

Use Cases Notes from a Service Oriented Perspective: Draft 2009-11-18)

The purpose of this note is to talk about *Use Cases* - their context, uses, and construction. Along the way I'll give a few simple examples drawn from the business world I have encountered. I intended these notes to be a supplement to an instructor led class, but I hope some of the comments will prove useful to the student or professional who studies alone.

Note: I want to give considerable credit to Alistair Cockburn (*Writing Effective Use Cases*, 2001) for his descriptions and insights into use case construction and analyses. I have carefully studied his approach and hope my interpretations and extensions will support your own further understanding. In any event, I highly recommend your reading his works.

What are Use Cases?

Use cases have a variety of uses although, using them as descriptions for the written *requirements* of a system seems to be the most popular. Other uses will be discussed below.

The intent of a use case is to describe, in writing, a *limited contract* between a user of a 'system' and its expected behavior. That is, what 'value-add' can a user of the system expect upon submitting a request to this system, under prescribed circumstances? (Note that a user could be another module of automation (a service) or other organizational unit).

Use cases can be written by anyone, although most often by business subject matter experts (SMEs) in consultation with their clients. Elaborations of a use case will involve the technical staff as well, all collaborating to validate the content and comprehensions of the use cases. These are essentially *narrative* documents although, some very high level graphics (for example, labeled UML bubbles) can be used as pointers or indexes into the written material. If you like, you can think of use cases as being the 'content' of the graphical bubbles. Note: Ivar Jacobson, the creator of the use case perspective, agrees that the use case is essentially a *textual* document even though most illustrations show graphical 'bubble pictures'. These graphs may best be thought of as a table of contents for the underlying written use case documents.

Describing the Modern Context of Use Cases as Documenting Services

A modern approach, (2000 to the present) is to consider all of these use case narratives from the point of view of describing, documenting, and possibly prescribing the infrastructure of *service oriented organizations*. That is, a use case can be considered as a description of a *service oriented component* of the overall life cycle of any 'system'. For example, if a use case is supposed to describe the interaction of a user with the operations of a particular search engine, then the service oriented perspective would presume that the interaction was as a user against a software module with service-specific features.

This *service-oriented* organizational perspective is a natural evolution that has been going on within the business and technical community for the last decade. (This is the follow-on to the *Object Oriented* perspective that swept the industry during the late 90's). This service approach builds upon the lessons learned from Structured Analysis, Object Oriented Analysis, Aspect Oriented Analysis, Functional Analysis, Client-Server, and Distributed Computing. The business automation result is modules of logic encapsulated in software that displays features consonant with the principles of service orientation which are: loosely

Use Cases and Service Orientation

coupled, autonomous, stateless, independent, contract prescribed, discoverable, reusable, and composable. Note though, that these principles can have a larger scope than just software, for example, business processes in particular and generic work flows.

Modern authors such as Thomas Erl (*Service Oriented Architecture, 2005 Principles of SOA*, 2008), have made a compelling case for constructing software modules following a set of principles, a few of which were just mentioned. Given this service oriented perspective, *use cases* have a natural place in describing the constructed modules, at various levels of aggregation. A good place to start to learn about service orientation and its place in organizational development is to read the Prentice-Hall SOA series edited by Erl.

Use Case Genesis

Alstair Cockburn wrote the seminal book on applied use case construction in 2001, called *Writing Effective Use Cases*. Ivar Jacobson, then of Ericsson Telecom, had much earlier introduced them as part of his Object Oriented (OO) work in the late 60's. In 1992 he published an influential book called *Object Oriented Engineering* that featured use cases as a crucial tool for front end knowledge engineering and, in particular, requirements engineering. From that point on, everybody either incorporated use cases in their methodology or used them as a contrast to *their* approach. Use cases came to be a front end to most all the subsequent software development methods, both OO and non-OO (such as Structured Analysis). Their versatility and appeal springs from their use of natural language, in a structured format, to describe systems in terms of *what that system can do for a user*. The use case clearly explains the 'value proposition' to a user of that system.

