventout, hifreq_eventin, hifreq_eventout, free1-6
- Debug output selection
  - select debug level
  - this can be done individually per component
- Conditional output
  - especially for duration tests and hard to reproduce or very specific problems
  - define a special trigger
  - debug output only around this trigger
- Time stamping
  - system time added to debug output
- Fatal error handling
  - debug output sent from exception handler routine

DVD stack checking

- Specific command can be sent via test generator
- Returns for each task
  - allocated stack size
  - maximum stack usage until now
- Used to
  - detect stack overflow
  - tune stack sizes / system memory requirements

System evolution

- Over time, systems tend to grow
- Think about this in advance when selecting a test framework
  - It is difficult to adapt the framework later on
  - DVD examples
    - +: debug flags and masks
      - » Select output per component
      - » Trigger mechanism for duration tests
    - -: debug strings
      - » Strings too large in SW image
      - » Better use tokens?
      - » Turned out to be too much effort to change all over the code base

Test requirements process

Requirements management

- Testability focus:
  - Limit diversity
  - Prevent ‘configurable items’
  - Sense and simplicity

Risk management

- Steer test effort
  - Less testing on less critical parts
  - More testing on critical parts

Test design

- In practice, often many normal situations are tested and (too) little boundary conditions
- Test design may impose additional requirements for development

Examples

- UI testing
  - Capture tools vs. abstraction layer
- CD track boundaries
  - Difficult to create test discs with boundary conditions
  - Even if available, exact timing and positioning is very difficult
  - Solution: ‘programmable stub’
- Critical section testing
  - How to check proper event handling during critical section?
  - E.g. stretch critical section period, install callback from critical section, etc.
- State machine testing
  - Less critical component: check single state transitions
  - Critical component: check chains of state transitions
  - What is event X in state S?
    - How to enforce state S?
    - How to enforce event X during state S?

Design reviews

- Involve a tester in design reviews
- Test view
  - Boundary conditions
  - What if … ?
- Result
  - Designer later on not surprised by what is being tested
  - Leads to better code, so less problems during test

Enablers

- Early communication
- ‘Trained’ architects
  - Awareness of importance
  - Awareness of test methods and strategies
  - It helps if an architect goes all the way from specification to release

Conclusions

Conclusions

- Use test framework as basis
  - DVD test framework is a good example
  - Take evolution into account when choosing a framework
- Make test requirements part of your process
  - Limit diversity and thus test effort (requirements management)
  - Steer test effort (risk management)
  - Design test functionality (development)
  - Early communication (architect’s awareness)

PHILIPS