ACTOR PERCEPTION IN BUSINESS USE CASE MODELING

Sergio de Cesare,
Mark Lycett,
Ray J. Paul
Department of Information Systems and Computing
Brunel University
Sergio.deCesare@brunel.ac.uk

ABSTRACT

Mainstream literature recognizes the validity and effectiveness of use cases as a technique for gathering and capturing system requirements. Use cases represent the driver of various modern development methods, mainly of object-oriented extraction, such as the Unified Process. Although the adoption of use cases proliferated in the context of software systems development, they are not as extensively employed in business modeling . The concept of business use case is not a novelty, but only recently did it begin to re-circulate in the literature and in case tools.

This paper examines the issues involved in adopting business use cases for capturing the functionality of an organization and proposes guidelines for their identification, packaging, and mapping to system use cases. The proposed guidelines are based on the principle of actor perception described in the paper. The application of this principle is exemplified with a worked example aimed at demonstrating the utility of the proposed guidelines and at clarifying the application of the principle of actor perception. The worked example is based on a series of workshops run at a major UK financial institution.

Keywords: actor perception, business use cases, modeling

I. INTRODUCTION

Mainstream literature recognizes the validity and effectiveness of use cases as a technique for gathering and capturing system requirements [Cockburn, 2001]. Use case modeling is a requirements engineering technique aimed at understanding the functional specifications of the modeled system from the perspective of the parties (or actors) interacting with it. A use case, as originally defined by Jacobson [Jacobson et al., 1995], 'is a sequence of transactions in a system whose task is to yield a result of measurable value to an individual actor of the system'. This definition was criticized for its vagueness [Graham, 1996] and led to the adoption of different versions of use case modeling by most organizations. Consequently, the understanding, application and representation of use cases varied greatly across companies and development environments [Firesmith, 1999]. The lack of consistent guidelines in use case modeling also contributed to its misuse or misinterpretation [Lilly, 1999].

Use cases are predominantly employed in software development and to a lesser extent in business modeling. The issues concerning use cases at a software systems level are echoed for business use cases. Therefore, problems concerning the ambiguity of definition, usage, and consistency not only remain, but are accentuated given the specific characteristics of business modeling, which involves both business and technical people with different mindsets and terminologies. The adoption of use cases for business modeling strengthens the need for a consistent view of what use cases represent and how they should be modeled. Such a consistent view would allow greater understandability and communicability of the business model amongst the different stakeholders of the business and of the information systems developed. To adopt use cases for business modeling, guidelines and techniques need to be defined.

The view and guidelines proposed in this paper derive from an analysis of the definition of use case. The Unified Modeling Language (UML) [Booch et al., 1999] seems to reinstate Jacobson's definition, but with an interesting variation. In UML 1.1, a use case is defined as

'*a description of a set of sequence of actions, including variants, that a system performs that yields an observable result of value to a particular actor'.*

The most significant difference lies in the term 'observable' rather than 'measurable'. Subsequent versions of the UML, including the current 1.5 version [OMG, 2003], reformulate the definition, but substantially confirm the observable nature of a use case. Hence, a use case must be observable by an actor. The only type of system functionality definable in terms of a use case is functionality that an actor perceives and thus is aware of. This 'perception' is the basis of the principle and the guidelines defined in this paper for business use case modeling.

The main focus of this paper is on business use cases and the problems related with their identification, definition, and mapping to system use cases. A behavioral decomposition approach is proposed for the identification of business use cases. Use case packages are the means to achieve behavioral decomposition. This decomposition serves two purposes:

- It allows both the modeler and the business stakeholders to understand and define the area of study according to groups of logically related functionalities.
- It provides an initial structure to the business architecture.

The paper also aims at providing guidelines to enable the mapping between business and system use cases. Actor perception is the principle underlying the guidelines proposed for these problems.

The paper is structured as follows. Section II briefly defines business modeling, outlining its underlying principles and issues. It then relates these general issues with the more specific modelling technique of business use cases. Section III presents the proposed business modeling approach based on use cases, use case packages and actor perception. Guidelines are defined to fill the current gap existing in the area of business use case modeling. Section IV exemplifies the approach with a worked example based on banking account services. The example is the result of a series of workshops held with a major UK bank aimed at clarifying the application of use cases as a business modelling

technique. Implications for theory and practice are drawn in Section V and conclusions are presented in Section VI.

II. BUSINESS USE CASE MODELING

A GENERAL OVERVIEW OF BUSINESS MODELING

Business modeling is the representation of the structure and the behavior of a business organization for the purpose of understanding the business itself. The structure of a business is defined in terms of its entities and the relationships amongst them; business behavior is defined in terms of processes, events and rules essential for the fulfillment of the organization's objectives. Business modeling approaches must therefore provide techniques for defining elements essential to both the structure and the behavior of the organization.

Most business modeling approaches place emphasis on the dynamic aspects of the business. The business can be viewed as a provider of *services*. Service is an elusive concept that can be defined in numerous ways (e.g., [Johns, 1999]). In the context of this paper, a service is defined as an act or performance provided by one party to another [Lovelock and Vandermerwe, 1996] and is achieved through the execution of business processes. Business processes are initiated in response to an event (e.g., customer request). A business process is defined in terms of process elements whose combined behavior enables providing a specific service. Parties external to the organizational area of study (e.g., people, other companies, other internal organizational units, and governmental bodies) are the beneficiaries of these services; hence the understanding of the business is necessarily integrated with the definition of those parties external to the organizational area of study and interacting with it.

A service-oriented model of an organization is applicable even to businesses whose main purpose is the production and sale of goods. The traditional division between goods and services is long outdated [Gummesson, [Gummesson, 1994]. Consumers buy an offering whose value may consist of many components, some of them being activities and some things. For example,

when purchasing a good what is being offered in reality is not the good itself, but the property of the good. In a way the business provides the service of transferring the property of a good when making a sale. Consequently the sale of a product also requires the delivery of a service.

The study of business processes is a useful means for identifying and defining entities or resources of the business. Processes use, manipulate and/or transform these entities. Hence, the definition of business behavior is integrated with the identification of business entities. Moreover, the analysis of business processes also allows the modeler to define the business architecture by grouping and relating functionality with similar scope. Business process models can represent the organization as it currently behaves (descriptive 'as-is') or as it could behave if changes in the business processes are required (prescriptive 'to-be'). Whilst the forms of model are complimentary, the prescriptive view is instrumental to strategies such as business process reengineering (BPR) [Hammer and Champy, 1993] and improvement (BPI) [Davenport, 1993].

