PHILIPS

Test Driven Development

René Barto
SES
Agile Development - Test Driven Development
27-09-2006

Contents

- About Myself
- About SES
- Agile Development
- A Typical Developer's Day
- Test Driven Development
- Questions

About Myself

- Born 1966
- Graduated from TU Eindhoven (Information Technology)
- Around 20 years of experience in software development
- Joined Philips CFT in 1995
- Joined Philips Research in 2001
- Moved to SES in end 2002

About SES

- Founded in 2000 with three permanent staff, to support Philips Research in software prototype development
- In 2002, SES moved to the Prototyping and Instrumentation sector, to facilitate combined hardware / software prototyping
- Since 2003, many start-up companies (incubators) have been served as well, moving the focus more to product development
- Now (September 2006), SES consists of 11 permanent staff and some 70 temporary staff

Agile Development – An Introduction

Agile development is based on a lightweight process. The concepts are formulated by the Agile Alliance in *"The manifesto for Agile Software Development"*

The Agile Alliance Values:

- Value **Individuals and interactions** over process and tools,
- Value **Working software** over comprehensive documents,
- Value **Customer collaboration** over contract negotiation,
- Value **Responding to change** over following a plan.

http://www.agilealliance.org

Agile Development - What Is It?

- Lightweight (lean and mean) process for development, originated in the SW development community
- Iterative, close co-operation way of working
- Based on small, effective teams of skilled developers (4-8)
- Based on best practices which are as old as software engineering itself (evolution, not revolution)
- Very disciplined (do what you promise, fixed deadlines)
- Responding to changing requirements
- Have fun AND be productive

Agile Development - What Is It Not?

- Hacking
- Chaotic
- Silver bullet

Agile Development – Methodologies

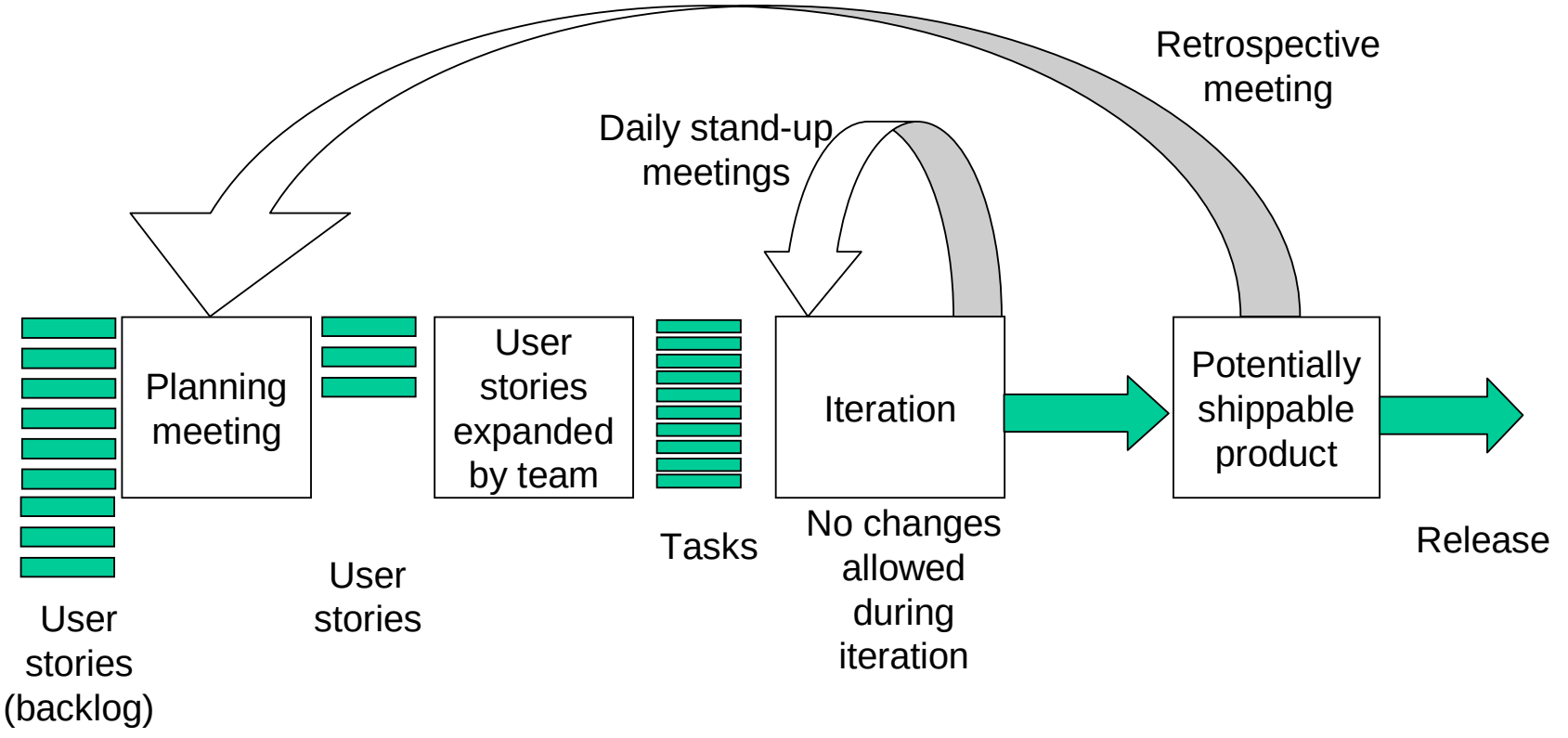
Some agile methodologies:

- eXtreme Programming (XP)
- Scrum
- Dynamic System Development Method (DSDM)
- Feature Driven Development (FDD)
- Crystal Clear
- Agile Modeling (AM)

Agile Development – Some Key Issues

- Iterative process
- User stories
- Tasks
- Stand-up meeting
- On-site customer
- Simplest solutions
- Refactoring
- Pair programming
- Continuous integration

Agile Development – Process Life Cycle



Agile Development – Iterative Process

- A development project is made up of small (1-4 weeks) time frames, called iterations
- In every iteration, a number of steps is performed
  - Requirements inventory
  - Estimation
  - Planning
  - Realization & testing
  - Review meeting (retrospective)
- Every iteration delivers working software
- A number of iterations together form a release of the product
- Iterations are fixed, their deadline never slips!

Agile Development – User Stories

- Planning with the customer is done based on user stories
- User stories are descriptions of features of the system to be built
- User stories are broken down in size, such that they can fit well into an iteration (normally 1-5 user stories / iteration)
- User stories are prioritized (priorities can change over time!)
- Prioritization determines which user stories to take up first
- The development team and customer agree on the user stories to work on depending on priority set by the customer, and effort estimations by the team
- Effort estimations are not discussed, the team has to be able to take responsibility to finish the chosen user stories

Agile Development –Tasks

- When user stories to be worked on in an iteration are defined, the team creates a work breakdown into tasks
- Tasks are pieces of work that can be finished in 1-2 days, they are estimated
- All work by team members is done based on tasks
- Team members are free to choose the tasks they will work on, however all tasks have to be finished before the end of the iteration
- A task is finished when the functionality defined for the task is created, and this can also be proven (testing!)

Agile Development – Stand-up Meeting

- Every day (preferably first thing in the morning), have a short meeting of all the developers
- Aimed at two goals: status reporting and problem recognition
- Every member will answer three questions:
  - What did you do yesterday
  - What are you planning to do today
  - What is getting in your way, how can team members help you (no one is perfect)
- Not meant for finding solutions
- Customers may attend, but not interfere

Agile Development – On-site Customer

- During development, questions are bound to come up that are related to the customer's business
- Make sure the customer is always near to answer questions
- When questions are left unanswered this can cause problems
- When answers are delayed, so is the progress
- Sometimes, the customer is part of the team (i.e. pair programming)

Agile Development – Simple Solutions

- Try to solve a problem in the simplest way possible
- Do not do up-front architecture / design of frameworks, they will evolve with the system (no one knows what the framework / architecture will be like on beforehand)
- Provide bullet-through solutions:
  - Involve all layers of the system
  - Build something that the customer can actually see / work with

Agile Development – Refactoring

- As the system / product / prototype evolves, the architecture becomes more clear
- In course of time, code duplication will arise
- Refactoring is meant to change the structure of the code, in order to:
  - Remove duplication
  - Restructure the whole
  - Build the architecture / framework as it evolves
- In case the changes don't work out, fall back to the previous version
- Always use tests to make sure everything still works!

Agile Development – Pair Programming

- Two programmers working on the same code together
- The keyboard is used by one developer (driver), the other (passenger) checks what is happening and thinks ahead
- The keyboard is switched whenever the driver gets tired, or feels less confident
- Slightly decreased development time (around 80%)
- Highly improved quality (50-70% less bugs)
- Sharing of knowledge between team members
- When pairs switch regularly, everyone knows about everything (no critical resources) (important when turnover is relatively high)
- Very focussed co-operation between two people, also very exhausting

Agile Development – Continuous integration

- For every task that is done, code is
  - Tested in isolation
  - Integrated
  - Tested in the complete system
- Automatic build / test / deployment systems help in ensuring that the complete system still works well, and can be shipped

