Business vs. System Use Cases – Part one

Author: Martin Langlands and Charles Edwards
Version: 1.0 Date: 6 May 2008

Abstract

Use-cases are now all but universal as the basic concept for specifying requirements of commercial information systems. However, one area that causes problems is distinguishing between “Business” and “System” Use Cases. The aim of this three-part series of articles is to shed some light on this issue by highlighting the *differences* between the two, and proposing a diagrammatic way of showing how they are *related*. This is illustrated using a more detailed and meaningful example than is used by many introductory texts.

Introduction

Use-cases are now all but universal as the basic concept for specifying requirements of commercial information systems. However, there is still a huge amount of confusion in how best to apply the idea. One area that continues to cause particular difficulty is distinguishing so-called “Business” and “System” Use Cases.

Why has this confusion arisen? Well, we need look no further than the man who gave the Use-Case idea to the world: Ivar Jacobson. In his 1994 book “The Object Advantage” [JACOBSON94], he introduces the concept of “The Use Case” as: “A use case is our construct for a business process.” (p104) So far, so good. But four lines later, he says: “A use case is a sequence of transactions in a system …”. No wonder there’s confusion: does the concept apply to business processes or computerised systems?

A good example of this confusion came up recently when a client asked: "*We are generating a Business Use Case Model for a project. The Project is mainly to develop a system which can enable users to be notified by WAP/SMS on their cell phone regarding their preferred stock prices, important Emails, news, weather etc. Now which element shows the ‘Cell Phone’ usage in the diagram? A business actor, a business worker, a business entity or a use case? Also, can Business entities be shown in Business use case diagrams?*".

Now, when faced with such a question, we’re usually tempted to respond: “Read the standard literature!”. Two of the most popular books on Use-Cases, by Cockburn [COCKBURN01] and Bittner & Spence [BITTNER03], have a lot of useful advice on exactly this. However, it is still surprisingly easy to absorb all that these and other authors have to say, and still be lost when actually starting to work on a real project. The aim of this paper is to help those stuck in this bind.

We’ll come back to answer this client’s questions directly later in this series. But before doing so, we need to do a lot of preparatory work.

Two Concepts

OK, so if there’s a possibility of confusion, how should we apply the terms Business Use-Case and System Use-Case?

Well, first, it’s helpful to note that there is commonality between the two concepts: both define a pattern of repeatable interaction or behaviour that is intended to deliver a result of value to the Actor.

© 2008 Martin Langlands and Charles Edwards

However, it's equally important to note there *are* clearly two useful but distinct concepts here:

- A **Business Use-Case** is to do with "using a business": this recognises that businesses[1] are created and organised in order to *do things for people* – mainly customers, but also other "actors". So a Business Use-Case is a way in which a customer or some other interested party can *make use of the business* to get the result they want – whether it's to buy an item, to get a new driving licence, to pay an invoice, or whatever. An important point is that a single execution of a Business Use-Case should encompass *all* the activities necessary to do what the customer (or other actor) wants, and also to do any necessary internal tidying-up. (We'll see a little later an example of what we mean by this last point.) So the duration of a BUC execution can vary greatly, depending on its nature. Some BUCs, like withdrawing cash from an ATM, can be done in less than a minute; others, like ordering goods for delivery, or getting a new phone line installed, can take days, weeks or even longer.
- In contrast, a **System Use-Case** is a way in which a user of a *computer system* can *make use of the system* to get the result they want. This will typically be something we can readily imagine as being done in a single sitting on a single computer, usually with a single UI, or a small number of closely-related screens such as a wizard, and taking maybe between a couple of minutes and a half-hour at most. Alistair Cockburn suggests that a useful guideline is the "coffee-break test": once the user has completed (a single execution of) the System Use-Case, s/he can take a coffee-break with a clear conscience. The system use case also avoids all manual issues such as "file the printout" or "phone the customer once the order is confirmed", etc.

A Realistic Example

Let's look at an example that allows us to focus on the differences. This example is a little more complex than many that are presented at the introductory level, but therein lies the root of much of the problem: examples used in many books and articles just have insufficient complexity to illustrate the kinds of thing that come up in real life, and often are too simple to allow us to see the differences between Business and System Use-Cases.

So let's take SupaStores, an imaginary grocery chain that allows customers to place orders on its website, www.supastores.com, and have the orders delivered to the customer's home. We'll look at an example Business Use-Case (BUC) and some System Use-Cases (SUCs) to highlight the key differences. We've assumed a basic familiarity with UML modelling concepts.

The Business Use-Case

What Business Use-Cases might be of interest for SupaStores? Let's focus on the most obvious:

- ***Buy groceries***, from the customer accessing SupaStores to completion and recording of delivery for that customer.

A first-cut Use-Case diagram for this BUC would look very simple (Fig 1). This shows the principal actor – the Customer – and one Use-Case. The result of value to the customer would be the correct Grocery delivery.

