Action Within Information Systems: Outline of a Requirements Engineering Method°

Göran Goldkuhl[1,2] and Pär J. Ågerfalk[3,1]

[1] Centre for studies on humans, technology and organization (CMTO).
Linköping University, Sweden.
ggo@ida.liu.se

[2] Jönköping International Business School, Sweden.

[3] Dept. of Business Administration, Computer Science, Economics and Statistics.
University of Örebro, Sweden.
pak@esa.hoe.se

Abstract. In this paper an action approach to requirements engineering is outlined. The approach is based on a language action view of information systems, organisations and business. Theoretical standpoints and methodological consequences are presented and elaborated. One conclusion is that requirements engineering ought to be viewed as actability engineering in order to perceive information systems as action. It is argued that such an understanding of the communicative aspects of information and information systems are vital to the design and use of information systems as communication tools when doing business. The presented method thus aims at building information systems to support business action, in the context of business rather than computerised system.

° Formal reference to this paper is: Goldkuhl G, Ågerfalk P J (1998). Action Within Information Systems: Outline of a Requirements Engineering Method. In *Proceedings of 4th Intl. Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'98)*, June 8-9 1998, Pisa, Italy.

1 Introduction

Information systems (IS) fail many times to meet the requirements of their users. The specification and evaluation of requirements is a crucial part of the systems life cycle. The management of requirements is a difficult task. It is hard to elicit the appropriate requirements and to formulate them in a way so they are understandable to users and formal enough for designers to build software. A requirement specification can be seen as a contract between users and designers. The specified requirements must be accepted by the users as an assignment for design. The specified requirements must also be accepted by the designers - as a proper basis for the design of software.

There are many approaches to requirements engineering. There are functional methods, as e.g. Structured Analysis - SA [40], information engineering methods, e.g. [27], and object oriented methods, as e.g. OMT [32].

All these approaches aim at modelling requirements. Advocates of prototyping, e.g. [34, 5], have challenged such kind of modelling as to abstract. It is hard for users to understand how an information system will function from abstract models (as data models or data flow models). The use of simple prototypes of software is claimed to enhance the understanding of future systems among users. Through experience of the behaviour of the future systems, users will come to an understanding of what it might be to use those systems. Prototypes are often used as an aid for designing user interfaces. The underlying principles of the system are however harder to visualise in a prototype. Abstract requirement models are important to use for description, analysis and critique of principles and features that go beyond the user interface. We think that reconciliation is needed between modelling and prototyping. In requirements engineering there is need for, on one hand, abstraction and modelling and, on the other hand, visualising and experiences.

One important aspect to learn from the prototyping - modelling debate is the need to go beyond static properties of a system. Many system models tend to be modelling only structural and static aspects of systems. Dynamic and active aspects of systems need also to be treated and modelled during requirements engineering.

This is also apparent if we look at the evolution of methods for requirements engineering. Classical methods for data modelling, like the Entity/Relationship approach [7], had a focus on fairly static aspects. Object orientation (OO) can to some extent be interpreted as a critique of such a static emphasis. In OO there is an attempt to model both static and dynamic aspects of IS, e.g. [32]. The further development of OO goes into more dynamic aspects of IS. The use case notion [24] is an attempt to capture the intended behaviour of a system from it's outside.

The historical evolution of requirements engineering show an increasingly interest in information systems as dynamic and active objects. Both prototyping and object orientation are examples of this evolution. Unfortunately these approaches, although many good features, lack a proper understanding of information systems as organisational phenomena. They fail to see the genuine social character of information systems. Such systems are important instruments for organisations to perform their action. Information systems are *systems for action*.

The purpose of this paper is to give an outline of a requirements engineering method which is based on an action view of information systems. We present some parts of a requirements engineering method and its theoretical background. Relationships to some existing methods are commented.