Let me simply abbreviate some use case suggestions advanced by Cockburn as well as my own experiences in their use. I will start out in the form of a list, and then expand on each item as time permits. There are a few underlying ideas that need to be described first though.

For the purposes of this tutorial, I'll introduce several terms that have specific meaning in software development and business analysis.

Glossary

System

A *system* is some entity whose purpose and value is determined by the perceptions of external actors, in effect, it is an agreed upon *convention*. In other words, a 'system' is defined and delimited in the eyes of the beholder. In the information technology world, a system is often a *software intensive* system consisting of software, hardware, peopleware, documentation, and networks, although in practice, a 'system' is often limited to a single application program. From a business perspective, a system could consist of one or several business processes, business modules, an enterprise, or even linked enterprises (for example, supply chains). Again, what is to count as a system depends on the perspective of the observer. Most analysis effort therefore goes into setting up *conventions*, as mentioned above, so that a 'system' can be agreed upon by the interested stakeholders. When this task is accomplished, we refer to the system as the *system under discussion* (SuD). That is, once an agreed set of conventions is set up, then we can begin to discuss the 'system' that conforms to those conventions.

Use Cases and Service Orientation

External Actor

An external actor is an entity that is not explicitly part of the system but that can interact with the system and thereby derive perceived benefit. External actors interests and their mode of satisfaction becomes part of the use case description validation. That is, any use case step involving an external actor needs to resolve in such a way so as to satisfy the external actors interests.

Scenario. A scenario is any sequence of steps through the use case that ends in either success or failure. You can think of a use case as a coherent collection of scenarios, a *bundle of scenarios*.

Stakeholder

A generalized kind of actor who is interested in the operation of the system and will derive some kind of impact/benefit from its operation, however indirect.

Use Case

A use case is a natural language description of the process of attaining a designated actor's goal. It describes a purposeful interaction between a system and an external actor, in which the actor intends to achieve some stated goal involving the system. To put this another way, the actor expects the system to render a stated service(s), and so satisfy the stated goal.

Note: There has been an effort by computer vendors (especially those selling graphical software) to cast use cases as graphical entities. That is not the case, although a graphical table of contents is often useful. So, those bubbles you see when viewing a graphical set of use cases are best thought of as entry points into written narrative descriptions.

Use Case Goals and Business Analysis Levels

Depending on the level of the *endeavour*, a use case may have higher or lower level goals. For example, a use case describing how an individual could make an request admission to a university and receive benefits, could take days or weeks (or longer). This would be a high level goal. In contrast, a use case describing how she might download a school policy brochure might take only a few minutes.

How to Approach Use Case Construction?

In thinking about this for a while, it looks to me like an approach that is used in business process analysis (conforming to service-oriented-analysis principles) is the most useful context to place this work in. Let me tentatively set up the following goal levels that match use case descriptions:

Examples of Use Case Levels (using Service Orientation ideas)

Consider the context of a user accessing a web site for various services. As a business analyst suppose you are tasked to write up use cases for various intentions of such a user. You might come up with something like the components below.

Technique level service: a very low level use case that describes something like (validate user name and password). The time frame here is on the order of minutes.

Task level service: a user level use case that can be accomplished in a matter of one sitting. Such as logging onto a web site with all the qualifying questions answered, browsing web

Use Cases and Service Orientation

sites and making a purchase. This could be on the order of hours or a few days.

Activity level service: A high level use case that can extend over many days or even months. This would be characterized by the number of interacting players and the time taken for a common conclusion. This might, for example, be illustrated by a use case requesting home insurance from an agency, or attending traffic school and its legal consequences!

Orchestration level service: This is a very high level use case that might entail a description as to how an organization might convert from a legacy system to another middleware computer system. Or perhaps laying out how clients of a hospital could be moved to another section of the hospital (or worse, to another hospital), under emergency conditions. Generally restricted to interactions with one organization.