A Typical Developer's Day

Test Driven Development – Why tests

- How do you know when you are ready without tests?
- Well-written tests provide documentation on expected and unexpected use
- Enables refactoring (when the tests succeed, you did well)
- Knowledge spread throughout the whole team (no islands / ivory towers)
- Code quality - only code that is tested is checked into versioning system!
- Helps ensure you made a shippable product

Test Driven Development – What kinds of tests

- Unit Tests - tests to ensure that a module / component / subsystem works
- Acceptance Tests - tests to ensure that the system as a whole does what the customer expects
- Management tests - tests to ensure that business goals will be made

Issues:

- You can't learn it out of a book
- Anything can be tested!
- Is as much about design as it is about testing
- Automated and isolated tests are always possible but this requires a mentality shift

Test Driven Development – Why tests first

- Testing a module in isolation means that it must be decoupled from other modules
- If you don't start with writing the tests, you probably will not have/get the time to write them after the code is there
- Creating a test for a module enforces you to think about its interface (functional / testability)
- Forces design to be done from user's point of view (the test is a user)

Test Driven Development – How test

- Make tests run automatically as much as possible
- Numerous xUnit frameworks available
  - Run automated tests with simple feedback (red / green)
  - Tests are written as code, so they are part of deliverables of the team
- Many teams use web-servers to show the status of builds / tests, or use coloured lights to warn the team in case of any problems