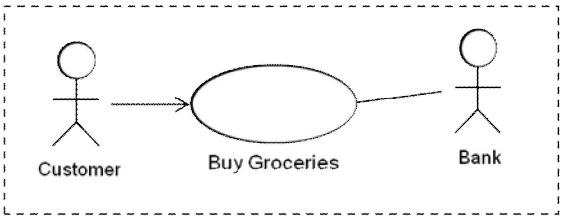

Figure 1: "Buy Groceries" Business Use-Case Diagram

[1] Calling these "businesses" can itself be (unintentionally) misleading. A large part of the economies of all countries is conducted by organisations other than commercial businesses – local and central government agencies, charities and other non-profit organisations, etc. Their activities equally fall into this area. Strictly, we should be saying something like "Businesses and other organisations", or maybe just "Organisations"; but we'll stick with convention.

© 2008 Martin Langlands and Charles Edwards

We also need to show one supporting actor, the Bank, because it is a distinct business entity required to complete the job: specifically, it needs to approve the customer's bank-card transaction. Note that we don't show on *this* diagram the "internal actors" in the SupaStores business that perform the use case – neither the employees such as warehouse and delivery staff, nor any systems (computerised or otherwise) involved – because they are "inside" the Use-Case. This Business Use Case Diagram describes "Black Box" behaviour; that is, it shows only the business interaction, not the internals of how a Use Case is implemented by SupaStores.

One of the most important things to understand about any given BUC is its *scope* – what triggers it, and what marks its completion. In this case, the event that starts this BUC is the customer accessing the SupaStores website to place an order. The event that completes the BUC is recording the details, such as the time and date, of the successful delivery. Note that the duration of this BUC could be from around a day to a week or more.

In order to see how BUCs and SUCs differ, and also how they relate to one another, we'll have to open up this BUC and take a look inside.

The first problem we meet is: UML, the modelling language that defines the Use-Case concept, says nothing about what the inside of a Business Use-Case (or any Use-Case, for that matter) should look like! However, a sensible approach – and one that is very useful, as we'll see – is to look at the sequence of steps, or *activities*, that constitute the BUC. Figure 2 illustrates this diagrammatically.

We've drawn this more-or-less as a UML Activity Diagram, with a very simple drawing palette:

- rounded rectangles for activities (aka steps)
- simple arrows showing activity sequence
- a bullet for the trigger start event, and a bullseye for the end event
- swim-lanes representing which roles or organisational units are doing what activity – the Customer and the Bank, as actors external to the organisation, get their own swimlanes, as does each internal organisational unit involved.