Many techniques are applied to business process modeling, each technique focusing on a specific aspect or set of aspects of the business to model. Kettlinger, Teng et al. [Kettlinger et al., 1997], in a study on methodologies, techniques and tools for BPR, identify several techniques, most of which (e.g., flowcharting and data flow diagramming) derive from the software modeling domain. The applicability of software techniques for business modeling is questionable given that they were not n developed in light of the specific needs, issues, concepts, and semantics of business organizations. To better comprehend the characteristic features that a business modeling technique should possess, it is useful to clarify the purposes of business modeling.

Business modeling is aimed at defining and representing a social system (i.e., business organization). More specifically, business modeling can serve the following purposes [Penker and Eriksson, 2000]:

- To improve understanding of the key elements of an existing business, its dynamics. and underlying structure.

- To act as the basis for creating suitable information systems that support the business.
- To act as the basis for improving the current business structure and operation by identifying problem areas and improvement potentials.
- To show the structure of an innovated business.
- To experiment with a new business concept or to copy or study a concept used by a competitive company.
- To identify outsourcing opportunities.

The representation of the organization, for any of the purposes listed, involves communication with and participation of the business stakeholders. Communication and participation are essential to obtain an acceptable understanding of the organization's behavior and structure. The product of this communication should be documented in a way that allows the business stakeholders to understand the business model clearly. In turn, comprehensibility and clarity of the model increase active stakeholder participation. A business model that provides a fair and accurate representation of the organizational area of study provides developers with a point of reference to use across the whole development process. Business use cases can be applied as a means for achieving such objectives.

BUSINESS USE CASES

Use case modeling represents a technique that drives most present-day object-oriented development methods. In the Unified Process [Jacobson et al., 1999] use cases are employed for both business and systems modeling. The route through the former to the latter is through collaboration diagrams. Select Perspective [Allen and Frost, 1998, Apperly et al., 2003], on the other hand, is an example of an object-oriented method in which use cases are employed only for system modeling. Business modeling is carried out with diagramming techniques (hierarchy diagrams and process thread diagrams) not directly related to business use cases, but mapped to system use cases in a subsequent phase. The application of use cases to business modeling, i.e. business use cases, is still immature. Although the adoption of use cases proliferated in the context of

software systems development, their implementation in business modeling is not as extensive. The concept of business use case is not a novelty [Jacobson et al., 1995], but only recently did it begin to re-circulate in the literature [Jacobson et al., 1999] and in case tools (e.g., Rational Rose).

A business use case is the description of functionality that provides a service to an actor, with the functionality described in terms of a business process. A business use case also defines other properties such as triggering event, pre and post conditions, and stakeholders. In business use case modeling, the modeled system relates to the organization or one of its sub-units. As a consequence the actors are external to the organizational area of study. Examples of business actors are customers, suppliers, and other organizational units. Conversely, internal workers (e.g., employees of the business) lie within the system boundary and therefore cannot be defined as actors in this instance. Workers would typically be considered actors in system use cases.

Business actors are normally parties identifiable as either persons or groups of persons (e.g. a company). In some cases it may appear that the actor of a business use case is not a human; for example, when a bank's computer system automatically requests a credit check to a credit scoring company. However, the bank's computer system is acting on behalf of the bank. In a non-automated system an employee could forward the credit check request. In either case, for the credit scoring company, the bank (and not the bank's computer system or employee) is the party with whom the business interaction is taking place. In both cases the bank is always the actor of the hypothetical 'Request credit score' business use case. At a system level it may well be necessary to define the bank's computer system as a system actor.

The description of the business process is mainly textual, but can be combined with graphical forms of representation. This combination of representations allows the modeler to approach the definition of business functionality through a gradual transition from a less structured/formalized representation to a more structured/formalized one. One of the key issues in gathering requirements is adopting a form of documentation that is clearly

understood by the business stakeholders. Natural language is normally the means for expressing requirements at an early stage. However, since natural language lends itself to ambiguities and inconsistencies (not making it ideal for the purposes of software developers), refinement in other forms is recommended; for example, more structured and/or graphical representations can be used to refine the use case's textual description. It is now common to utilize activity or interaction diagrams for this purpose. State diagrams can also be employed when the use case involves the manipulation/transformation of one type of object. However, graphical representations need to be kept as simple as possible to provide the business users with a clear understanding of the model. These different forms of representation constitute different and alternative ways of representing a use case's textual description. They form an integral part of the use case. From this perspective a use case can be viewed as the fundamental package of behavior encapsulating all diagrams intended to describe its functionality in terms of 'what' (service) is provided to the actor and 'how' the service is realized (process).

Hence, business use case modeling serves the following purposes:

- To capture the functional requirements of an organization or an organizational unit.
- To facilitate communication amongst business stakeholders and modelers.
- To lay down the foundations of the business architecture.
- To allow for a gradual and preferably seamless transition toward the information system model.

III. GUIDELINES FOR BUSINESS USE CASE MODELING BASED ON ACTOR PERCEPTION

In a business modeling and software development environment, the effectiveness of use case modeling for the elicitation of business requirements requires at least two conditions to be satisfied.

- A consistent view amongst business stakeholders and developers on what business use cases represent and how they are to be employed.
- All parties must adopt common guidelines for the documentation of business use cases in order to guarantee consistency across the organization.

Guidelines for use case modeling can be categorized as follows [Anda et al., 2001]:

- **Minor (or identification) guidelines**: Guidelines describing how to identify actors and use cases. Minor guidelines generally provide limited guidance on how to represent the use cases themselves.
- **Template guidelines**: Guidelines defining the structure of a use case in terms of its properties. Typical use case properties are listed in Table 1.
- **Style guidelines**: Guidelines on how to structure the flow of the use case. Style guidelines refer to the textual description of the underlying process. Different recommendations are suggested by the literature and summarized by Anda et al. [Anda et al., 2001] and Cockburn [Cockburn, 2001].

The guidelines proposed in this paper fit into the above three categories and build upon those commonly accepted in the literature and by practitioners. The driving principle of these guidelines is actor perception. Actor perception facilitates the identification of use cases and is employed in the following subsection to define a use case template based on the distinction between the service perceived by the actor and the process to deliver it. Subsequently guidelines for grouping business use cases are defined as a means to architect the business. Finally, a technique for mapping business use cases to system use cases is presented.