Choreography level service: This is the interaction between multiple organizations. A common example here would be a supply chain description involving factories, distributors, and retailers.

A Task Level Description

O.k., enough abstraction, lets take a look at a simplified use case named *Get Product Catalog*. I would characterize this as a *task level service* description since it can be accomplished at a single ‘sitting’.

I’ll comment on its structure after its presentation below. My choice of a single column format is only one of many styles you may encounter.

(1) Use Case id=2; GOAL: Retrieve On-Line Product Catalog
Goal in Context: Requestor can get an on-line listing of company “A’s” catalog.
Description: The context is that an internal or external requestor may request an on-line view of one of the company’s product catalogs.
Scope: SuD - Organization (note: this is not concerned with what software produces the catalog, only the business process to do so) In this case, System under discussion (SuD) is the organization
Primary Actor: Requestor
Stakeholders: marketing dept. (CRM), sales, IT (number of customer hits may revise load balancing protocol)
Preconditions: requestor accesses site location (note that I don’t assume this is a browser log-in, but keep it general for any kind of over the air device)
Postcondition: user interaction is saved and user data captured as desired (for CRM)
1. Company web site is presented to requestor (generic expression for the presentation, not necessarily a screen shot)
2. Requestor is asked for demographic data
3. System identifies type of requestor device
4. Requestor selects catalog option
5. System tailors catalog output to device capability
6. System presents (possibly on-line) view of catalog
7. (option) Requestor goes back to step 4
8. Requestor exits
Extensions: these are alternative paths making up a scenario
0. system goes down (does it matter at what point in the scenario?)
0.1 requestor exits at any point in scenario

Use Cases and Service Orientation

(1) Use Case id=2; GOAL: Retrieve On-Line Product Catalog
Goal in Context: Requestor can get an on-line listing of company "A's" catalog.
1a. web site down
2a. requestor refuses to provide data
3a. unknown device (what to do about this?)
4a. (allow user to suggest additional options?)
5a.catalog option involves media/graphics that customer can't process

Discussion of Use Case

Think of a use case as a *bundle of possible scenarios* where a scenario is a path through the use case resulting in either a success or a failure. The straight forward path that results in success is called the *Main Success Scenario* (MSS). In the case above, that would be steps 1-8 carried out to the satisfaction of the requestor.

General comments on writing use cases:

Writing a use case is no harder, or easier, than writing good, readable text. Remembering that the use case will be read and referenced by multiple parties means that very clear prose is required. Keep in mind that if your use case is well done, it will be the basis for sequential elaborations as the document goes from group to group, task to task, activity to activity, orchestration to orchestration, and choreography to choreography. That is, from a high level use case like the university registration example above, you can expect that use case to be successively elaborated as it goes from vision, to requirements, to analysis, to design, to implementation, to test, to training, to deployment, to maintenance, and possibly to withdrawal/obsolescence/graduation! Analyses that capture value-added stakeholder interactions, last for the lifetime of the 'system' and often form the basis for a system's replacement.

Some cautions to keep in mind:

- Don't try for perfection, write a readable document with detail tailored to the level of the interaction. "perfection is the enemy of the useful"
- Try to capture the 'big picture' in broad strokes so that a major feature doesn't get lost. This has to do with 'scope', which determines what is 'in' and what is 'out' of scope for the system (this is the most volatile feature of analysis at the beginning, middle, and end!)
- Cockburn (*Writing Effective Use Cases*) recommends identifying the major actors and their goals as a good way to start the writing cycle. A context diagram showing various events impinging on the system is a very useful starting point which will help to also determine scope.
- Don't prematurely place interface interactions in the use case. That is, don't describe data fields yet or which buttons correspond to those data fields. (That detail should come later as the use case is elaborated, besides, your process may be implemented manually).

