Business vs System Use Cases – Part three

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 article series 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 many introductory texts.

Previously in part one and part two

In Part One we started to discuss the difference between introduced the terms Business and System Use Case, and showed our worked Business Use Case example.

Part two showed worked examples of system use cases based upon our business use case.

The Process Use Case Support (PUCS) Diagram

In order to fully appreciate the value of the separate Business and Systems Use-Case concepts, we need to understand not only where they *differ*, but also how they are *related.* The fundamental nature of the association is:

System Use-Cases *support* Business Use-Cases

– meaning that the System Use-Case provides some automated means of doing part or all of the Business Use-Case.

We are proposing here a new diagram-type, which we call the *Process / Use-Case Support Diagram*, or PUCS Diagram, to provide an immediate visual representation of this support. Figure 5 (discussed in Part 2 of this series, and repeated on the next page) shows an example. It is a simple but effective addition to the UML diagram set, using existing UML concepts, and with additional stereotypes that could easily be implemented in a simple UML profile.

Each PUCS Diagram is drawn for one Business Use-Case. The key elements of the diagram are:

- A UML Activity Diagram showing the steps (activities) in the Business Use-Case (BUC) and the transitions between steps
- A UML Use-Case Diagram for each “supporting” System Use-Case (SUC)
- An association, named “supports”, from each SUC to each business activity / transition that it supports. (Some activities or transitions may have no such SUC support.)

The nature and extent of the support will vary from business to business and from one system to another. For example, the support given by a SUC to a business activity can vary

- … from simple data presentation and / or capture, such as a straightforward data-entry UI with limited validation …
- … through partial decision-support, such as an insurance rating tool …
- … to full implementation of the whole step, such as an automated share-trading system.

© 2008 Martin Langlands and Charles Edwards

These variations could be indicated on the PUCS diagram with stereotyping of the "supports" association, although care needs to be taken not to attempt to show too much on the diagram.