2 Action and Communication as Theoretical Foundation

Many approaches and theories within information systems are based on a strict representational view of information. One main purpose of requirements engineering is to get an accurate "image" of the reality in order to have this piece of reality properly represented in the (data base of) the system. This strict representational view can be challenged in several ways. It seems to be an example of what the philosopher John Austin [1] calls a "descriptive fallacy". Such fallacy means that language is considered to be used *only* for descriptive purposes. But language is used for many other purposes. Language is used to command, request, question, promise, warn, assign and many other purposes [1, 33, 21]. Information systems are also used for other purposes than pure description. From this language action view information systems should not be considered as "containers of facts". Information systems are vehicles for communication among people and organisations.

Supplier:	Pencil-Maker Inc
Customer:	Office Service Inc

Delivery of 200 pencils

Fig. 1. Example of a propositional content, which can be embedded in different types of communicative acts

This paper takes the standpoint of language action theory, which emanates from speech act theory [1, 33] and communicative action theory [21]. Such an approach emphasises that people *act* when they communicate. This theoretical approach has a growing impact on the IS community. Flores & Ludlow [12] and Goldkuhl & Lyytinen [18] made Early contributions. Winograd & Flores [39] made a seminal work. Many recent publications can be found in [10] and [11], i.e. proceedings from the international workshops on the language action perspective on communication modelling.

The main message in the language action approach is that an utterance (message) is combination of a propositional (information) content and an illocutionary force (an action mode). Propositional content is what is talked about and the illocutionary force means what kind of action is performed. When we communicate we formulate a propositional content and embed this in a communicative action type. A propositional content, as described in figure 2, can be used in many different utterances (messages). It can be used in an offer, an order, a promise, a contract, a prediction, a question and a report and also possibly other types of communicative acts.

2.1 Organisations as Action

An enterprise performing business must sell and buy; i.e. it must be an actor in the market place. It will usually act in different roles - as a supplier trying to sell its products and as a customer purchasing products in order to create conditions for selling. Organisations can be viewed from a language action perspective. Business interaction means besides exchange of value also business communication. Proposals are given from customers and suppliers. These proposals are evaluated and possibly transformed into orders and contracts, which are mutual commitments. A business transaction means exchange of value; i.e. a product (goods or services) is delivered and payment is made in return. Business communication means exchange of proposals and commitments.

The Action Workflow model [9] can be used for description of business interaction between a customer and a "performer". It describes the business interaction consisting of four phases: 1) preparation, 2) negotiation, 3) performance and 4) acceptance (ibid.). A more thorough model is the Business Action Theory (BAT) model [14]. Business interaction has here been divided into six generic phases:

1. Business prerequisites phase
2. Exposure and contact search phase
3. Contact establishment and proposal phase
4. Contractual phase (order and delivery promise)
5. Fulfilment phase (delivery and payment)
6. Completion phase (acceptance or dissatisfaction with possible claims)

The first phase means the establishment of prerequisites (of both supplier and customer) for the consequent interactive business performance. The five other phases means exchanges of different character. These exchanges imply actions directed towards the other party; i.e. exchange of 2) interests, 3) proposals, 4) commitments, 5) value and 6) acceptances or claims.

These two models (Action Workflow and BAT) build explicitly on the language action perspective. It helps us to see the action performed by organisations. To sum up: organisations are business actors performing business action. Such business action involves to a large degree communicative action directed towards other organisations (their customers and suppliers). Such communication is not mere information transfer. It can include intentional influence and the making of commitments. In a business setting, the recipient of a “business message” must trust what is said (and thus communicatively done) by a speaker.

2.2 Information Systems as Action

Using an information system in a business firm must be understood within a framework of the organisation as a business actor. Every information system should in principle be possible to relate to one or several of the six phases of business interaction (described above). The *business value imperative* by Porter [29] and BPR advocates, e.g. Hammer & Champy [22], is understood that all activities in the firm should directly or indirectly contribute to the value for the customer. An information system should include activities that contribute to the creation of value to the customer.

An information system can be used directly in the business interaction. IS can be used for

– exposing the capacity and products of the firm
– making offers
– giving delivery promises
– governing the deliveries
– invoicing
– reclaiming payments

In such situations the firm is using information systems for performing parts of its business action. An information system can also be used for other purposes than direct business interaction. It can be used for background activities supporting the direct business interaction.

