

The Use Case Problem: Time Testers Spoke Out

Software Futures Ltd

2 Waterloo Way

Bredon

Tewkesbury

Glos.

GL20 7NA

GB

p +44 (0) 1684 772 691

f +44 (0) 1684 772 639

e richard@softwarefutures.ltd.uk

w www.softwarefutures.ltd.uk

Topics

- What is the purpose of Use Cases?
- What is the Use Case problem?
- A UML Tester's Charter.

The OMG Definition

‘.... a kind of classifier representing a coherent unit of functionality provided by a system, a subsystem, or a class as manifested by sequences of messages exchanged among the system (subsystem, class) and one or more outside interactors (called actors) together with actions performed by the system (subsystem, class).’

OMG Unified Modelling Language Specification, V1.4, September 2001.

The Purpose of Use Cases

‘Use Cases are used to describe the outwardly visible requirements of a system. They are used in the requirements analysis phase of a project and contribute to test plans and user guides. They are used to create and validate a proposed design and to ensure it meets all its requirements. Use cases are also used when creating a project schedule, helping to plan what goes into each release.’

Schneider G & Winters JP, Applying Use Cases: A Practical Guide.

‘A use case describes a sequence of actions that provide a measurable value to an actor.....Use case models should be developed from the point of view of your project stakeholders and not from the (often technical) point of view of the developers.’

Ambler, Scott W, The Elements of UML Style

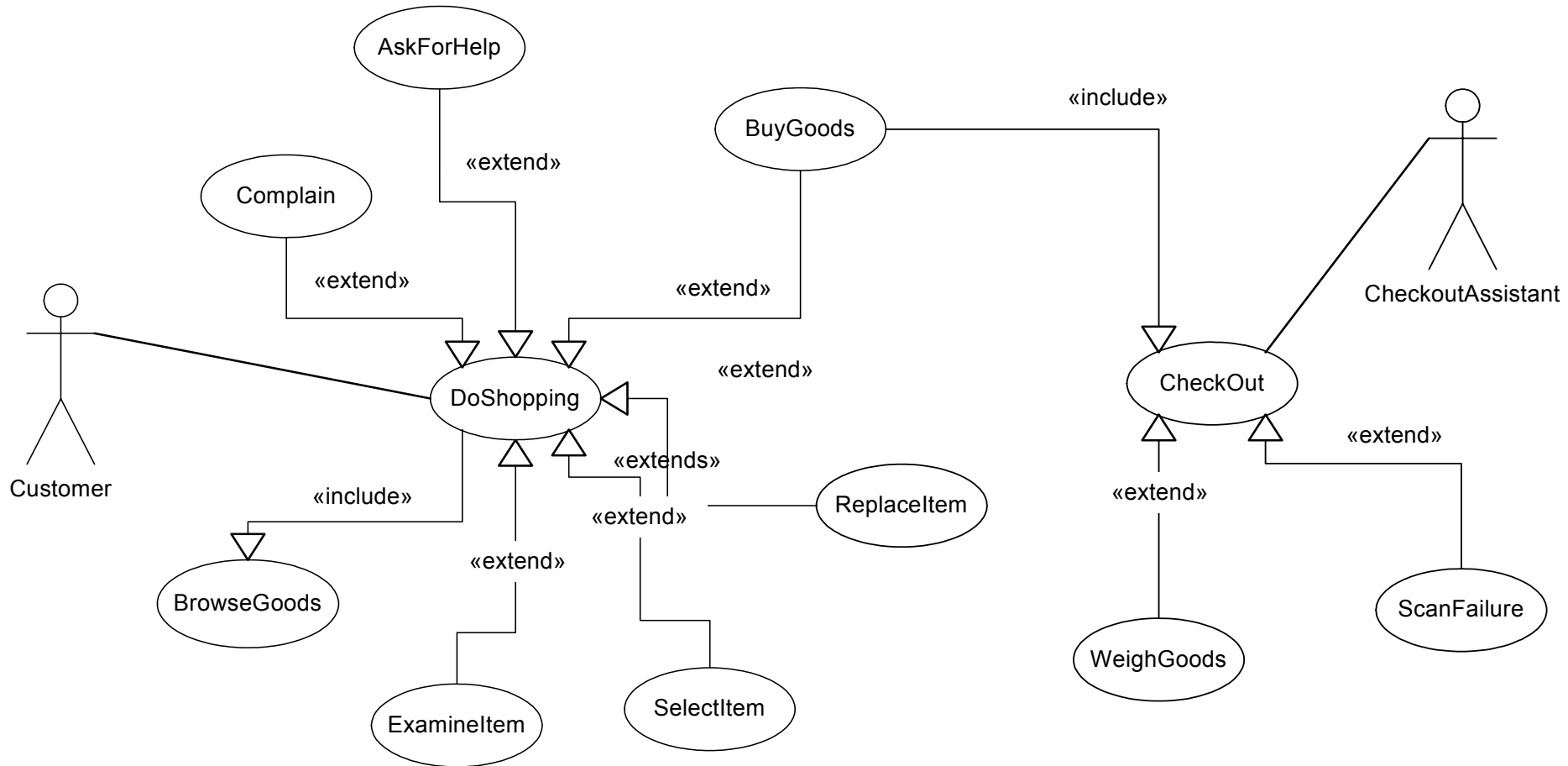
What does this mean?

