Relevance of Use Case Methodology for Zero Outage Industry Standard

Zero Outage

INDUSTRY STANDARD

CONTENT

Written by
Georg Bock, Sr. Director, Micro Focus
Lead of the ZOIS Architecture Board
Member of The Open Group IT4IT™ Forum

3 Motivation

4 Terminology clarification

8 Zero Outage Focus

9 Expanding the methodology

11 Context Diagrams

12 How to determine the data?

13 Appendix: Solution use case questions

17 Footnotes

Motivation

As leader of the architecture board, I'm always looking for triggers to challenge the status quo. Discussions involving real-life examples and scenarios of organisations trying to achieve zero outage often raises many valuable and insightful points. Imagination can only go so far – there is always the need for detail if a solid implementation is to be applied and from this, dependable design nuggets can then be extracted.

The difficulty is to identify the *use cases* and get to this required level of detail. We tend to use this term fairly regularly in our IT language, however, there does not seem to be a universally agreed definition, nor a standard methodology and format to develop and articulate use cases. However, if there was an agreed practice and standard, it might motivate business analysts, architects and product managers to document and share use cases. When a use case library becomes part of a shared IT knowledge system, it can easily be used to improve the quality, applicability and prescription of the architectural guidance.

Terminology clarification

Let's look at some definitions one can find quickly on the internet:

Wikipedia[1]:

- In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role (or actor) and a system to achieve a goal. The actor can be a human or other external system.
- Use case analysis is an important and valuable requirement analysis technique that has been widely used in modern software engineering since its formal introduction by Ivar Jacobson in 1992.

The Open Group TOGAF® standard[2]:

- Use-Case: A view of organisation, application, or product functionality that illustrates capabilities in context with the user of that capability.
- Use-case Analysis: the breakdown of business-level functions across actors and organisations allows the actors in a function to be identified and permits a breakdown into services supporting/delivering that functional capability

" LIST OF ACTIONS OR EVENT STEPS

typically defining the interactions between a role (or actor) and a system to achieve a goal. "

1. See https://en.wikipedia.org/wiki/Use_case
2. TOGAF® is a registered trademark of The Open Group

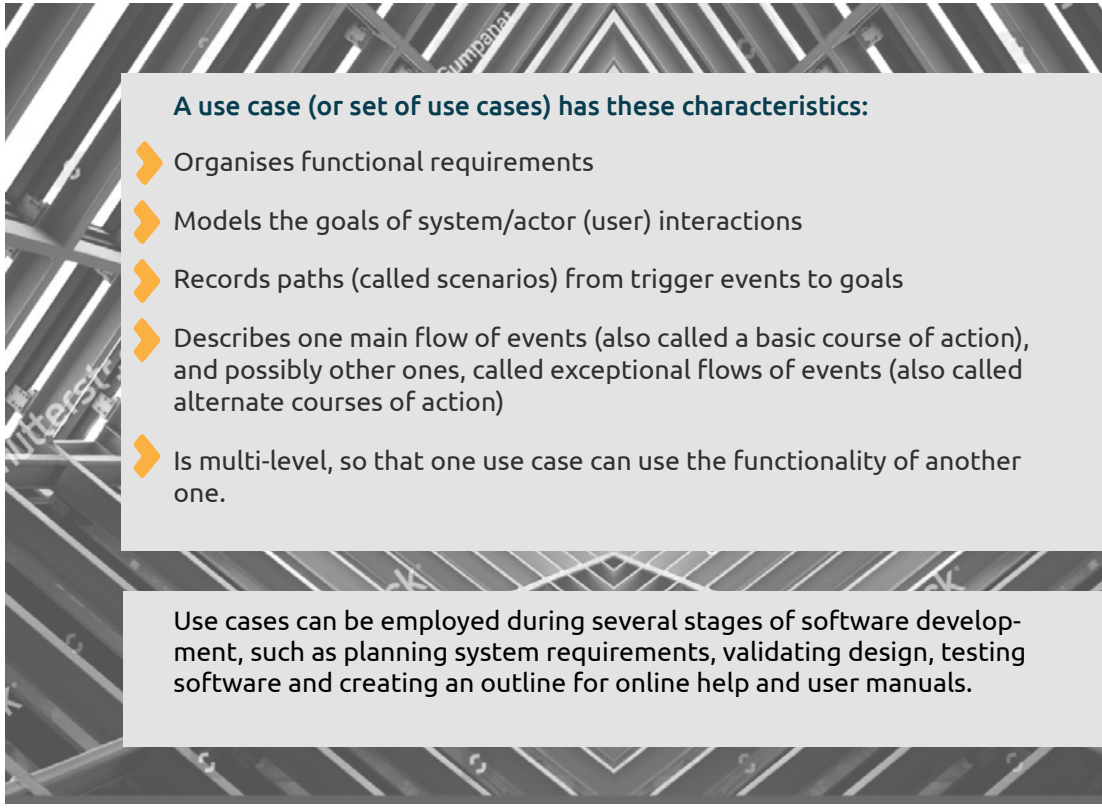
Techopedia[3]:

- A use case is a software and system engineering term that describes how a user uses a system to accomplish a particular goal. A use case acts as a software modelling technique that defines the features to be implemented and the resolution of any errors that may be encountered.
- Use cases define interactions between external actors and the system to attain particular goals. There are three basic elements that make up a use case:
  - Actors: Actors are the type of users that interact with the system.
  - System: Use cases capture functional requirements that specify the intended behaviour of the system.
  - Goals: Use cases are typically initiated by a user to fulfil goals describing the activities and variants involved in attaining the goal.

TechTarget[4]:

- A use case is a methodology used in system analysis to identify, clarify, and organise system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. It consists of a group of elements (for example, classes and interfaces) that can be used together in a way that will have an effect larger than the sum of the separate elements combined. The use case should contain all system activities that have significance to the users. A use case can be thought of as a collection of possible scenarios related to a particular goal, indeed, the use case and goal are sometimes considered to be synonymous.

3. See https://www.techopedia.com/definition/25813/use-case
4. See https://searchsoftwarequality.techtarget.com/definition/use-case



A use case (or set of use cases) has these characteristics:

- Organises functional requirements
- Models the goals of system/actor (user) interactions
- Records paths (called scenarios) from trigger events to goals
- Describes one main flow of events (also called a basic course of action), and possibly other ones, called exceptional flows of events (also called alternate courses of action)
- Is multi-level, so that one use case can use the functionality of another one.

Use cases can be employed during several stages of software development, such as planning system requirements, validating design, testing software and creating an outline for online help and user manuals.

Alistair Cockburn[5]:

A use case captures a contract between the stakeholders of a system about its behaviour. The use case describes the system's behaviour under various conditions as it responds to a request from one of the stakeholders, called the primary actor. The primary actor initiates an interaction with the system to accomplish some goal. The system responds, protecting the interests of all the stakeholders. Different sequences of behaviour, or scenarios, can unfold, depending on the particular requests made and conditions surrounding the requests. The use case collects together those different scenarios.

5. Writing Effective Use Cases 1st Edition by Alistair Cockburn, ISBN-13: 978-0201702255, ISBN-10: 0201702258

"... a method to capture user requirements in order to fulfil business requirements/goals."

Karl Wiegers[6]:

- A use case describes the sequence of interactions between a system and an external actor that results in the actor being able to achieve some outcome of value.
- Use case is a method to capture user requirements in order to fulfil business requirements/goals. It focuses on actual, but abstracted, usage scenarios from which one can determine functional requirements to guide the development of systems meeting the use user requirements (note: this is a summary, not a literal quote).