This language action view of an information system emphasises that such a system is more than an “information container”. An information system is not restricted to only hold information about its environment. An IS is used to perform communicative acts important to business.

In the language action perspective, which we follow, an information system is interpreted to be a “information action system”. This means that the system is capable to perform actions. It is important to add that we do not presuppose computers to have human properties of consciousness and ethical responsibility. An information system performs actions that are predefined to them by human actors. IS does not find out what kind of actions to perform. This is always predefined to the system. This view helps us to see IS as information agents in an organisation [38]. There can be different agents acting and interacting communicatively in an organisation; i.e. human agents and IT artefacts (information systems). Both IS and human actors can thus perform actions on behalf of the organisation.

Even if IS and humans have partly similar action capabilities, this does not mean that we confuse their other properties. It is only human actors who have consciousness and responsibility. We do not accuse a computerised for errors performed. We accuse those humans who have given the system faulty instructions.

An information system has thus possibilities to perform communicative actions. These actions are governed by specified rules. Those rules can be interpreted as the *action potential* of the system. An IS has an *action repertoire* predefined by such action rules. When performing IS actions it is many times necessary to have access to other earlier actions. E.g. when checking an incoming delivery there should be possible to access the corresponding order. An information system must hold a *memory of actions* already performed and other important prerequisites for action.

As a system for communicative action it will of course hold a vocabulary of the action topic. Different concepts and their associated terminology is part of the system's action potential. This means a conceptualisation of what is talked and talked about.

An information system has a dual action character. It can both 1) be an instrument (tool) for users to perform action and 2) to perform action independently of its users (but of course not independently of its predefined rules). A user (e.g. an order clerk) can put an order assisted by an order information system (1). An IS can also perform actions by itself, e.g. checking for delivery possibilities (2). In situation (1) as well as (2) the information system can be furnished with an ability to keep record of these communicative actions (in its action memory) and to use them for other purposes later on.

To summarise our view of an information system: such a system consists of

- action potential, including a vocabulary (a conceptualisation) and repertoire of predefined actions
- actions
- action memory

Another way to conclude is to say that information systems have action ability; *actability*. In the HCI literature, e.g. [30], it is well argued that information system must have usability. We do not deny this, but we will argue that the most fundamental property of an information system in a business context is its actability; i.e. its ability to support actions and to perform actions. Usability should be considered as part of this actability. A good usability means among other things that the actions of the system are transparent and accessible to its user. The action potential and the action memory of the system must be possible for the users to access.

This language action view of information system may have several functions. It can serve as a new and fresh way to *conceive* information systems; i.e. a way to interpret IS. It can also be used to *evaluate* existing information systems. And this view can be used to *design* new information systems and to *redesign* existing ones.

We do not propose that existing information systems (which are not designed according to this paradigm) do not involve any IS action. They do, but perhaps in an obscure way. One major problem of many existing information systems is probably that their action character is not visible enough for their users. The action of the system may be implicit. We argue that new systems should deliberately be designed for actability of high quality, which we claim, is the most important quality issue of an information system. Old IS might be redesigned to be more action transparent. We call this action enhancement *pragmatisation* of IS.

3 An Action Approach to Requirements Engineering

We have above described an action perspective on information systems. What consequences will such a perspective have on information systems development and especially on requirements engineering? One key issue is how to perform requirements engineering from an action perspective? What kind of method support is needed? Our purpose here is to give an outline of a method for requirements engineering based on an action perspective. This approach is called Business and Information requirements Analysis (BIA) and it is part of the SIMM method family. SIMM stands for Situation adaptable work and Information systems Modelling Method. Parts of BIA/SIMM are described below.

3.1 BIA/SIMM: Principal Approach and a Brief Example

When performing business there will be be different types of actions. There will be actions that are external, i.e. actions directed towards customers etc outside the organisation. There will also be internal actions, i.e. actions performed inside the organisation. Different “doers” can perform these different actions. They can be performed by different human actors and by different information systems.