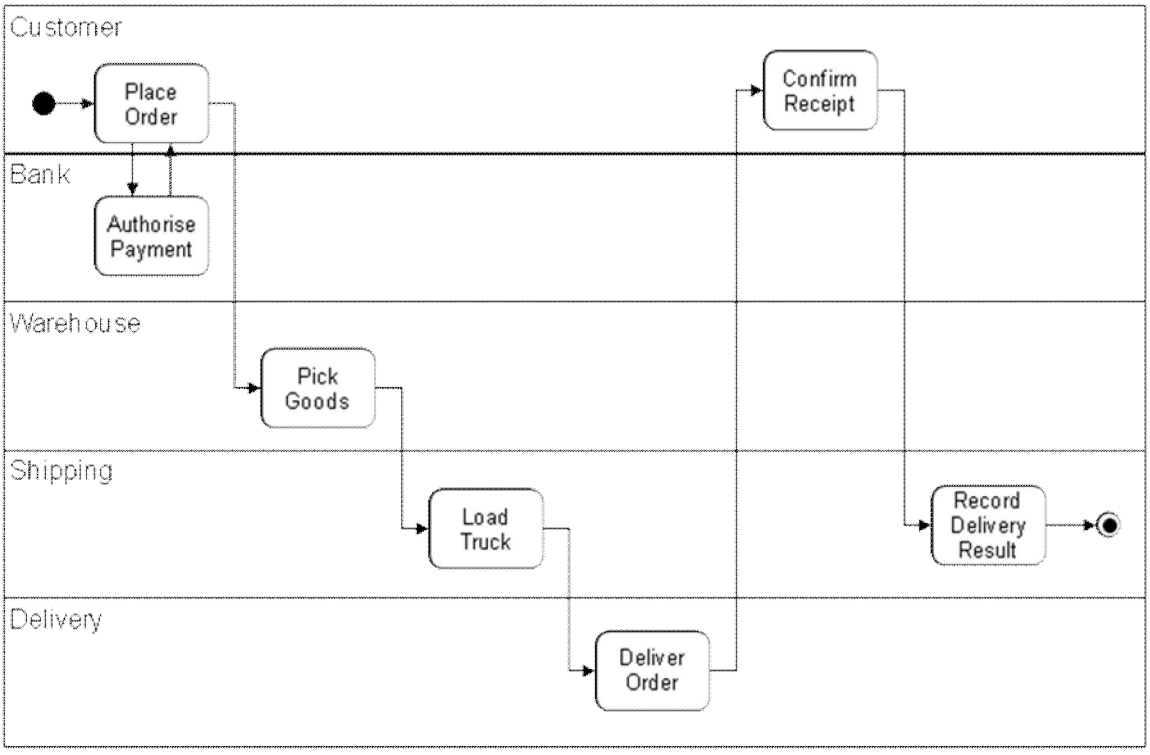

Figure 2: "Buy Groceries" Business Use-Case Activity Diagram

© 2008 Martin Langlands and Charles Edwards

As mentioned, this is a non-trivial example, but one that we hope is readily understandable without having specialised domain knowledge. We've deliberately kept this as simple as possible, without getting into the complexities of different modelling approaches or notations such as Business Process Modelling Notation [BPMN]. Also not shown are the different possible alternate flows or terminations – eg. the BUC could terminate immediately after the first "Place Order" activity if the customer decides to cancel, of if the card payment fails; in "Deliver Order" the driver could be unable to find the address, or the customer could be out; and so on. Showing these would result in a considerably enriched, but more complex, diagram

Now, although the diagram sets out the basic steps and their sequence, we can clarify our understanding further with a textual description. Figure 3 shows the Use-Case description based on Cockburn's widely-used template.

Name : BUY GROCERIES ***Business Use-case***

Brief Description : In this Use-Case, a Customer places a new order for delivery. SupaStores assembles the order, delivers it to the customer, and records the result of the delivery

Principal Actor : Customer

Precondition : Customer must reside in the country of the business to qualify for delivery.

Main Flow

Step Name	Description
Place Order	The Customer contacts SupaStores; selects the required quantities of products from the product catalogue; selects a delivery slot from those available; offers payment by an approved method; advises delivery address; and confirms all order details.
Authorise Payment	The Bank confirms that the offered payment is valid, and authorises the payment.
Pick Goods	For all confirmed orders for a forthcoming delivery run, the SupaStores warehouse finds the order items on the shelves; assembles the orders, noting any out-of-stock items or substitutions; packs them for delivery; and prints a Delivery Sheet for each order.
Load Truck	For a particular delivery run, the SupaStores Shipping department determines the planned delivery route, and loads the packed orders (accompanied by their Delivery Sheets) into the assigned truck in the correct order for the route.
Deliver Goods	A SupaStores Delivery driver makes a delivery run, following the planned route; for each order on the run, the driver finds and confirms the delivery address, and delivers the order to the recipient
Confirm Receipt	For each delivered order, the Customer confirms that the delivered order corresponds to the items on the Delivery Sheet (including any unavailable and substituted items), indicates the delivery time and order acceptance, and notes any comments.
Record Delivery Result	After the completion of a delivery run, the delivery driver returns the Delivery Sheets to SupaStores Shipping department, who record the details of each order, including the time of delivery, and any notes made by the customer.

Figure 3 : Text Description of Business Use-Case
"Buy Groceries"

© 2008 Martin Langlands and Charles Edwards

This deliberately shows only what Cockburn calls the *Main Success Scenario* or *Main Flow* for this Use-Case – ie. it still omits any alternative or exception flows such as those leading to the different terminations considered above.

The textual description introduces some important subtleties that are not indicated on the activity diagram, such as the handling of out-of-stock items and substitutions. It also illustrates – in the step *Record Delivery Result* – an example of the point made earlier about "tidying-up" at the end of the BUC: although the customer has received the goods in the penultimate step, and the BUC is therefore complete as far as s/he is concerned, we can't regard it as finished from SupaStores' viewpoint until the final step is done.

Note that this is, in Cockburn's terminology, a "white-box" view of the BUC; we've gone beyond describing only the dialogue between the actors, to look behind the scenes and see how the BUC is implemented within the business. In this view, we've given a swimlane in the diagram to each Actor, and also to each internal SupaStores organisation unit involved.

A black-box view, showing only the interaction across the interface between the primary actor (the customer) and SupaStores, would be limited to the steps "Place Order", "Deliver Goods" and "Confirm Receipt", and on the diagram would need swimlanes (if we chose to show them) only for the Customer and SupaStores.

The intention at this stage is to show the activities that constitute the BUC – *what* gets done – without (yet) asking *how* they are done, in particular what computerised systems do or could support each activity.

Next time

In Part Two we will show the System Use Case equivalents for this example and highlight the key differences between the two types of Use Case. In part three we will look further at a Process Use Case Support Diagram and make observations and conclusions.

References

[BITTNER03] Kurt Bittner & Ian Spence, "Use case Modelling", 2003, Addison-Wesley, ISBN 0-201-70913-9

[COCKBURN01] Alistair Cockburn, "Writing Effective Use Cases", 2001, Addison-Wesley, ISBN 0-201-70225-8

[JACOBSON94] Ivar Jacobson et al, "The Object Advantage", 1994, Addison-Wesley, ISBN 0-201-42289-1

[BPMN] Object Management Group – see http://www.omg.org/bpmn/

Contact details

Martin.Langlands@blueyonder.co.uk

Charles.Edwards@processwave.com

www.ProcessWave.com and www.AgileEA.com

© 2008 Martin Langlands and Charles Edwards