Cockburn lists the following writing steps (amended slightly)

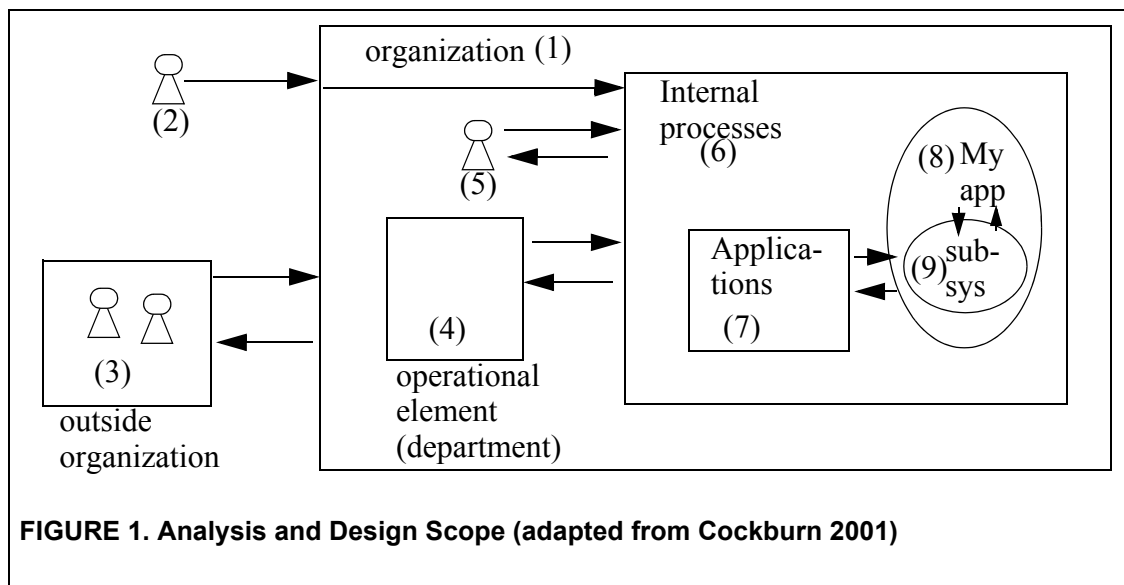
- set up the scope of the system and its boundaries and constraints.

Use Cases and Service Orientation

- brainstorm to determine the actors and goals
- capture the outermost use cases to see how far the effect of the system extends.
- capture stakeholders interests and concerns. note, stakeholders include actors but also others such as government regulators, stockholders, potential customers, and other businesses.
- work on selected use cases (don't try to do them all).
- only *list* extension conditions and then *later* try to write their steps. The extensions represent alternatives that usually involve business decisions and so require more care and often a whole lot more time than the main success scenario. For example, for a customer trying to access your web site, what do you do when they log off after entering their credit card number for a purchase but before pressing submit?
- identify what synthesis and decomposition, merging and separating, can be done with the various use cases. Remember that the more use cases the more maintenance required.

Design Scope

Cockburn provides a very handy diagram of the various design scopes you might encounter. I have modified it slightly since use cases apply to both manual as well as computer aided processes.



Putting the Use Case in an (Cybernetic) Organizational Context

The diagram below is a first cut at putting the business analyst (Subject Matter Expert) 'in the organizational context. Writing a use case usually involves cutting across multiple departmental and group lines. Be aware of this and try to determine where your analyses ought to be situated. The diagram below is from the managerial cybernetics perspective of Stafford Beer who treats an organization as a sequence of recursions of standard components. I have labeled some parts of one level of recursion to give an idea of the types of processes that a use case might be called upon to describe.