These different actions (of various kinds) together form an integrated whole of the business. Such actions will be part of business processes. There is an obvious need to model such actions and how they are related within a business process. We will perform business process modelling which includes the modelling of IS action. The main instrument for this is Action Diagrams [13, 14]. Two examples of Action Diagrams are found in appendix. It describes both actions performed by human actors and information systems.

In Action Diagrams we describe different actions and how they are related to each other. One important notational feature is that we describe the performer of the action; i.e what actor/actor group or which IS is supposed to perform the particular action. Some actions (in the Action Diagrams) are delimited to be interactive with several performers (e.g. Customer↔Salesman↔System). The action results are described and they can be both informational (communication) and material. The same holds for the prerequisites (the “input”) of the actions.

One important aspect of Action Diagrams is the semantic power to describe action logic. It is possible to describe sequential order of actions (i.e. the flow aspect), alternative actions (decision points), conjunctive actions, contingent actions (i.e. actions occurring only sometimes), trigger (initiation) of actions (by time or communication), interruption of actions (by time or communication), condition for actions, and parallel actions. A contextual descriptive approach is preferably used when working with Action Diagrams. Each Action Diagram describes a business context within a business process. Different Action Diagrams are related to each other through descriptive connectors (i.e. links to other Action Diagrams). The limits of each Action Diagram (=business context) are arbitrary; i.e. the analyst has the freedom to choose the appropriate borders of the described context.

The Action Diagrams give a possibility to model and to design the business and its information system as an integrated whole. The actions of different performers - human actors (within and outside the organisation) or information systems - are described as a whole. We use Action Diagrams to identify and delimit IS action. We can distinguish between different types of IS action as interactive (together with a user) and automatic (by itself). Action Diagrams are used as an important and basic instrument for designing the actability (action ability) of an information system

Through this business process & activity analysis we will also identify documents (IS and other). Concerning IS documents we can distinguish between different types, such as paper documents, EDI documents and screen documents.

A document is a result of a communicative action. A screen document is not only an action result. It is an *action medium*, i.e. a medium for users and IS to perform actions (action interface between user and system). A screen document is a dynamic document. It is possible to use in order to perform actions; i.e. it has a predefined action repertoire. A screen document thus involves

– action potential (predefined set of actions to be performed)
– action (that is performed and at least partly documented in the screen documents)
– action memory (earlier messages/actions can be viewed in screen documents)

The different IS documents must be described in requirements engineering. Required properties of a document are described in a Document Definition (example in appendix).

The content and structure of each document is then analysed and modelled. We perform a conceptual modelling of each document; an example of a Concept diagram (a kind of E/R Diagram) is found in appendix. In BIA/SIMM we use a contextual approach (“bottom up”). Different Concept Diagrams (of each document) can be later be put together into a global conceptual model of this IS.

The relationships of different actions and their documents are described in a business action context in the Action Diagrams. Relationships between documents should be highlighted in a requirements definition. We make an abstraction from Action Diagram and model explicitly the relationships in Document Diagrams. There are two types of Document Diagrams. We describe the conceptual relationships in a Conceptual Document Diagram. Action relationships are described in an Action Document Diagram; an example is found in appendix.

We have described and exemplified some important notations of BIA/SIMM. These different models do not give full requirements definition. We need also to model and evaluate other aspects. It is however beyond the purpose and scope of this paper to go into detail of more models. In section 3.3 some other parts are mentioned.

3.2 BIA/SIMM and the Development Phases

Information systems development according to SIMM should be preceded by Change Analysis (CA) [19, 20].

The CA-phase is important to understand the goals, problems and action logic in the business The CA ends with a decision point on whether a computerised information system is to be developed or not. If such a system is to be developed the next step is Business and information requirements analysis (BIA).