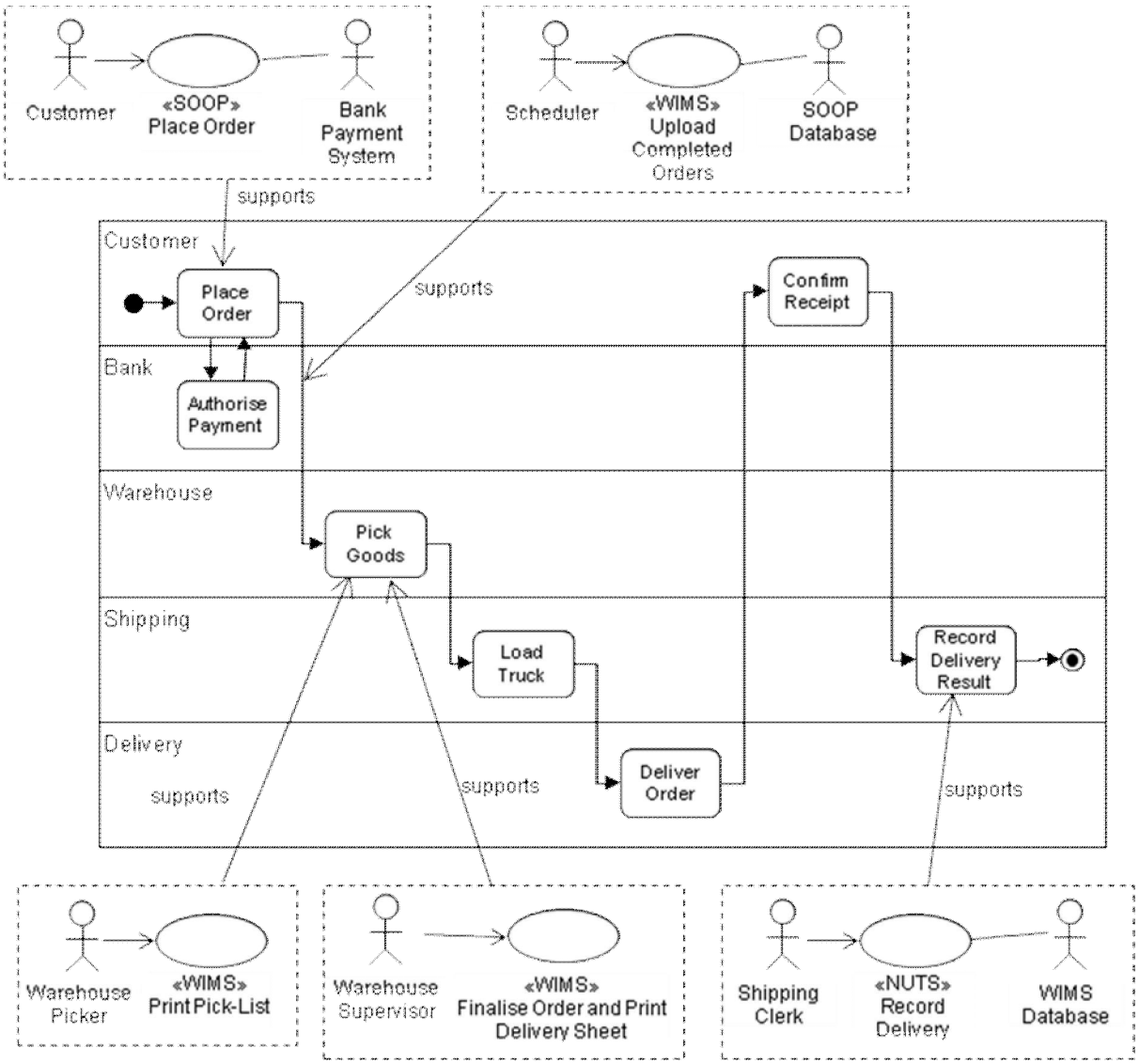

Figure 5: "Buy Groceries" Business Use-Case
Activity Diagram with supporting System Use-Cases

What were we doing in the above example?

In following the SupaStores "Buy Groceries" example discussed in Parts 1 and 2 of this series, the astute reader may have asked: "Why are we doing this? What's the 'project' here?".

This is a fair point. From what we said, many of the SUCs in question are realised by old legacy systems. When those systems were first developed, there's a very good chance that Use-Case Analysis did not figure as a technique. So why are we going back in time now to represent their functionality as SUCs?

Well, apart from the obvious answer "to illustrate this article", a real-world example would be a project that wanted to improve system support for a part of the process – say, the back-end "delivery" steps. Let's say that the SupaStores executive in charge of on-line orders has identified a problem that the keying of the delivery-sheet details into the NUTS system (in the step "Record Delivery Result") is error-prone and often done late, resulting in poor manage-

© 2008 Martin Langlands and Charles Edwards

ment information, and incurs extra staff costs to do the data entry. She has sponsored a project for IT to look at the problem and come up with a solution.

If we look at the problem in the fairly narrow terms stated by the project sponsor, we are likely to come up with a narrow solution. If we don't have the benefit of the BUC model, and just focus on the use of the NUTS system by the shipping clerk, we might conclude that the problem lies partly in a poor design of the delivery-sheet, or an unfriendly old character-based NUTS UI, and tackle these issues. However, this would be missing an opportunity.

We can propose a more imaginative solution by taking a wider view to look at the whole Business Use-Case as in the example. Now we can see where the recording of the delivery-sheet fits in the bigger picture, and ask: "What opportunities are there to improve the whole process, by enhancing systems support, and / or by other changes?".