STRUCTURE OF A BUSINESS USE CASE

Of particular importance for business use cases is that they are predominantly textual in nature. In business modeling, models are both about people and for people [Ould, 1995]. During the elicitation of business

requirements, the business analyst needs to discuss, correct and improve the model with the business people. Text is a form of representation, which facilitates interaction and communication with the business representatives since it requires no special training for it to be understood. The textual nature of business use cases allows business people to capture the essence of the technique fairly easily, enabling them to become active modelers. In such a situation, the business analyst would primarily assume roles of coordinator and moderator.

Business use cases capture a narrative told by the business representatives about the way their organization or organizational unit delivers services. The description of the underlying business process follows the flow of the narrative in which a dialogue between the actor and the organizational system interact as a means to achieve the ultimate end of receiving and providing the business service. Narratives captured by use cases are structured textual descriptions. However, no standard structure is yet defined for use cases in general. The UML [OMG, 2003] overlooks this important aspect and concentrates on the less important matter of the graphical representation of use case diagrams [Cockburn, 2001].

Several use case templates are suggested in the literature [Anda et al., 2001, Cockburn, 2001, Jacobson et al., 1995, Rosenberg and Scott, 1999]. Each template defines a set of properties that define a use case. For reference, typical use case properties are summarized in Table1.

Table1. Properties of a Use Case

Property	Definition
Title or Name	Defines the name of the use case.
Actor(s):	Party who obtains the observable result of value of the use case, also known as the primary actor. An actor can be a person or another system. A use case can have supporting actors, i.e., other parties who contribute toward the execution of the process defined by the use case for the ultimate delivery of the service.
Trigger	Event that initiates the process defined by the use case
Scope	Corresponds to the boundary of the system under study, e.g. business, software system.
Preconditions	Conditions that must be satisfied for the use case to take place.
Basic flow	Description of the flow of activities that ordinarily take place for the execution of the process defined in the use case.
Extension points	References to other use cases extending the normal process flow. Extension points are generally referred to in the description of alternate courses.
Alternate courses	Courses defining alternative paths of execution of the process defined in the use case.
Post-conditions	Conditions that must hold true after the termination of the process.

Source: [Anda et al., 2001]

Most of the properties in Table 1 provide a fairly comprehensive description of what defines a process. This type of template, however, is limited when adopting a service-oriented approach to business modeling. From the actor's perspective, services represent the observable or visible part of a use case; hence the principle of actor perception is tightly associated with the concept of service provision. Actor perception refers to the actor's awareness of the existence of specific system (e.g., business organization) behavior from which the actor expects a finite number of possible predefined outcomes. The actor knows about the service in terms of what it is and what can be achieved from it. The actor does not require detailed knowledge of the delivery process. In some cases, however, some aspects of the process may be transparent to the actor.

Transparency occurs, for example, when the actor takes part in the process (e.g., therapy services) or when the Quality of Service (QoS) is measured at specific stages of the process.

Consequentially, a business use case can be defined as consisting of two main sections:

- **Business service section**: Defines the properties of the business service provided to the actor.

- **Business process section**: Defines the properties related to the activation and execution of the business process.

The proposed template of a service-oriented business use case is illustrated in Table 2. The template is divided into three sections:

- The whole of the business use case and is dedicated to its name and primary actor.
- The properties of the business service provided to the primary actor. These properties are drawn from the business service literature [Hart, 1988] and fundamentally relate to the guarantees that the service provider obliges (or is obliged) to satisfy in favor of the primary actor.
- The business process delivering the service. It includes all the elements necessary for the initiation, execution, and termination of the process.

PACKAGING BUSINESS USE CASES

Logically related business use cases can be grouped together to form business use case packages. The grouping of business use cases is based on a common packaging rationale that takes into account the characteristics of actors, services, and their relationships. Packaging serves two fundamental purposes:

- Packages are defined according to a common underlying theme. This common theme can be used as a basis for discussion during workshop sessions with the business stakeholders to identify further services and processes. It can be used as a means to structure discussion and reflection.
- Business use case packages are architectural elements, which allow for the initial definition and representation of the business architecture. The business architecture is an essential part of the business model, which serves as a conduit toward the translation into the model of the software system.

Table 2 – Template for a Service-Oriented Business Use Case

Business Use Case Name	
Primary Actor	Recipient of the service.
Business Service	
Service promise	Description of the outcome that the actor can expect. The value of the service is strongly dependent on the service outcome.
Necessary Conditions	The conditions that must hold true for the provider to offer the service to the requesting actor.
Quality of Service Standards (QoS)	Set of constraints that define measurable characteristics of the delivered service.
Payout	Any obligations that must be carried out by the service provider whenever the QoS is not met.
Business Process	
Supporting Actors	Parties involved in the business process and whose presence is necessary for delivering the service.
Pre-conditions	Conditions that must be satisfied for the use case to take place.
Trigger	Initiating event of the business process.
Description (or Basic Course)	Description of the flow of activities that ordinarily take place for the execution of the process defined in the use case.
Alternate Courses	Description of alternate courses of execution of the process.
Post-conditions	Conditions that must hold true after the termination of the process.

Business use case packages are, therefore, a way to structure human interaction and thought, as well as the business model itself. Architecture is a means of achieving these goals. It is defined as the structure of components of a system, their interrelationships, and the principles and guidelines governing their design and evolution over time [Garlan and Perry, 1995]. In the area of business modeling, however, the concept of business architecture is not consistently defined throughout the literature. The problematic definition of business architecture may be due to the contrasting nature of the terms ‘business’ and ‘architecture’. Business refers to the pre-existing area of study or the problem domain, whereas architecture normally refers to the structure given to a proposed or developed solution, e.g. the architecture of a bridge or software architecture. Both the bridge and software are solutions to a need representing

the problem. This duality between problem and solution space residing within the same concept can be clarified by understanding the purpose of business architecture for information systems development.

When modeling a business organization, the business architecture assumes a primary role in preparing the terrain for the transition toward the subsequent software model, including the software architecture. The business architecture is, therefore, that part of the business model that gives form to the organizational domain, shaping the problem in a way that it can be more readily comprehended by software analysts and designers. The business architecture pulls and holds together the key components of the business system. These key components subsequently drive the representation of the software models.

