IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4227

So What's In A Use Case?

D. C. McDermid
School of Computing and Information Science, Edith Cowan University, d.mcdermid@ecu.edu.au

ABSTRACT

This paper challenges established wisdom with respect to how to elaborate use cases. Use cases are refined into (business) objects which are then modelled by identifying methods and data. The research reported in this paper indicates that there are other better constructs for modelling use cases, at least initially, and that (business) objects are not a particularly good medium for discussing requirements with users. This paper describes the arguments leading up to these conclusions.

INTRODUCTION

This paper will describe some conclusions from two action research studies concerned with modelling businesses and their rules. Initially, business objects were used as a means of capturing and documenting the requirements of the information system and so the ability of business objects to describe requirements in the early stages of developing a system was tested in these studies. It was found that business objects, as currently used and defined classically in the literature did not contain constructs which were **directly** conducive to requirements gathering with users and neither did they facilitate presentation and discussion of requirements at an appropriate level of abstraction. It was found that a more appropriate vehicle for analysing requirements at this stage (particularly with users) was brought about by structuring a **use case** into a **business rule and by using constructs other than methods and attributes**. Having said that, business objects were used in the creation of business rules and so there is an intrinsic relationship between the two.

This paper will attempt to explain the relationship between use cases, business rules and business objects. In doing so, it highlights some inadequacies of business objects as a medium for gathering and expressing requirements at an early stage of development. The paper proceeds as follows. The next section critiques the difficulties in identifying objects, methods and attributes from use cases. This is done by presenting simple problems in requirements gathering and leads to the conclusion that structuring use cases is more desirable. In the following section some background to business rules is given. Business rules are proposed as an alternative to a use case i.e. a use case with structure. Also a documented example of a business rule is provided which is developed from the earlier examples.

CONCEPTUALISING REQUIREMENTS

In a typical requirements gathering session, users might be asked to brainstorm 'their requirements'. Table 1 contains statements that are typical aspects of requirements of use cases in an order processing system.

Table 1: Typical requirements in an order processing system

Orders sent by mail or telephone
Omission on order line leads to deletion of that order line
Credit balance >= order value to accept order, otherwise reject
'Bad' customers do not get credit sales
Stock qty >= order qty for normal order, otherwise outstanding
One invoice for one order
Sum of payments = order value - sum of credit notes
One order may have many credit notes
Many payments per invoice possible

The problem with such requirements is that they are **unstructured**, that is there is no predefined format of the nature of the constructs within any requirements statement. Consequently, and especially when brainstorming, users are apt to identify partial or incomplete aspects of a requirement and thus a use case. In other words, arguably important aspects of a requirements might not be captured.

Use cases are typically refined by successive rewriting of the use case until the analyst is satisfied that all aspects of the use case are clear (Jacobson et al 1992) but this is a rather subjective process with no guarantee that requirements have been completely captured.

Once refined, the objects within a use case are identified. In particular, the methods and attributes of (business) objects are identified from the refined use case. So we have a rather haphazard process by which 'scraps of text' in a use case are mapped onto objects, attributes of objects or methods of objects. This begs the question as to whether objects, methods and attributes are typically articulated during the brainstorming of requirements. Tables 2 contains typical order processing requirements that have been 'structured' in a manner consistent with the constructs of objects. In table 2 we can detect objects, methods and attributes.

Table 2: Typical object style thinking

Receive customer order
Reduce credit limit by X
Reject order
Create new order
Send invoice
Generate credit note

However, this kind of encapsulation does not go far enough in three ways. Firstly, it omits any articulation of conditions or the criteria by which activities in business are undertaken. Refer back to table 1. There are numerous requirements there that refer to conditional circumstances (e.g. credit balance >= order value to accept order, otherwise reject) which are just not captured explicitly in table 2 but are assumed will be detailed later within the method. Arguably it is important that due attention is paid to completeness of requirements as early as possible in requirements gathering.

The second omission is that the statements in table 2 say nothing about the roles of the statement in the system. Table 3 contains an additional column over table 2. This columns structure the statements into E for an event in the system, T for a trigger to an activity in the system and M for a message leaving the system.

Table 3: Adding structure to methods

Receive customer order	T
Delete line	E
Reject order	E
Create new order	E
Send invoice	M
Generate credit note	E

Thirdly, there is no explicit formalisation of state in table 2. In table 1, reference is made to 'bad' customers. 'Bad customer' in this system is one state that a customer may occupy and of course there are rules that follow these different states.

This paper appears in *Issues and Trends of Information Technology Management in Contemporary Organizations,* the proceedings of the Information Resources Management Association International Conference.

BUSINESS RULES

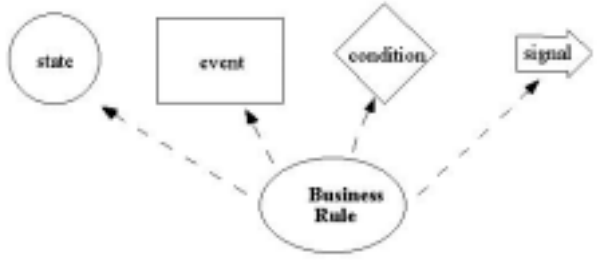
The definition of a business rule provided here suggests something of the context and nature of a business rule as well as identifying its constructs. It is defined as *'...an explicit state change context in an organisation which describes the states, conditions and signals associated with events that either change the state of a human activity system so that subsequently it will respond differently to external stimuli or reinforce the constraints which govern a human activity system'* (McDermid 1998, p20).

