Modelling in an Agile World

John Daniels
Fastnloose Limited

John Daniels

- Co-founder of Fastnloose Ltd
  - Software development by dispersed teams
- Co-author of “UML Components” & “Designing Object Systems”
- Co-creator of the Syntropy method
  - Highly influential on UML (esp. OCL)
- Software designer and developer for 20 years
Agenda

- About agile
- Is UML the spawn of Satan?
- Where modelling can help

About agile

- “Agility is the ability to both create and respond to change in order to profit in a turbulent business environment”
  
  Jim Highsmith, *Agile Software Development Ecosystems*

- “Agile” is the new “Object”
  - “agile” is hip
  - even Microsoft are in on the act
The Agile Manifesto

“We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.”

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick – February 2001

Agility in practice

What does it mean to follow those values?

– Acceptance and explicit incorporation of the things we all know, deep down:
  • A skilled, motivated, enabled team can be extremely productive
  • Every project is different
  • The Devil’s in the details
  • It’s hard to keep documentation up to date
  • The customer often doesn’t know what they want
  • Detailed project plans going out more than a few weeks aren’t usually worth the paper they are written on
**Agile methods**

- **XP**
  - “No big design up front”, “You ain’t gonna need it”
- **SCRUM**
- **DSDM**
- **Crystal**
- **etc….**

**Example practices (from XP)**

- **The planning game**
  - define stories, estimate, prioritise, schedule
- **Continuous automated testing**
  - test-driven development
- **Continuous integration**
- **Refactoring**
- **Collective ownership**
- **Pair programming**
Is UML the spawn of Satan?

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

UML and its supporting tools are the cornerstone of my detailed and rigorous development process
Is UML the spawn of Satan?

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

UML and its supporting tools are the cornerstone of my detailed and rigorous development process

With UML I can spend years documenting my software

UML lets me freeze the requirements early
Is UML the spawn of Satan?

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

UML and its supporting tools are the cornerstone of my detailed and rigorous development process

With UML I can spend years documenting my software

UML lets me freeze the requirements early

Argh! I'll have to redraw all those nice UML diagrams

---

UML – it’s just a language!

- UML is a language which can be used for many different purposes
- We all use UML profiles all the time, even if we don’t realise it
- Different UML tools assume different purposes
So where can modelling help?

- Aiding customer collaboration
- Aiding interactions between developers
- Aiding the creation of working software
**Aiding customer collaboration**

- How do you get the stories straight?
  - Models of the business, not the software
    - Use of models to understand what the business does / should do
    - Nothing to do (directly) with the software systems that support the business
  - Models that specify the software scope
  - Models that can expose conflicts or omissions

- How do you manage hundreds of separate stories? (esp. if they are written on index cards!)

- Need tools that can support abstract models

---

**Direct customer interaction**

- Small-ish projects
- Developer acts as analyst-designer
- Any models are private to developer
- “Pure XP”
Explicit requirements model

- Large-ish projects
- Analyst and designer roles separated
- Requirements model is explicit communication tool

The Analysts’ Loop provides value only if it cleans up the requirements by raising questions about inconsistencies and gaps.
Finding more use-cases: CRUD

- Inspect class diagrams; ask:
  - what changes this association or attribute?
  - what creates/deletes this object or association?

Finding more usecases: states

<table>
<thead>
<tr>
<th>Room</th>
</tr>
</thead>
</table>

- Occupied
  - check out
- Dirty
  - build
  - reinstates
- Maintenance
  - withdraw
  - clean
- Unoccupied
  - withdraw
  - check in

Customer
  - pocket: Money
  - possessions 0..*
Supplier
  - stock 0..*
Thing
  - close
  - open
  - refund
  - sale
  - theft
  - restock

© 2003 fastnloose ltd
Use-cases and objects – consistency

what else do they do/happens to them?
-- CRUD; statecharts

who/what is affected?
-- write postconditions, look at terms used

what objects are there?
-- things, concepts, relationships

what happens?
-- tasks, jobs, use-cases

Aiding developer interactions

Use of models to allow developers to visualise, share and discuss code design ideas, solutions

Use of models to allow newcomers to see the big picture quickly

Need tools that can move quickly and easily between code and diagrams
Aiding working software

- Generation of test cases from requirements (i.e., specification) models
- Understanding the good test cases
- Need tools that can generate tests

From requirements to tests

Running a test

1. Set-up test scenario
2. Execute test
3. Check invariants and post-conditions

- The structural constraints in a requirements model define the interesting edge and normal conditions, and hence inform the choice of test data
- A requirements model provides a catalogue of test points when testing is at the external event level
- A requirements model will define the structural invariants and the required results of stimuli
Tests as specification

- Traditional textual functional specification
  - broad, covering all aspects of the problem
  - lacking depth, precision
  - big picture is often all you get

- Tests-as-specification
  - precise where you have them
  - but nothing where you don’t
  - hard to see the big picture

- Models can bridge this gap

Agility and the MDA

- Model Driven Architecture
  - “A complete MDA specification consists of a definitive platform-independent base UML model (PIM), plus one or more platform-specific models (PSM) and interface definition sets, each describing how the base model is implemented on a different middleware platform.” OMG web site!

- Assuming MDA can be made to work, it is compatible with a high level of agility

- The modelling language becomes the “programming” language, but the same principles apply
Summary

► Modelling is compatible with agile methods

► Models can help:
  – manage complex requirements
  – improve communication between developers
  – with the creation of good tests