The way architectures are defined and how their constituent parts are connected is dependent on the approach that the modeler adopts. For example, architectures can be defined via objects, components, agents, patterns or a coherent mix of these various, yet similar, approaches. Architectures can be led behaviorally. This means that the key architectural components are derived from the behavior of the modeled system. Behaviorally led approaches to defining business architectures are more consistent with the dynamic nature of business organizations. Organizations are, of course, societal systems in which the complexities of human and/or human/machine interaction determine the overall and emergent behavior of the business. Business use case packaging can be considered as a behaviorally led approach to representing business architectures. Analyzing business behavior via use cases, in terms of services and processes, highlights both the complex interactions occurring between the business and the external world and the dynamics of the processes delivering the services requested. Packaging business use cases, ultimately, gives structure to the representation of behavior.

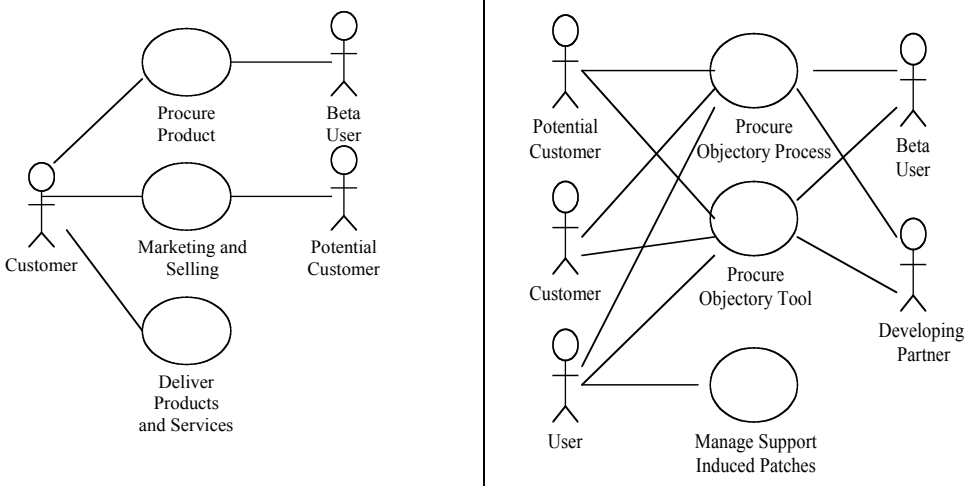
In use case modeling, no concepts for modularization are given to manage large use case models [Regnell et al., 1996]. As a consequence, loose collections of use cases are defined as separate and partial models, addressing narrow aspects of the system requirements [Regnell et al., 1995]. Given the

complexity of business organizations, the definition of cohesive groups of logically related use cases is essential for business modeling. Closely related is the problem of use case granularity in terms of scope of a use case. Jacobson [Jacobson et al., 1995] indirectly takes these problems into account and describes how use case models can be represented at different levels of abstraction to satisfy the perspectives and interests of different 'handlers'. The first level is an overview model addressed to the organization's executive management (Figure 1a). The second level model is instead intended for the 'process handlers', i.e. those stakeholders more closely related with the everyday functioning of the business processes (Figure 1b). The use cases of Figure 1b can be considered as 'packaged' inside the corresponding higher-level use cases of Figure 1a.

Use case packaging is introduced in the Unified Modeling Language. The UML 1.5 [OMG, 2003] defines three use case stereotypes:

- **Use case system**: A use case system is a top-level package that may contain use case packages, use cases, and relationships.
- **Use case model**: A use case model specifies the services a system provides to its users, i.e., the different ways of using the system, and whose top-level package is a use case system.
- **Use case package**: A use case package contains use cases and relationships. A use case is not partitioned over several use case packages.

Use case packaging enforces the simplicity, understandability and communicability of the model. With use case packages, focus can be streamlined into a group of logically related functionalities of the modeled system. This approach allows for more meaningful and self-contained representations. Jacobson in his original work [Jacobson et al., 1995] does not use the term



a. The Objectory AB Use-Case Model Overview	b. The Procure Product Detailed Use Case Model

Source: [Jacobson et al., 1995], p. 321-322

Figure 1. Use Case Models

package to identify groups of use cases, but refers to them as first level use cases. Use case package is a more expressive concept that intrinsically communicates sense of grouping. However, as an interpretation of Jacobson's 'Object Advantage' shows, such packages are use cases in their own right. Hence, use case packages should be described with a list of properties just as (lower-level) use cases are. However, given that the level of granularity is different, the properties defining a business use case package are different than those utilized to define business use cases. The difference lies in the scope and purpose.

The scope of a business use case is a specific service expected by an actor. Hence, a business use case is defined in terms of a service and a process delivering the service. A business use case package is defined by several logically related services whose individual specific properties are detailed in their corresponding business use cases. The list of services provides the main description of a business use case package. No temporal sequence between the services can be implied from this list; the primary actor(s) can request any service at any time as long as the pre-conditions of the related business use case are met. Table 3 defines the properties of a business use case package.

Table 3. Properties of a Business Use Case Package

Property	Definition
Name	Designates the name identifying the package.
Packaging rationale	Reason for grouping the services together.
Actor(s):	Persons or systems that can request one of the services provided by the package and benefiting from it.
Services provided	Name and purpose of all services defined within the package

The purpose of a business use case package is architectural. The package pulls together various use cases around a common theme. One of the fundamental characteristics of a good business modeling technique is understandability by the business stakeholders whose vocabulary and semantics do not include software development terms such as architecture. This consideration raises the question of whether business stakeholders should be exposed to the concept of business use case package and whether the concept should be employed with them during the identification of business use cases. This question should be answered affirmatively. It is true that architecture is a term typically applied in the realm of engineering, however architectural techniques are tools for the organization of thoughts as much as they are for structuring systems. Since the organization of thoughts is the basis of any modeling endeavour, then business use case packaging should be used at the forefront of business modeling with the business stakeholders. Thus, grouping mechanisms help 'architect' both mental models and business and software models.

In business modeling, the relationship between use case packages and use cases is that of decomposition. A use case package can be decomposed into other packages or ultimately into use cases. Although decomposition usually does not go beyond two levels of representation as with Jacobson's example (i.e., use case packages containing use cases), use case packaging can, in theory, allow for multi-level hierarchies.