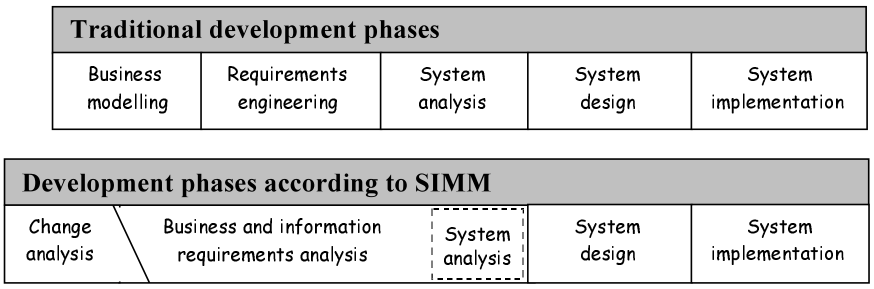


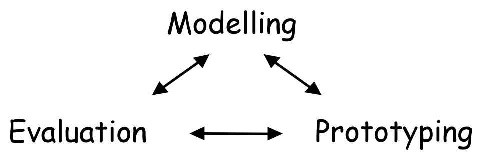
Fig. 2. The SIMM framework for business development

When using BIA/SIMM the traditional partitioning of early analysis into business modelling and requirements engineering is, more or less, obsolete. BIA is analysis of business, requirements and information system, all at ones. We believe that development of information systems always implies business development and the two activities should therefore never be viewed in isolation. Note that the SIMM framework doesn't specify any particular lifecycle model for the development process. Instead SIMM can be used with for example a traditional waterfall model [36] but it will work as well, or even better, with a more iterative incremental model such as Boehm's spiral model [3]. It is our intention that the SIMM framework can and will appear differently in different development situations. Just as McConnel [28] points out "… the most effective [*lifecycle*] model depends on the context in which it's used".

3.3 Components of BIA/SIMM

Business and information requirements analysis according to SIMM (BIA/SIMM) consist of three interacting parts: modelling, prototyping and evaluation. Evaluation is thus based either on the analytical model or on developed prototypes. The modelling part is performed in two primary activities; Business Process Modelling (BPM) and Information System focused Modelling (ISM). In BPM there is a focus on the business context and the roles of information systems within it. In ISM the focus is on action, interaction and conceptualisation of the information system, with the business as context. Even though BPM and ISM are considered distinct activities there is overlapping between them and they are typically iterated many times during development.

Fig. 3. BIA/SIMM as three interacting parts



The main activity within business process modelling (BPM) is Activity Analysis (AA) during which the business structure is analysed and expressed in action diagrams. The model shown in action diagrams is focused on business actions, actors and action objects. The model captures the flow of both information and physical objects. Another important part of BPM is Goal Analysis (GA) during which the business's goals are identified and structured in goal/sub-goal hierarchies. The two activities are tightly coupled since the actions identified during action analysis are intentional and such intentions are typically goals, and therefore studied, within the goal analysis.

The information system focused modelling consists primarily of document analysis (DA), conceptual analysis (CA) and interaction analysis (IA). Embedded within the action diagrams from AA are the main *interactive situations*, i.e. the activities that are performed by an actor in interaction with the system. The granularity and precise content of these action-aggregates is not yet verified so we name them *interactive situation proposals* or ISP's for short. The ISP's form the basis for extracting requirements concerning both the interactive documents and the systems functionality. To find and verify the basic interactive actions that aggregate to an interactive situation we use a mixed analytical and experimental approach as shown in figure 4.

By use of early prototypes [34, 5] we can verify the analytical assumptions at an early stage. We do however believe that an analytical approach still is necessary to ensure that the required actability is satisfied and to help express functional requirements. Speed can never compensate wrong direction. The analytical approach is, in this case, interaction analysis (IA) where the ISP's are analysed and refined. Relating the ISP's to each other increase potential reuse of requirements and analysis results.

During IA the action potential of the system (functional requirements) is in-depth analysed and related to users interactive actions. The systems screen documents are identified (or invented–depends on how you look at it) and their different roles as action medium are elaborated and defined.