Harmonising and simplifying the definition and meaning:

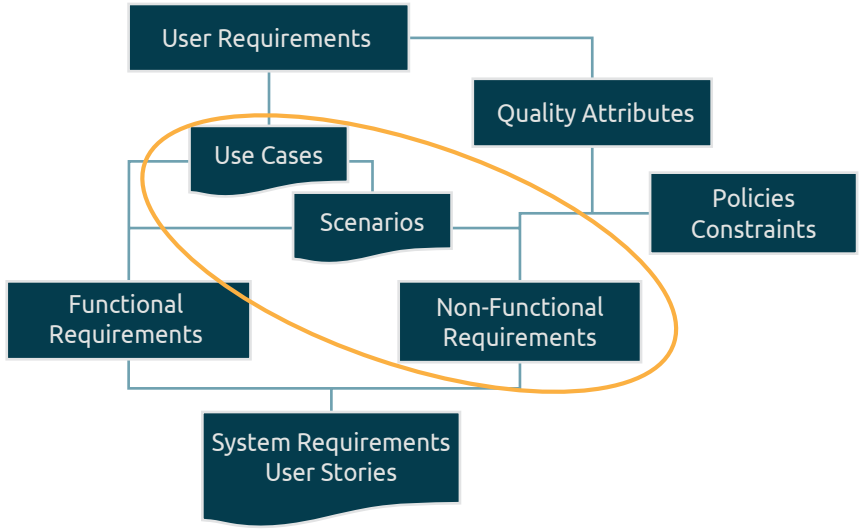
- A use case captures:
  - *Why: the user's goals*
  - *What: the interactions between a user and a system (process, actions, events, …)*
- A use case does NOT describe:
  - *How: the systems functional requirements → captured by user stories in agile dev*
  - *How specifically: different successful or failure paths → captured by scenarios*
  - *How technically: the system non-functional requirements → quality attributes, design guidelines, policies etc.*
- Use case analysis is a methodology to gain a shared vision and understanding of requirements across stakeholders, primarily business analysts, product managers and developers.

6. Karl E. Wieger, Software Requirements, Second Edition (Pro-Best Practices), ISBN-10: 0735618798, ISBN-13: 978-0735618794 Karl Wiegers Describes 10 Requirements Traps to Avoid, Copyright © 2000 by Karl E. Wiegers, see also http://processimpact.com/articles/reqtraps.pdf

Zero Outage Focus

Zero Outage is about preventing any disruption to the business usage of a service. That has a lot to do with designing and implementing a "perfect" service system underneath. One of the problems we detected is the lack of maturity in developing meaningful, actionable, non-functional requirements that prescriptively translate quality attributes into guidance for developers. The Zero Outage architecture publication talks about the observation that (especially in software development) a lot of focus goes into the functional requirements specification, while the non-functional perspective is often just a side note, mainly due to the lack of skills and maturity across the stakeholders.

When looking at the graphic it seems obvious that use cases can play an important role for the collaborative development of non-functional requirements. Specifically, the scenario aspect of the use case and the methodology to analyse use cases across stakeholders of providers and consumers may give us a way to overcome the issue.



Expanding the methodology

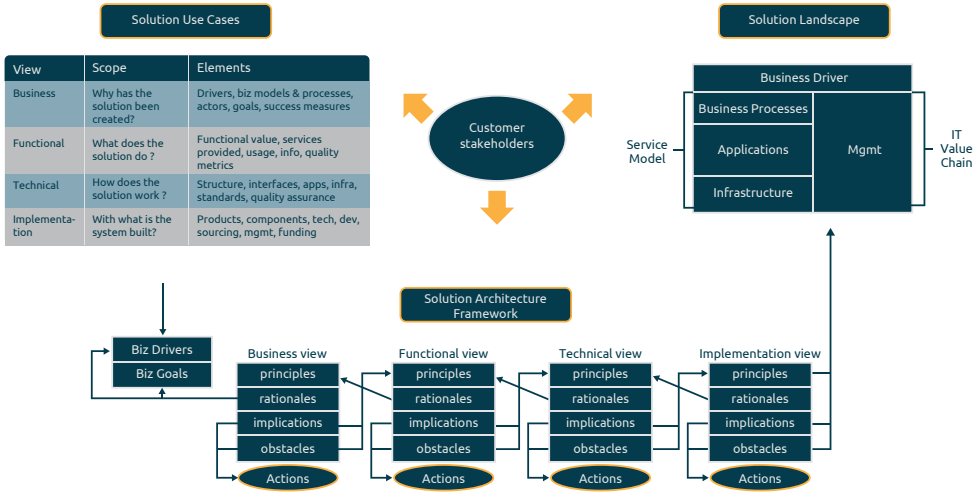
Going back to the definition of use case, the focus is really on the translation of the use case, describing the 'why' and 'what', to the specification of the system, meaning the 'how' and 'with what'.