The main problem with such an approach is being able to understand where to terminate in the process of decomposition. Sometimes modelers may not be aware that they reached the level of a business use case and risk

decomposing further. This problem can be resolved by applying the principle of actor perception. Since a use case must be visible to an actor, decomposition terminates when the business use case is described in terms of activities that are internal to the organization and therefore not externally visible to any actor. Use case packages, on the other hand, are described as a set of related services deliverable to actors. Each one of these services is externally perceived by an actor.

The organization of business use cases into packages facilitates the representation of the business architecture. Business use case packages represent the foundation of the business architecture, which would need completion in terms of dependencies and interfaces amongst packages and their internal static representations. All these enhancements are added on top of the model constructed with the business stakeholders. This refined model is more technical and developed outside of the arena of discussion with the business stakeholders. It is beyond the scope of this paper to delve into the refinements that the business model undergoes. However, it is sufficient to state that as part of a gradual and possibly seamless transition between the business and software models, various equivalent business models are to be produced before transiting into the software-modeling domain.

MAPPING BUSINESS USE CASES TO SYSTEM USE CASES

Information systems play a fundamental role in fortifying business competitiveness. The information counterpart of ‘real’ business behavior is nowadays generally modeled within software-based information systems. The underlying models of such systems require continual alignment with the business model. Unlike business organizations, which are living systems, software systems are developed systems. This distinction implies that the living nature of a business inevitably changes at a much faster pace than that of developed software systems. Consequently software models are merely snapshots [Lycett and Paul, 1999] in time of the corresponding business system (or subsystem). To minimize the lead-time between business change and software amendments, methods, techniques and/or guidelines for mapping elements of the business

model to those of the software model should be defined and introduced into the development process.

Deriving system use cases from business use cases is, therefore, part of a more generalized problem regarding the alignment of the information system to the business model. Transition from the business model(s) to software (analysis and design) models, and their mapping to implemented software components, involves semantic, human, and technical aspects. Semantically speaking, business stakeholders and software developers describe the world with different ontologies. Their interpretation of the same problem is different and contextualized in accordance with the purpose and domain of personal reference. No rigorous techniques currently exist to overcome these difficult problems. Without investigating in depth the reasons underlying such problems, a few general criteria can be suggested to alleviate them:

1. Participation of Stakeholders

The development of an information system requires the continual participation of business and development stakeholders in an integrated effort of collaboration. The participation of the various stakeholders is required to manifest the different perspectives and diverse semantics.

2. Iterative and Incremental Development

Because of the multiple views of stakeholders and the evolutionary nature of business organizations, iteration is necessary and the translation from the business elements to the system elements should be carried out as an ongoing process throughout development. In the development of information systems, the business model should be gradually translated into a model of the computer system. Preferably such a translation should be as seamless as possible. Iterative and incremental development facilitates this transition. Therefore, passage from the business model to the system model should not be carried out in purely sequential phases.

3. Consistency of Approach and Modeling Language

A way to preserve seamless transition is to adopt the same underlying philosophy in both business and system modeling. Utilizing the same approach and modeling language for capturing and representing both the business and the system requirements reduces the semantic inconsistencies between techniques and notation. Hence, the adoption of use cases for business modelling should be coupled with their use in driving system development as well.

Before deriving system use cases from business use cases, the modelers and the stakeholders must decide which activities should be automated or supported by the resulting system. Many factors can influence such a decision (e.g., strategic or tactical objectives, cost/quality implications). Once this decision is taken, the next step is to derive system use cases from the business model. No rigorous approach exists to mapping business use cases to system use cases.

In the Rational Unified Process (RUP) the mapping between business and system use cases is carried out through the analysis of collaboration diagrams. In RUP the textual description of a business use case is combined with the graphical representation of a collaboration diagram. A collaboration diagram represents the interactions between objects of the business system. In a business collaboration model some of the objects represented are business workers. System use cases are defined around business workers. Business workers are defined as system actors and the system use cases reflect the task they carry out in the business use case realization. These tasks are defined as system use cases only if a decision was taken to automate them. The technique proposed by RUP is dependent on the adoption of an object-oriented method and is embedded in RUP itself. It does not easily fit into methods based on other development approaches.

The technique proposed in this paper is method independent. It derives system use cases from the activity diagrams employed to represent the process underlying the business use case. As stated above, a business use case

contains a textual and a graphical description of the business process. The graphical description usually assumes the form of an activity diagram in which roles, activities, events and results are represented. When deriving system use cases, only those activities that will be automated or supported by the information system should be considered. Each activity should be taken as a candidate system use case. Since a use case provides an actor with an externally visible result, each activity should be taken individually and the result produced by the activity should be analyzed. If the result is visible to anybody or anything lying outside the boundary of the computer system than that activity most likely represents a system use case and the individual or system benefiting from the result represents an actor. If the activity does not represent a use case then it should be grouped with other adjacent activities. The analysis is then applied to this group of activities.

A variation of the above technique can also be applied. The modeler can initially identify the actors of the system and group the activities according to the actors' expectations of what the system can deliver. In this variation the actors are identified first.

Sometimes an entire business use case can be mapped to a system use case. In these situations one should keep in mind that:

- The business and system use cases are not the same use case.
- A business use case serves a business actor (e.g., customer, supplier), whereas a system use case serves the computer system user (e.g., clerk).
- The system use case is described in terms of interaction between the computer system and the user, whereas the business use case is described in terms of business interaction (e.g., negotiation, agreement, contractual obligations)
- Business use cases should be kept as simple as possible; therefore relationships between use cases (such as extend and include) should be avoided.

- System use case modeling is much more detailed than business use case modeling. The descriptions contained in system use cases form the basis for the design and implementation of the computer system. Reuse should be taken into consideration and, as a consequence, extend and include relationships should be modeled.

The proposed technique has a much wider range of application than the RUP mapping technique. In fact, it is based on the use of activity diagrams, which in various forms, are used by a wide range of methodsbased on diverse paradigms. This potentially allows to extend the utilization of use cases to non object-oriented techniques during business modeling.

IV. WORKED EXAMPLE

The worked example presented in this section is aimed at demonstrating the utility of the guidelines presented throughout the paper and at clarifying the application of the principle of actor perception for the identification and packaging of business use cases, and their mapping to system use cases.

The example is based on account services offered by a typical bank. The models presented in this section were produced during a series of workshops on business use case modeling with a major UK bank. The scope of the workshops was to present the modeling technique highlighting its benefits and limitations. The models presented in this section are not meant to be a complete representation of account services, but sufficient to illustrate the applicability and utility of the guidelines based on actor perception and service-orientation.