As far as the relationship between a use case and business rule is concerned, we may say a business rule is simply a use case with structure. As far as the above definition is concerned, a use case will either change the state of the system or not. Business rules are elaborations of use cases in that they contain four explicit constructs. These are states, events, conditions and signals. **States** reflect the status of an object of interest at any given time, so for example a manufacturing work order might occupy the states *not started, in progress* or *completed*. **Events** are actions carried out internally by the organisation which change the state of one object. They are considered to be instantaneous occurrences that reflect the organisation's policy on what should happen in a particular circumstance e.g. cancel work order. One important role of the event is to avoid specifying processing detail. This was seen as a pitfall at this early stage. Events differ from methods in that methods may not change states (for example a method may simply reveal data in response to a service request). **Conditions** define the criteria by which objects of interest in the business move from one state to the next as events take place. Sometimes, many conditions must met in order for an event to take place, thus increasing complexity. It is argued that modelling conditions without the context of states and events (and vice versa) is far less powerful or useful. Lastly, **signals** either enter or leave the human activity system. Signals that enter the system will typically initiate activity within the system and so these are called **triggers**. Triggers may be external such as a customer sending an order or internal such as one department sending a document to another department which then triggers off some activity. Further, a trigger may be a time trigger e.g. an activity beginning at the start of the day or the end of the month. Those signals which leave the system serve the purpose of informing those outside the system of what has occurred inside the system and therefore are referred to as **messages**. Thus, though some might argue that the idea of a condition is at the heart of a business rule (Loosley 1988), the related constructs of state, event and signal provide a context for the business rule. So, as an aid to memory we might say:

Business Rules = States + Events + Conditions + Signals

The abstraction of these constructs is shown diagrammatically in figure 1.

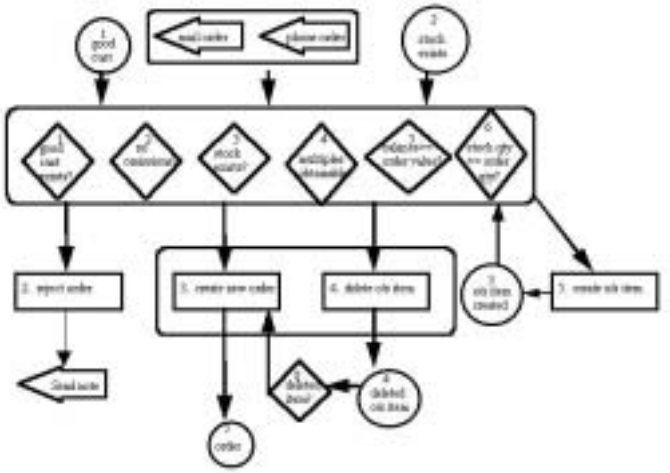
Figure 1: Abstraction of key constructs of a business rule



Over the course of the action research studies, six different versions of a (structured) BRD were developed (McDermid, 1998). Earlier versions contained fewer constructs; as each version was evaluated, it was concluded that there was a need for additional constructs to ensure completeness in the description of a business rule. The four major constructs identified were seen as the minimum for holding a reasoned discussion with users at this stage. Observe that data was not one of the four constructs and neither was method, although there is some overlap between the event construct and method. By the end of the studies a methodology for constructing the BRD had been developed.

Figure 2 shows an example of a single formal business rule. States are represented by circles, events by rectangles, conditions by diamonds and signals by thick arrows. The softbox is a Harel blob (Harel 1988) which acts as an encapsulator of constructs. The example in figure 2 is the most complex state change context so far modelled; the vast majority of business rules are much simpler involving typically no more than five or six constructs. While a full description is outside the scope of this paper, figure 2 illustrates the potential complexity of a state change context.

Figure 2: A Complex business rule



In this particular example, a single business rule may result in different events taking place (since events are tied to a single object). For example, if the order ends up being rejected only event 2 is executed. On the other hand it is possible for an order to be accepted but some of its items held as outstanding items until sufficient stock is available. In this case, event 3 (create new order) takes place but also event 5 (create outstanding item) is executed. Also, two state changes occur in this scenario. So it can be seen that a state change context has to be able to describe all potential events that may occur and thus it requires a sufficiently rich notation to support this.

REFERENCES

Harel, D. (1988). 'On Visual Formalisms'. *Communications of the ACM, 31*(5), pp514-30.

Jacobson, I., Christerson, M., Jonsson, P., and Overgaard, G. (1992). *Object-oriented Software Engineering: A Use Case Driven Approach*. Wokingham, England: Addison-Wesley.

Loosley, C. (1992). 'Separation and Integration in the Zachman Framework'. *Data Base Newsletter, 20*(1).

McDermid, D. C. (1998). 'The Development of the Business Rules Diagram', PhD thesis, Curtin University of Technology.