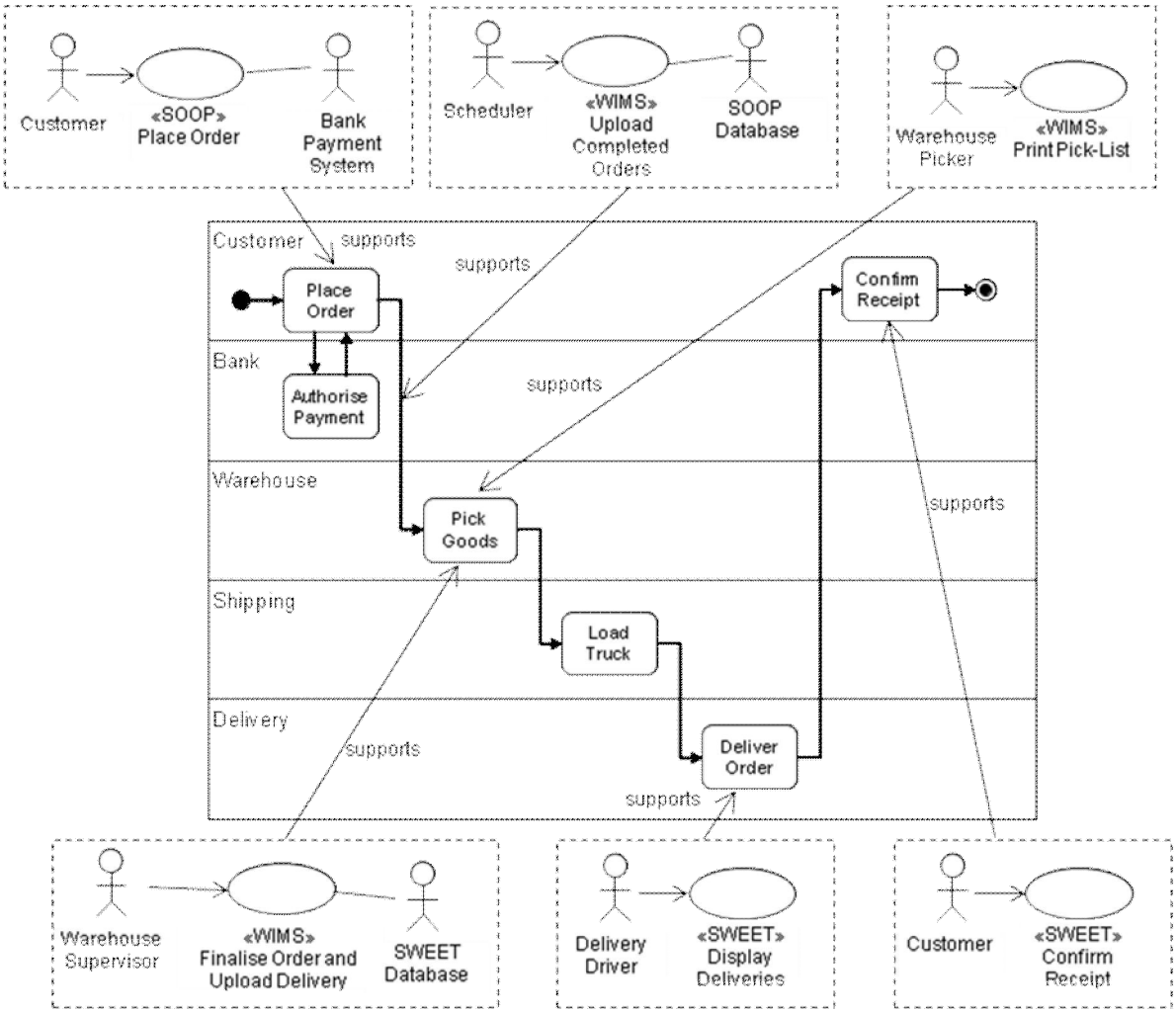

Figure 7: "Buy Groceries" Business Use-Case
Activity Diagram with supporting System Use-Cases – Proposed enhancement

We could, for instance, propose giving the delivery-drivers web-enabled hand-held devices which access and display delivery lists, and which are used to capture customer signatures. This would be supported by a new system, SWEET (SupaStores Web-Enabled Enhanced Tracking) that would maintain a database of current deliveries. Now we can completely eliminate the costly and error-prone "Record Delivery Details" step. SWEET realises a new "Confirm Receipt" SUC, in which the customer (the SUC actor) uses the handheld screen to record delivery details, which are immediately recorded on the SWEET database. Figure 7 shows the result on a new PUCS Diagram, showing the changed BUC Diagram and a new set of supporting SUCs.

© 2008 Martin Langlands and Charles Edwards

We can also see opportunities for further improvements – for instance:

- We can help the delivery driver view the deliveries in a delivery run, perhaps displaying these dynamically on a local-are map on the handheld (SUC "Display Deliveries")
- If we further add GPS functionality to the handheld, allowing SWEET to track the location of each delivery truck, we can provide the customer with a "when's my delivery coming?" function that she can access through her own home PC (this supports a different BUC, so it's not shown on Fig 7).

Of course, all of these ideas, and any others we come up with, need to be subject to a proper business case assessment, including a cost / benefit analysis, and not all will necessarily be approved.

But what we have done is to show a straightforward but powerful representation of both the underlying *business process* and the current and potential *systems support* for that process. This representation is readily understood by business and systems people, and is an ideal tool for discussing, scoping and planning the systems-development aspects of a business change project. Newly-defined system use-cases identified this way can, of course, go on throughout the project as the basis of detailed specification, design and implementation. In addition, the fact that our legacy systems may not have been originally specified or implemented using explicit SUC modelling does not detract from the usefulness of the System Use-Case concept for describing the functionality of those systems retrospectively.

Three Caveats

We have attempted in this series of articles to help modellers with the sometimes tricky problem of distinguishing Business Use-Cases from System Use-Cases. However, there are still some issues that can lead to possible confusion.

"The Business as a System"

Sometimes we encounter the view that "there's no real difference between the two – after all, the business can be viewed as a system, can't it?". This is as much a question of definition of the terms: yes, we can say this, if by "system" we mean something like "a collection of interacting elements, organised as a whole and collaborating to achieve a purpose". While defensible – and there is, of course, a whole range of disciplines related to understanding "systems" with this meaning – it seems to us unnecessarily confusing in this context. When we say "system" here, we specifically mean *computer(ised)* or *automated* system, and not *"business-as-a-system"*, and we urge readers to do the same.