The business area modeled is ‘Banking Account Services’. Architecturally it represents the business use case system, i.e. the highest level of the model within which business use case packages are to be defined. In an initial brainstorming exercise the modelers (along with the stakeholders) must adopt the perspective of the focal actor of the business system (i.e. the main party who benefits from the services provided) and understand what type of services the focal actor would benefit from. In this example, it is fairly simple to assume that the services offered by this banking area are provided to the customer. In fact, it

is the customer who requires an account in order to carry out different types of transactions and operations. An initial brainstorming session on what the customer expects from the bank may produce a list similar to the following:

- Apply for an account
- Close an account
- Carry out financial transactions (e.g., deposit, withdraw, transfer money)
- Make amendments to personal details (e.g., change address, change PIN)
- Order stationary (e.g., check books, paying-in books, reference letters)
- Request account information (e.g., statement)

According to the principle of actor perception this list only includes services or groups of services observable by the customer. From this initial list, possible groupings of services can be identified. These groupings are represented as business use case packages. Four packages are identified:

- Administer account,
- Manage Customer Profile,
- Manage Money and
- Request Account Information and Documents.

These four areas can serve as a theme for discussion in order to identify other business services. Table 4 defines the properties of these packages and their related services.

The services provided by the business use case packages can be defined as business use cases. In fact, any further behavioral decomposition would lead to activities no longer observable (or perceptible) by the customer. The Apply for Account business use case is defined in Table 5. This example highlights the different sections of the business use case and the distinction between delivered/expected service and underlying business process. Figure 2 refines the textual description of Apply for Account into an activity diagram.

Table 4 – Business Use Case Packages of the Banking Account System

BUC System: Banking Account Services	
BUC Package: Administer Account	
Packaging rationale	This package comprises all services that concern the account as a whole.
Actor	Customer
Services provided	Apply for account Close account
BUC Package: Manage Customer Profile	
Packaging rationale	This package comprises all services managing individual properties of the account or individual aspects of it.
Actor	Customer
Services provided	Change contact details Change security details Request overdraft limit increase Request replacement card Dispute account transaction
BUC Package: Manage Money	
Packaging rationale	This package comprises all services managing financial transactions.
Actor	Customer
Services provided	Deposit money Withdraw money Pay bills Create standing order Cancel standing order Transfer money Create direct debit Cancel direct debit
BUC Package: Request Account Information and Documents	
Packaging rationale	This package comprises all services that allow the customer to receive information or documents related to the account.
Actor	Customer
Services provided	Request statement Request mini-statement Order Stationary Request reference letter

Given its simplicity, the diagram has been developed with the business stakeholders. More refined and structured diagrams can be developed, if necessary, by the modeler, once the requirements of the business area of study are well defined. This simplified example shows different levels of refinement of a business model.

Table 5. Business Use Case: Apply for Account

Business Use Case: Apply for Account	
Primary Actor	Customer
Business Service	
Service promise	To open an account for the applicant if the applicant's credit check is successful and, in any case, inform the applicant of the outcome of the application.
Necessary Conditions	Applicant must be 18 years of age or older and reside in the European Union.
Quality of Service Standards	The applicant is entitled to know about the status of the application at any time and to receive a response after 5 days at the latest after reception of the application.
Payout	The applicant is entitled to a free crate of wine if the bank does not communicate the outcome of the application within 5 days after receiving the application.
Business Process	
Supporting Actors	Clerk
Pre-conditions	None
Trigger	Customer request
Description (or Basic Course)	Following the customer's request to open a bank account, the bank clerk collects the customer's details and those of the requested account. The customer is given information related to when and how he/she will receive a response of approval or rejection from the bank. The clerk submits application form with valid details to the credit-checking department for validation. The credit-checking department proceeds with the validation of the application and informs the accounts department of the outcome. If validation is ok the account is created otherwise the request is rejected. The customer is informed of the outcome and provided with all necessary information.
Alternate Courses	None
Post-conditions	Creation of new account. Customer informed. (Main success scenario) or Customer informed of rejected application