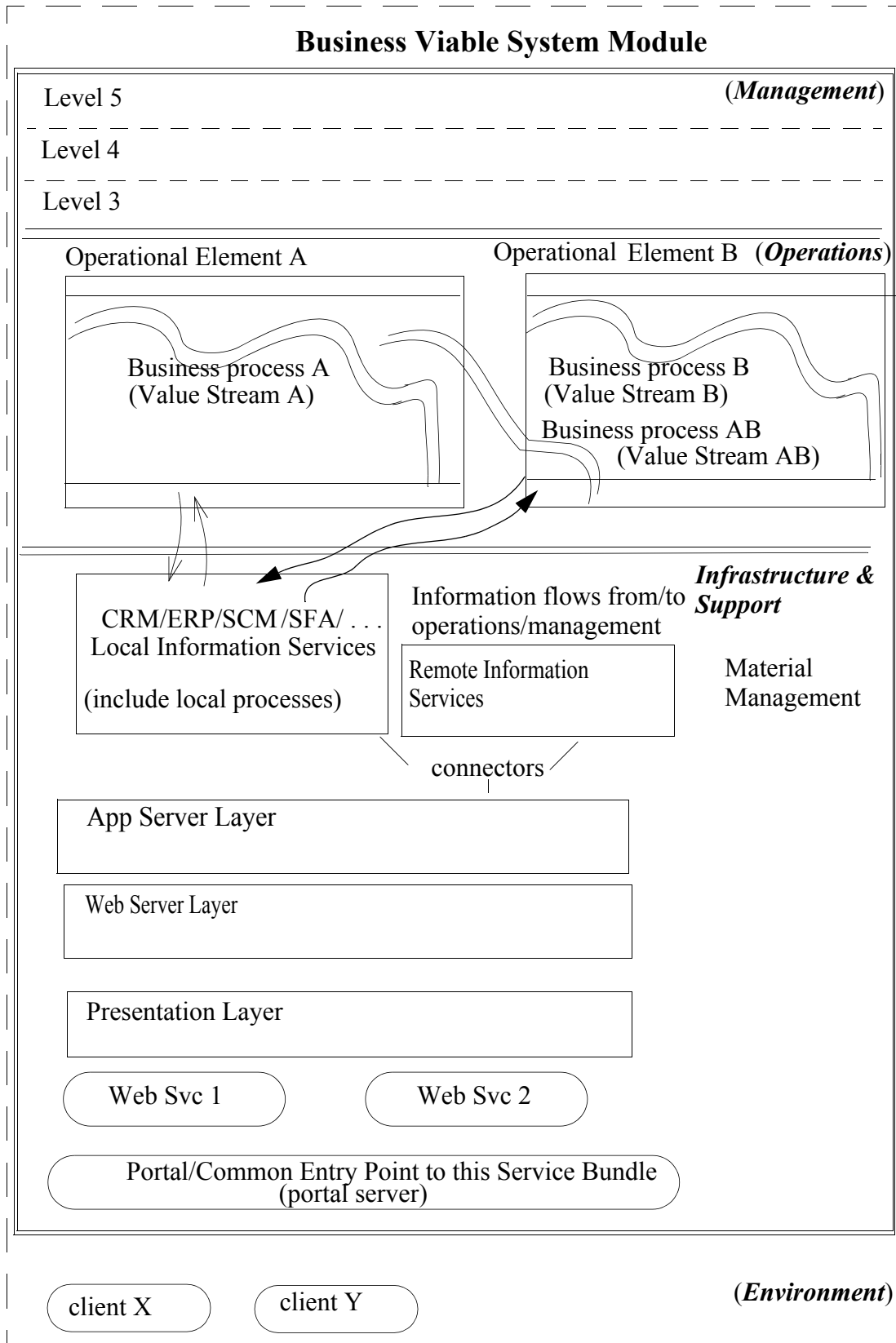


FIGURE 2. Business Viable System Module - Client Targets Shown in the Environment

References

- Cockburn, Alistair, (2001), *Writing Effective Use Cases*, Addison-Wesley
- Rucker, Rob (2007) MOISE & DESMIA, milagrosoft.com (web site)