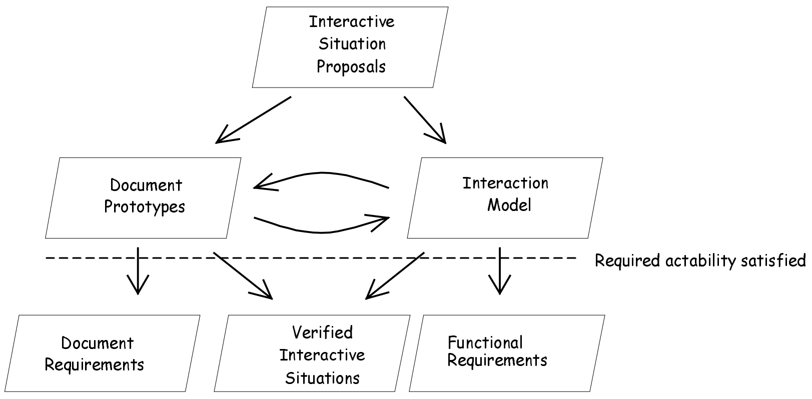

Interactive Situation Proposals
Document Prototypes
Interaction Model
Required actability satisfied
Document Requirements
Verified Interactive Situations
Functional Requirements

Fig. 4. A mixed analytical and experimental process

One important part of IA is the mapping of the interactive (business) actions to the creation, use and updating of action memory (e.g. the systems database). These mappings are then used during conceptual analysis (CA) to create life history programs for the concepts used within the business (and the IS). CA also means definition and analysis of the concepts used within each document (the propositional content), their relations to each other and the roles each concept and relation play in the total IS. The documents are then related to each other by means of action relations and conceptual relations during document analysis (DA). DA also involves the visual layout of documents and analysis of document properties such as action mode and document type (i.e. medium such as "screen document" or "EDI document").

Thus, information system focused modelling consists of the following three interrelated parts:

- *Conceptual analysis*, which is oriented towards *what to communicate* and means requirements engineering of: definitions, conceptual relationships and concept dynamics.
- *Interaction analysis*, which is oriented towards *how to communicate* and means requirements engineering of: roles of screen documents, relations between different interactive situations, interactive actions (both user and system) and finally creation, use and updating of action memory.
- *Document analysis*, which is oriented towards the *documents as action objects* and the *communication context* and means requirements engineering of: docu-

ment properties, relationships between documents (both action and conceptual) and document layouts.

4 Relations to Some Current Approaches

A method can be described and understood at different levels of abstraction and from different points of view. At the core level a method can be viewed as consisting of three integrated parts; the concepts used in the method, their notational representation and a process to guide the method users actions, e.g. [17, 31, 2]. Every method is also based on some theory or at least some more or less verbalised notion of the world [17]. We call this frame of reference the methods perspective (even though it's actually the method creator's perspective that is implicit in the method). In this section BIA/SIMM is related to some current approaches at these two abstraction levels.

4.1 To Watch or to Participate

One basic assumption within the field of e.g. semantic modelling, e.g. [7], and object-orientation, e.g. [4, 32], is that the information system should serve as an image, or simulation, of reality (or rather one piece of reality called "the universe of discourse"). The information system could then be used to inform its users about the world, so that the users don't have to observe the world directly. We do however believe that there exist dimensions in our minds beyond the strict conceptual one. People don't just look at the world and talk about it. They do things, act and communicate in the world. Utterances carry more than just mere facts that tell something about something. They also carry the actor's intention. Utterances per se can therefore by viewed as actions. These properties of information as action are most often neither discussed nor taken to account in traditional approaches to requirements engineering. A fact that often leads to systems that doesn't fulfil the needs of its actors. We could talk about "unactability", i.e. the situation that occurs when actors are not able to fulfil their intentions.