Some projects

ExperienceLab

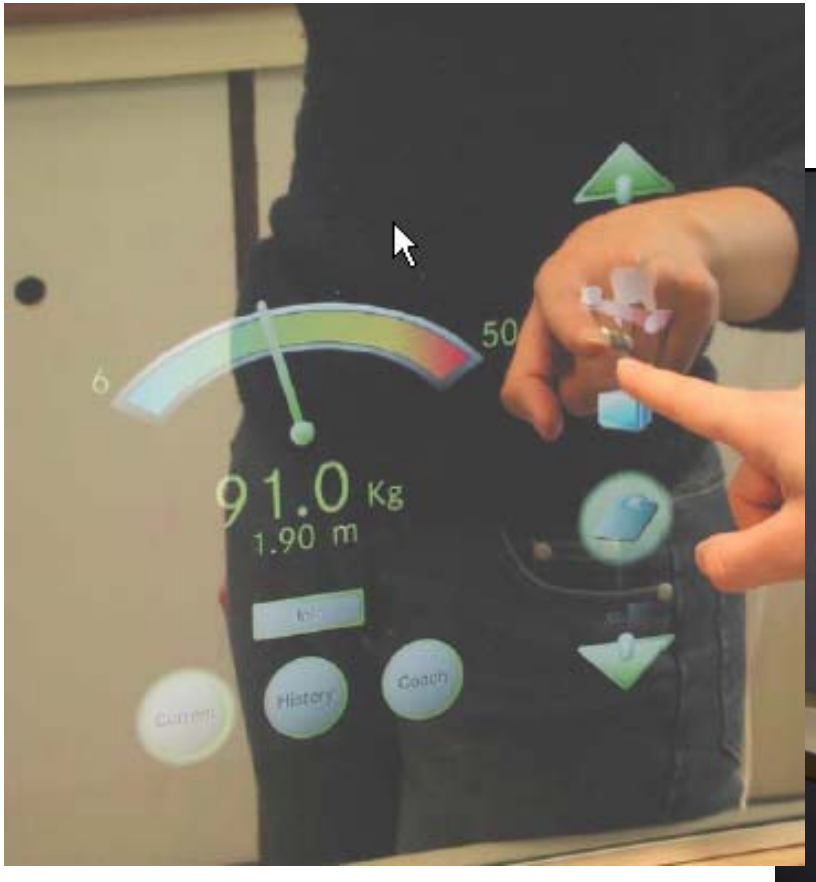
ExperienceLab

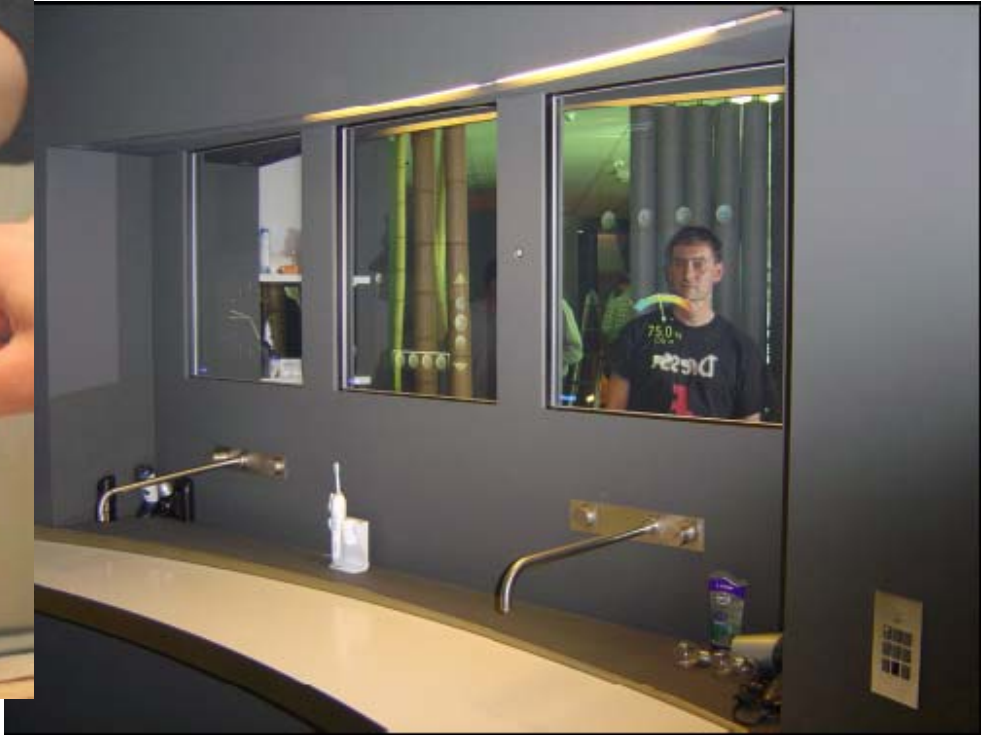
DreamScreen/DreamAudio/SmartFloor

WetLab

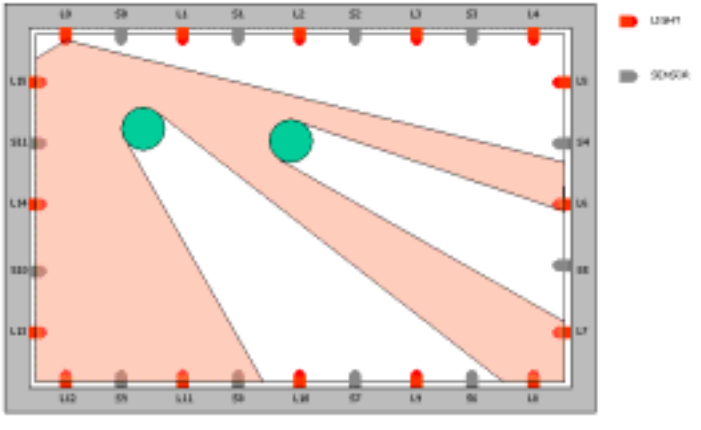
Eindhoven
leading in technology

Interactive Mirror in Bathroom LivingTomorrow (Amsterdam)





Entertaible (Entertaining Table) or Multiple Touch Screen



Polymer Vision (Rollable Display)

Content Identification

amBX

The SPC900NC PC Camera

Specifications and features:

- Top of the SPC range; “best in class”
- CCD sensor in stead of CMOS
- VGA resolution (640 x 480)
- Advanced image processing:
  - Pixel Plus
  - Digital Natural Motion
- 1.3 megapixel (1280x960) snapshots
- Frame rate: up to 90 fps
- Automatic face tracking
- Motion detection

Questions?


PHILIPS

Agile Development – Review Meeting

- At the end of every iteration a review meeting (retrospective) is held
- The review meeting is meant to
  - Discuss the results (acceptance tests)
  - Discuss any changes / updates needed
  - Discuss the process
    - What went well
    - What could be improved
    - What still puzzles us
- The customer and complete team is present

Agile Development – Metaphor

- A metaphor is a conceptual analogy (preferably taken from real world) to make it easier to communicate the concept of the system to build
- Naming of subsystems could be extracted from the metaphor
- Architecture of the system could also be in analogy to the metaphor
- The metaphor is a good way to test whether the team understands the requirements / concepts the customer is trying to communicate

Agile Development – Relevant Documentation

- Create documentation only when needed – do not create documentation for the purpose of writing it
- Documentation has a goal: communication; create / use it for that reason!
- User stories and tasks form the planning and requirements of the project, they should be well organized and archived
- In the beginning of a project, documentation could consist of pictures and some description of the concept, and possible designs
- As the team becomes more confident / experienced, the amount of documentation will decrease in size
- Code and test also form documentation

Agile Development – 40 Hour Week

- When you get tired, go home
- Most of the bugs are introduced because developers are tired / not focused
- Agile development requires a lot of discipline, and demands considerable effort
- This still means, however, that iteration dates have to be met, and the agreed functionality (user stories) has to be delivered! (time-boxing)