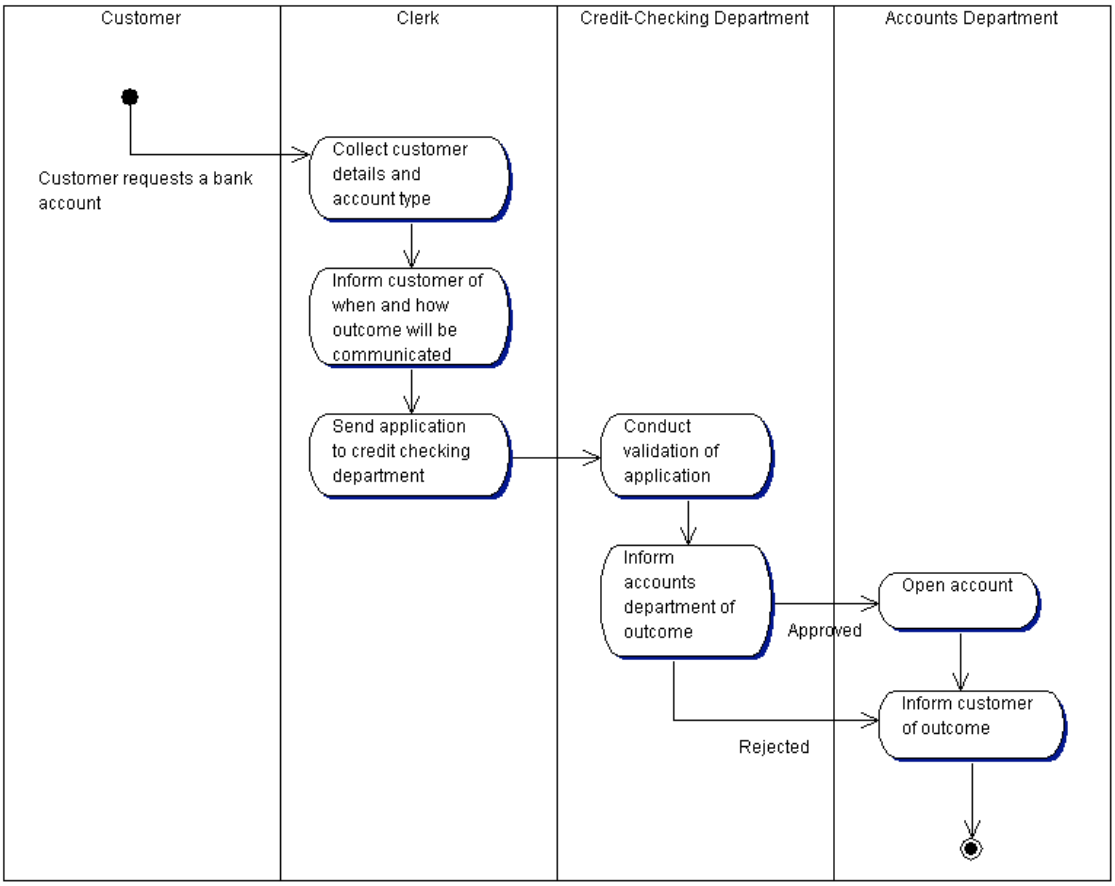


Figure 2. Activity Diagram of Apply for Account

The principle of actor perception can be applied to the individual activities of Figure 2 to identify possible system use cases. Before the identification of system use cases, a decision must be made in terms of which activities shall be automated. It is assumed that the business stakeholders decide to automate all activities except for 'Inform customer of when and how outcome will be communicated' performed by the clerk. This activity is to be performed vis-à-vis with the customer. Actor perception is applied to each of the remaining activities so as to determine candidate system use cases. Perception of these activities is defined in terms of the corresponding actor represented by the swimlane.

As an exemplification, the clerk actor is considered. The clerk is responsible for two automated activities: 'Collect customer details and account type' and 'Send application to credit-checking department'. The clerk's perception of these two activities when using the computer system is that they represent one

business process aimed at enabling the clerk to satisfy the customer's request. Although the clerk may be aware of the existence of the two separate activities (for example because of the messages shown by the system's interface), his or her perception is that of a unitary process. The computer system provides the clerk a complete service only if it provides the means for collecting details *and* sending them off for validation. Thus, one system use case can be defined: Process Application Form. The same process can be applied for the remaining activities. Table 7 illustrates how the activities of the Apply for Account business use case map to different system use cases.

Table 6. Mapping Between the Business Use Case and Potential System Use Cases

Business Use Case: Apply for Account			
Actors	**Activities**	**Automated**	**Possible System Use Cases**
Clerk	Collect customer details and account type	Yes	Process Application Form
	Inform customer of when and how outcome will be communicated	No	
	Send application to credit-checking department	Yes	Process Application Form
Credit-Checking Department	Conduct validation of application	Yes	Conduct Credit Check
	Inform accounts departments of outcome	Yes	Conduct Credit Check
Accounts Department	Open account	Yes	Create Account
	Inform customer of outcome	Yes	Create Account

V. DISCUSSION

IMPLICATIONS FOR THEORY AND RESEARCH

The theoretical contribution of this work is to propose an approach to business use case modeling based on the principle of actor perception. The theoretical relevance of this approach is twofold:

- It builds on pre-existing definitions and principles. Actor perception is an emphasized reaffirmation of the fundamental characteristic of use case 'observability'. The approach itself utilizes principles of behavioral

decomposition and sound architecture for the creation of the business model. The combination of these principles to use case modeling allowed defining an approach which is both theoretically sound and of practical value.

- The proposed approach introduces the concept of service in business use case modeling. Services are currently being applied at a technological level. The novelty of this research is to introduce services as a primary modeling concept in business modeling. Thus far, business modeling is dominated by data-driven and process-driven methods and techniques. A service-oriented approach builds on these previous techniques, especially in the case of process modeling. The integration of service and process to model behavior has been defined and demonstrated in the paper.

IMPLICATIONS FOR PRACTICE

From a practical perspective the proposed approach fills a gap concerning the identification, definition, packaging and mapping of business use cases. As stated in Section I, misuse and misinterpretation of use cases is not uncommon in companies. Workshops conducted within a major UK bank reconfirmed these problems. Without practical guidelines, business use cases, when utilized, tend to be applied in a non-consistent way throughout the organization and later on in the process become devoid from the development process lacking traceability between the business and the software models. The proposed approach enforces consistency and traceability. Furthermore, actor perception and the service view that derives from it make the approach more coherent to business stakeholders' perspective of organization as an entity that is expected to provide services delivered by processes in which roles and responsibilities are assigned.

VI. CONCLUSION

Use case modeling is a technique aimed at collecting and specifying system requirements from the point of view of the system users or actors. Originally defined by Jacobson [Jacobson et al., 1992], use cases have been subject of much debate related to their definition and usage. Issues concerning the ambiguities and inconsistencies surrounding use case modelling are

documented by the literature. Guidelines to overcome these problems were suggested, but the effectiveness of proposed techniques sometimes is less than desirable. In business use case modeling, these issues impact the modeling effort more given the diverse background of the people. The business stakeholders come from an organizational culture, not a technical one. Modelers tend to possess a more technical mindset. The latter must accommodate in order to relate better to the organizational way of thinking. Business use cases, in a way, reconcile these two worlds.

Business use cases are mainly textual descriptions of business services and processes, and they are based on the perspective of actors benefiting from the services offered by the organization or organizational unit under study. These two characteristics make the technique closer to the way business people represent (by text and natural language) and perceive (agents supplying and demanding services) the world. Use cases are also a technique deployed in software development for several years. Business modelers with a technical background are able to adopt a technique that is strongly accepted in software modeling, and is based on an underlying philosophy which reflects the business way of thinking. However, to adopt business use cases effectively, guidelines on how to identify, define and represent them are needed.

The guidelines proposed in this paper are based on the principle of actor perception. This principle derives from the observable nature of a business use case, i.e. observable to the actor interacting with the business system. Perception or observability is closely related to the concept of business service. An actor expects a service from the business system. The service is the observable and visible part of a business use case and is always known to the actor. The process, or the way the service is delivered, is not always visible to the actor. As a consequence, a dual business use case structure is proposed. One section is dedicated to the definition of service properties and a second section is dedicated to the definition of the business process. These representational guidelines are complemented by process or ‘how to’ guidelines

concerning the packaging of business use cases and their mapping to system use cases.

Business use case packaging groups together logically relate use cases. Packages serve the dual purpose of

- facilitating discussion around a common theme so as to streamline the attention and focus of those participating in the modeling activity and
- structuring the business model by providing an initial business architecture which will ultimately be translated into the software model.