"System Boundary = Business Boundary"

In many cases, the scope of the System Use-Case is exactly the same as the Business Use-Case, and the SUC completely implements the BUC. Another way of saying this is that the boundary of the system across which the SUC interaction takes place is the same as the BUC boundary – they are "coterminous". Examples include:

- Web sales of downloadable software, music or other content
- On-line flight check-in
- Withdrawing cash from an ATM (*but not all possible ATM-based BUCs – for example, "requesting a cheque-book")*
- Making a purchase from a vending machine
- Connecting a phone-call in an automated exchange

The key point about these BUCs that makes them completely implementable in a coterminous SUC is that there's no need for separate steps to make decisions or handle physical goods, or any inherent delays between steps. The SUC may need to involve other systems,

© 2008 Martin Langlands and Charles Edwards

as further secondary actors – for instance, to actually retrieve and deliver the software or music, check and update account balances and so on; but it's still the case that one System Use-Case implements the complete Business Use-Case.

Examples such as these are, of course, common and important, and are likely to become more so as digitisation of products, and the world in general, gets wider. They also tend to be beloved as examples by writers of text books and articles. Unfortunately, this does often obscure the real distinction between BUCs and SUCs – because in these cases there is often little value in writing descriptions of both.

However, there are of course still very many cases, such as the SupaStores example, where the SUC is not going to completely implement the BUC (short of a Star Trek transporter device to deliver cereals, steaks and kitchen-roll digitally).

"One Level Too Low"

A third problem that we sometimes encounter in distinguishing BUCs and SUCs is that modellers misapply the terms. This tends to be more common with technically-oriented people – perhaps developers doing Use-Case modelling – rather than "business analysts".

We've observed that they apply the term "Business Use-Case" to what we call a "System Use-Case"; so, for example, they'd define "Place Order", "Print Pick-List" and "Record Delivery" as *Business* Use-Cases. The terms have been moved "one level down" as shown in Fig 8.

The thinking seems to be that "these are things that the business user touches", so they must be "business"-things. In this view, the term "*System* Use-Case" is then applied to things that happen *inside* the system – in effect, the things that (in our view) would appear once we opened up the description of the SUC to the white-box level. They correspond to system design or implementation artefacts such as components, operations, or APIs that have no direct element of actor interaction or dialogue, and therefore can't count as Use-Cases at all. (We might label these *mis*-use cases.)

We say ...	***They say ...***
Business Use-Case	???
System Use-Case	Business Use-Case
Design / implementation Artefact	System Use-Case

Figure 8: Terms moved one level down

A problem with this misuse of the terms is, of course, that there's then no term left for what we have called a Business Use-Case. Unfortunately, since this arises from a technically-focused mindset, this isn't seen as presenting a problem – the true "business" thinking is way out of sight in any case. What's needed is to take a step back and appreciate that the system solution is there to support, and is intimately related to, a view of what's going on at a true business level, before we consider systems issues at all.

Revisiting the Original Question

We're finally in a position to discuss our client's question that we raised at the start of Part 1 of this article: "*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*

© 2008 Martin Langlands and Charles Edwards

worker, a business entity or a use case? Also, can Business entities be shown in Business use case diagrams?"

Unfortunately, there's no single answer to this. What we need is a longer discussion with our client. It seems possible that the questioner is suffering from the "one level too low" problem, among others. However, making some guesses, the response would be something like the following.

- Firstly, an important point is that the project team seems to be placing too much reliance on Use-Case Diagramming as a technique. Use-Case diagrams as such show very little; don't get too hung up on what you can expect to get out of them, and don't think that by drawing Use-Case Diagrams you're "doing Use-Case modelling". Move on to textual descriptions as soon as you can, initially at the "use-case brief" level and then on to main scenarios and finally alternative flows.
- To take the last part of the question next – "*can Business entities be shown in Business use case diagrams?*". The answer is most emphatically "Yes!". Business entities, such as the Customer and the Bank in our SupaStores example (see Figure 1 in Part 1), are critical elements of a BUC Diagram. Just don't start to think of "cell phones" as business entities!
- The key question here is: What *is* the core Business Use-Case? One possibility is something like "Advise Customer of Notifiable Event", where the primary actor is internal to "us" (ie. the company offering the service) and the customer – and probably the phone network provider – are supporting actors.
- Having decided this, create a white-box Business Use-Case Description – in both diagrammatic and textual forms; then determine what System Use-Cases are needed to support each step. The relationship between the BUC and SUC should be shown with a PUCS Diagram.
- Finally – and to answer the core question as posed: "*which element shows the 'Cell Phone' usage in the diagram?*" – we'd say: None of them! The Cell Phone should not appear at all on any of these diagrams – not the Use-Case diagram, nor Activity diagrams, nor PUCS diagrams, at either Business- or System Use-Case levels. The Cell Phone itself is simply the hardware that supports parts of the overall picture. To show it would be as inappropriate as showing the user's PC on the "Place Order" SUC Diagram, or indeed the "Warehouse Building" or "Delivery Truck" on the BUC Diagram in the SupaStores example.

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