At Hewlett-Packard, where I worked for a long time, we developed and used a methodology called IT Strategy & Architecture (ITSA)[7] which defines these levels in model detail:

- Why – Business View
- What – Functional View
- How – Technical View
- With – Implementation View

At the end of the day, the providers need to build a solution that enables the use cases, hence we need to expand the use case methodology to one level below and talk about *solution use cases.* This takes the user to interactions within the system, e.g. between multiple composite elements or between the software and infrastructure components it depends on.

Therefore we could apply ITSA to the case analysis in order to integrate the user, the architecture and the landscape perspective into a joint vision. Please see the diagram below. Note that the solution landscape includes both the value chain and the service model, which are key ingredients of the Zero Outage architecture framework.



Without going into detail, the solution use cases capture the purpose and the functional expectations of the system – but then also dive into the technical requirements of how the system should work and how it should be brought to life.

That requirements perspective can then be translated into an architectural one, which examines the architecture at the level of the use case and determines the required principles working top-down as well as showing the rationale of doing so, from the bottom-up.

The solution landscape then brings it to live in a model of interacting layers that is managed and governed as an IT value chain.

The key here, is the transitions and logic between the different elements of the solution and the methodology to develop and formally articulate this within a collaborative group of stakeholders, consumers and providers.

7. See http://www.opengroup.org/architecture/0404brus/presents/kruik/EABrussel22aprilFinal.pdf

Context Diagrams

One may argue: why try to adapt the use case method to describe the context of systems when there is a well-known enterprise architect construct, called Context Diagrams? The latter is primarily used for project context, e.g. in TOGAF®[2], but can also be applied to systems.

In engineering, a system context diagram defines the boundary between the system or part of a system and its environment while showing the entities that interact with it. This diagram is a high-level view of a system and is similar to a block diagram.[8]

While it succeeds in describing the external interfaces, it fails to explain the impact on the composite elements of the system and the reasons thereof, which are important if meaningful requirements are to be understood. Identifying those dependencies is where the different related ITSA layers come into effect.

How to determine the data?

As mentioned in the intro, ideally such a methodology would be standardised along with the artefacts that result from it. We may discuss this in the Zero Outage association and/or discuss it with other applicable constituencies, such as The Open Group. However, that is a future opportunity and doesn't help us here.

Regarding both the use case format and use case analysis workshop – you can find various, fairly consistent offerings on the internet, therefore I won't delve into these for this paper. I personally found Karl E. Wiegers work helpful (see footnote below) and those that are based on it.

But I'd like to provide some thoughts to fill the picture above with life in a manual fashion, leveraging the stakeholder workshop methodology of the use case analysis. Therefore, I compiled a couple of questions in the appendix that one can challenge different audiences with.

Now we need to look for Zero Outage solution examples within and outside our association to learn from the similarities, differences and most importantly from the complex dependency patterns between solution components and ITSA levels.

8. See https://en.wikipedia.org/wiki/System_context_diagram and also https://de.slideshare.net/alanmcsweeney/approach-to-it-strategy-and-architecture

Appendix: Solution use case questions

For every solution framed by a set of use cases and related scenarios:

Why has the solution been created?

- What is the set of problems and demands driving the decision to invest in this solution?
- What business process steps does the solution ultimately support?
- In what way, if so, does the solution change the way you fundamentally do things?
- What does success look like?
- What outcomes do you measurably intend to achieve/improve?
- What are the conceptual building blocks of the solution, including the re-use of existing blocks/services?
- Who is responsible for delivering the components (internal and external) and why (make vs. buy)?
- How do the building blocks relate to defined company-wide technology life-cycles? (e.g. OS or database strategies)?
- Were there specific hosting requirements/dependencies (e.g. private data storage)?
- What is the level of business criticality of the solution?