Actor perception is applied in business use case packaging as well. Packages are defined as groups of services (represented subsequently as business use cases) that an actor perceives and is able to relate together. Business use case packaging normally involves two levels, but can go beyond that in certain cases. Packaging, in this instance, represents a form of behavioral decomposition, which terminates with the identification of business use cases.

Guidelines for the derivation of system use cases from business use cases are also proposed. The principle of actor perception is applied to activity diagrams, which are defined from the textual description of the business process. Perception or observability, in this case, is considered from the perspective of the software system actor.

- First, a decision to which activities are to be automated is made.
- Second, system actors are derived from the activity diagram's swimlanes.
- Third, system use cases are derived from individual or groups of activities defined in the actor's swimlane.

Groups of activities that the system actor perceives, as part of the achievement of the same goal, represent possible system use cases.

The above guidelines were applied in a worked example on 'Banking Account Services' defined in a series of workshops with a major UK bank

(Section IV). The worked example demonstrates the practical applicability of the guidelines.

VII. REFERENCES

Allen, P. and S. Frost (1998) *Component-Based Development for Enterprise Systems : Applying the Select Perspective*: Cambridge Univ Press.

Anda, B., D. Sjøberg, and M. Jørgensen. (2001) “Quality and Understandability of Use Case Models.” *European Conference on Object-Oriented Programming (ECOOP), Budapest, Hungary, 2001*, pp. 402-428.

Apperly, H., R. Hofman, S. Latchem, B. Maybank et al. (2003) *Service- and Component-based Development*. London: Addison Wesley.

Booch, G., J. Rumbaugh, and I. Jacobson (1999) *The Unified Modeling Language User Guide*: Addison-Wesley.

Cockburn, A. (2001) *Writing Effective Use Cases*. Upper Saddle River, New Jersey: Addison Wesley.

Davenport, T. H. (1993) *Process Innovation. Reengineering Work through Information Technology*. Boston, Massachusetts: Harvard Business School Press.

Firesmith, D. G. (1999) “Use Case Modeling Guidelines.” *Technology of Object-Oriented Languages (TOOLS 30), Los Alamitos, CA, U.S.A., 1999*.

Garlan, D. and D. Perry (1995) "Software Architecture: Editorial," *IEEE Transactions on Software Engineering* (21) 4, pp. 269-274.

Graham, I. (1996) “Some problems with use cases ... and how to avoid them.” *International Conference on Object Oriented Information Systems (OOIS), London, 1996*, pp. 18-27.

Gummesson, E. (1994) "Service management:An evaluation and the future," *Journal of Service Industry Management* (5) 1, pp. 28-36.

Hammer, M. and C. Champy (1993) *Reengineering the Corporation: A Manifesto for Business Revolution*. New York: Harper Business.

Hart, C. W. L. (1988) "The Power of Unconditional Services Guarantees," *Harvard Business Review* (66) 4, pp. 54-62.

Jacobson, I., G. Booch, and J. Rumbaugh (1999) *The Unified Software Development Process*. Upper Saddle River, New Jersey: Addison-Wesley.

Jacobson, I., M. Christerson, P. Jonsson, and G. Overgaard (1992) *Object-Oriented Software Engineering: A Use Case Driven Approach*. New York: ACM Press.

Jacobson, I., M. Ericsson, and A. Jacobson (1995) *The Object Advantage: Business Process Reengineering with Object Technology*. New York: ACM Press.

Johns, N. (1999) "What is this thing called service?," *European Journal of Marketing* (33) 9/10, pp. 958-973.

Kettlinger, W. J., J. T. C. Teng, and S. Guha (1997) "Business Process Change: A Study of Methodologies, Techniques, abd Tools," *MIS Quarterly* (21) 1, pp. 55-80.

Lilly, S. (1999) “Use Case Pitfalls: Top 10 Problems from Real Projects Using Use Cases.” *Technology of Object-Oriented Languages (TOOLS 30), Los Alamitos, CA, U.S.A., 1999*.
Lovelock, C. and S. Vandermerwe (1996) *Services Marketing*. London: Prentice Hall.
Lycett, M. and R. J. Paul (1999) "Information Systems Development: A Perspective on the Challenge of Evolutionary Complexity," *European Journal of Information Systems* (8) 2, pp. 127-135.
OMG. (2003) *OMG Unified Modeling Language Specification* formal/03-03-01, http://www.omg.org/cgi-bin/doc?formal/03-03-01.
Ould, M. (1995) *Business Process: Modelling and Analysis for Re-engineering and Improving*: Wiley.
Penker, M. and H. E. Eriksson (2000) *Business Modeling With UML: Business Patterns at Work*: John Wiley & Sons.
Regnell, B., M. Andersson, and J. Bergstrand. (1996) “A Hierarchical Use Case Model with Graphical representation.” *Proceedings of the IEEE Symposium and Workshop on Engineering of Computer Based Systems (ECBS), 1996*.
Regnell, B., K. Kimbler, and A. Wesslen. (1995) “Improving the Use Case Driven Approach to Requirements Engineering.” *Proceedings of Second International Symposium on Requirements Engineering, York, UK, 1995*.
Rosenberg, D. and K. Scott (1999) *Use Case Driven Object Modeling. A Practical Approach.*: Addison-Wesley.

VIII. ABOUT THE AUTHORS

Sergio de Cesare holds a PhD in Business Information Systems from LUISS Guido Carli in Rome. He is currently a Research Fellow at Brunel University where he is actively involved in the Semantic Integration Environment (SITE) project. His research focuses on business modeling and information systems development. He publishes in the areas of object-oriented and component-based approaches.

Mark Lycett holds a BSc in Computing and Business Management (Oxford Brookes), a MSc in Information Systems (Brunel University) and a PhD in Information Systems (Brunel University). Prior to returning to education, Mark spent a number of years in industry where he both worked on and managed a number of national and international feasibility/development projects. His research concentrates on all aspects of component-based software engineering.

Mark publishes in the area of Computer-Based Systems Engineering (CBSE) in a number of leading journals and international conferences.

Ray J. Paul holds a Chair in Simulation Modelling at Brunel University. He was educated at the University of Hull in Mathematics (BSc) and Operational Research (MSc, PhD), before being appointed by the London School of Economics, where he taught Information Systems and Operational Research for 21 years. His publications include three books and over 200 papers in journals, books and conference proceedings. He serves as an expert for a variety of U.K. government departments, software companies, and commercial companies.