- A user's view of functions.
- A Use Case represents a meaningful piece of user behaviour.
- They are the starting point for design.
- They are used to validate designs.
- They contribute to test planning & design.

The Use Case diagram

- The glue that holds the requirements together (Ambler).
- Shows relationships that must be tested.
- Identifies test targets (Gross).

Part of the Supermarket



A catalogue of problems

- Fail to capture highest levels of business functions.

‘The models for key testing problems are "higher level" than most UML models (e.g. focus is not on classes and internal structure, but on behaviour and capabilities from a black box point of view. Implication: rethink the capability models (e.g. use cases)’.

Williams, C, UML and Testing: A Perfect Fit? 2002

Variations on a theme

- The Analysis Gap – going from abstract Use Cases to design.
- Pre-empting design - design detail in Use Cases.
- A Random Collection – Use Cases but no diagram to glue them together.
- Use Case Free Zone – just a Use Case diagram (c.f. LS Lowry)

A shopping list of Use Cases

- AskForHelp
- BrowseGoods
- BuyGoods
- CheckOut
- Complain
- DoShopping
- ExamineItem
- ReplaceItem
- ScanFailure
- SelectItem
- WeighGoods

The Minimalist Approach

- Building the world's smallest requirements model.
- Use Case only analysis.
- Terrible narratives, problem templates, no Activity diagrams.
- Forget about state behaviour.
- What is a Sequence diagram?
- Use Cases are temporary, aren't they?

Use Case model design

- or lack of it.
- Poor design = lack of testability.
- Arbitrary limits – 20 Use Cases.
- Many design rules to help or break.
- Difficult relationships
 - Control flow issues.
 - State issues.
- Early reviews essential.

What about data?

- Where do we get the data from?
- Not in Use Cases.
- From database design.
- Business object modelling.
- *'UML for Database Design'*
 - *Business Use Cases and Actors.*
 - *Design Use Cases and Actors.*

Anything non functional?

- For many projects more than 50%.
- Specialist modelling.
- How to extend UML to include non-functional behaviour.
 - Timing/response – in Sequence diagrams.
 - Usability - described in Use Cases.
 - Navigability - GUI design using Activity diagrams.

Effects of these problems

- Applications get more complex, testers unable to rise to the testing challenge.
- Low test productivity and coverage.
- Lack of early involvement by testers (reviews) increases error migration.
- How do developers improve?

Tester's Charter 1

- Bring together as much best practice as we can.
- Basis for negotiating with developers on UML usage.
- Basis for test strategy and planning.
- Define model usage by testers.
- Identify standards testers require.
- Map usage to test processes.

Tester's Charter 2

- For each UML artefact, starting with Use Cases:
 - Define how they are used.
 - Define when they are used.
 - Describe testability criteria.
 - Describe test techniques.
 - Identify relationships between artefacts.

Conclusions

- Model-based testing requires greater skills.
- Lack of collective wisdom.
- Testers working with UML have an opportunity to develop skills and wisdom.
- www.umltesters.org