What does the solution do?

- What are the use case flows (defined services) the solution enables (e.g. upgrading a new version of the SAP instance, on-boarding a new SAP user)?
- What are the managed objects of the IT services (e.g. SAP instance in form of a service model)
- How are the IT services interconnected that belong to the same use case (service architecture relating to the building blocks above)?
- What are the IT management capabilities required to deliver the IT services?
- Who are the IT actors involved with the capabilities (job profile, role in organisation, responsibility regarding capability)
- Who are the (Business or IT) end-users of the solution on the IT and/or the business side?
- What are the parameters driving the experience perception of these end-users?
- How does the usage model scale up, clustered by regional instantiation?
- What are the resilience, security and performance considerations relating to the defined business criticality?

How does the solution work?

- What are the resulting integrated IT functions that constitute a use case?
  - The result can be mapped onto the IT4IT reference architecture level 1 which showsquired integration points
- What are the goals of automating the use case flow?
  - Where is tooling required
  - Where are the manual touch-points required, which results in collaboration requirements between people
- What is the resulting solution architecture?
  - What is the picture of interconnected automated functions (tools)?
  - What are the required technical integrations between automated functions (data, logic)?
  - What does the layered service model look like, the stack of service logic linking to business process, underlying application(s), middleware, controller and infrastructure (e.g. ARIS, SAP and deployment requirements)?
  - Where is the service model maintained and governed along the life-cycle (e.g. release model, CMDB)?
- Per automated function (tool)
  - What are the required APIs and architectural integration patterns (e.g. API gateway)?
  - What are the resulting non-functional requirements (security, availability, performance, usability…)?
  - What are applicable standards (e.g. IT4IT)?
  - What are the deployment model requirements (e.g. on premise vs. cloud)?
  - What are the data persistence and retention requirements?

What is the solution built with?

Tools

- What are the tools mapping to the IT functions (incl. commercial, in-house, sourced)?
- What customisations are required/have been implemented?
- How change resilient are these (regarding new versions)?
- Which integrations need/had to be built?
- Which of these are compliant with IT4IT? Which are not and why?
- Which IT4IT prescribed integrations are missing and why?
- Which additional non-prescribed integrations will be/were built and why?

Infrastructure

- What is the deployment model to host the tools?
- What is the underlying server, OS, database instantiation?
- What is the high availability implementation?

Integrations

- Which out-of-the-box integrations are used?

Managed service objects

- Which applications are managed by services (e.g. SAP modules and versions and the underlying technology stack)?
- In which tool(s) is the respective service model implemented and maintained?
- How do other tools access the required information?

Footnotes

1. See https://en.wikipedia.org/wiki/Use_case

2. TOGAF® is a registered trademark of The Open Group

3. See https://www.techopedia.com/definition/25813/use-case

4. See https://searchsoftwarequality.techtarget.com/definition/use-case

5. Writing Effective Use Cases 1st Edition by Alistair Cockburn, ISBN-13: 978-0201702255, ISBN-10: 0201702258,

6. Karl E. Wieger, Software Requirements, Second Edition (Pro-Best Practices), ISBN-10: 0735618798, ISBN-13: 978-0735618794
Karl Wiegers Describes 10 Requirements Traps to Avoid, Copyright © 2000 by Karl E. Wiegers, see also http://processimpact.com/articles/reqtraps.pdf

7. See http://www.opengroup.org/architecture/0404brus/presents/kruik/EABrussel22aprilFinal.pdf

8. See https://en.wikipedia.org/wiki/System_context_diagram and also https://de.slideshare.net/alanmcsweeney/approach-to-it-strat-egy-and-architecture

The information contained on this website / in this document represents the view of the author, not necessarily the view of the Zero Outage Industry Standard Association. However, the thought leadership presented here is intended to catalyse evolving the standard.

CONTACT

Zero Outage Industry Standard Ltd.
Suite 1, 3rd Floor
11-12 St. James's Square, London
SW1Y 4LB
United Kingdom

info@zero-outage.com