4.2 Concepts of BIA/SIMM

The key concept in traditional dynamic modelling is the event, e.g. in Jackson System Development (JSD) [23, 6] and various object-oriented methods, e.g. [4, 8, 32]. Booch [4] defines an event as "… some occurrence that may cause the state of a system to change". In a social context things most often doesn't just happen. An event, from the systems point of view, is almost always the result of some action performed by some actor. Since an action is always based upon some intention we could say that an action is similar to an event but with additional attributes and constraints. Our approach thus gives an opportunity to identify the "events" needed for system design based on the actions performed in the business. In that way the requirements specification doesn't just state *what* the system must be capable of doing but also *why*. Note that the same business action model is also used to identify interaction situations. The traceability from computerised system to business is thereby straightforward regarding both functional requirements and sequence restrictions. Snoeck and Dedene [35] mention that "modelling interaction by means of common event types requires the identification of relevant business event types" but they do not mention how to do that. We argue that such relevant business types are to be found by identifying *business actions*. It is interesting to note that Blaha and Premerlani [2] suggests that dynamic modelling most often isn't necessary when developing database applications. They claim that objects in such a system doesn't have interesting dynamic behaviour. That is a claim in contradiction to Snoeck and Dedene [35]. The latter's method M.E.R.O.DE. is quite focused on dynamic modelling and the concept of shared events–similar to JSD. We believe that all actions, with corresponding events, that have an effect on (create, modify or delete) the action memory are important for information systems.

Our notion of interactive situations is quite similar to the use case concept [24, 25]. Also our way to derive interacting situations is quite similar to the way Jacobson et al [25] use the business object model to define use cases for the supporting information system. That similarity can be useful during systems design and implementation due to reuse of knowledge in those domains. One major difference lies in the fact that Jacobson's et al [25] definition of use case comprises "…flow of events" as a central notion. In contrast our interactive situations are constituted by "flow of actions". Therefore the above critic of the concept of event is also applicable to use cases. We thus disregard use cases and objects as modelling primitives during business modelling. Instead we use action diagrams to capture and describe interactive situations. We believe that the formal systemic approach that is implicit in object-orientation is not sufficient to understand a social system constituted by actors acting and interacting–a system such as a business. Therefore we have chosen not to talk about objects but rather about the terminology and concepts used in the business context. Since the concept's life cycles and propositional contents are analysed we are rather close to object-orientation. The important point is that we defer till design whether we should commit to object-orientation or not. In fact, we believe that a requirement engineering method should deliver models that don't depend on any particular implementation paradigm. We do however believe that object-orientation is a powerful and valuable tool when it comes to the computerised part of the information system. We also believe that if we chose an object oriented implementation strategy the documentation from BIA is a good starting point for object oriented analysis (i.e. much of the work is already done).

A problem regarding use cases is the level on which they should be defined, e.g. [37]. We aggregate activities based on the actor's intentions to form interactive situations. That approach should also be applicable to use cases if a move from the notion of events to the notion of actions were performed.

4.3 The Expressive Power of Action Diagrams

The results of BIA are expressed in a variety of notations covering different important aspects of the modelled business and information system. Since BIA is oriented towards business actions and such actions are expressed in action diagrams, that particular form of notation constitute a basis from which in-dept modelling of other aspects, expressed in other notations, is performed. At a first glance action diagrams may look a lot like data flow charts in structured analysis [40] or A-graphs in ISAC [26]. There are however important differences. Action diagrams are contextual and not compositional even though they can be used in a compositional way if necessary. Action diagrams are not limited to flow and storage of information. In action diagrams one can express things as material flows, and personal resources. The latter can for example be useful in modelling responsibilities, which is important in quality engineering. See [13] for a detailed comparison between action diagrams and other modelling techniques.

One can argue that added semantic power to a model yields lower precision in the sense that the model is hard to prove formally correct. There is no formal mathematical underpinning or even definition of action diagrams. We believe that high semantic richness is required to capture the complexity of a social system and that lower precision might be the price to pay for higher relevance. One must remember that precision can never substitute relevance in any model. When it comes to design of the computerised parts of the business, more formal models are often needed though.

5 Conclusions

This paper presents an action approach to requirements engineering named “Business and Information requirements Analysis” (BIA). As an implication of such an approach a quality of information systems that deserve special attention is the information system actability. We believe that requirements engineering ought to be viewed as *actability engineering* in order to perceive information systems as action. With such an understanding we begin to see how important information systems are for the business actions of organisations. Therefore, when using BIA, the IS action and interaction is derived from the modelling of business processes. Note that Business Process Modelling (BPM) form a basis for Information Systems Modelling (ISM) and that ISM, even though focused on the information system, still is action oriented. Thus, requirement specifications serve a double purpose within BIA. The first purpose is definition of IS actability and the second is to serve as a good basis for system design.

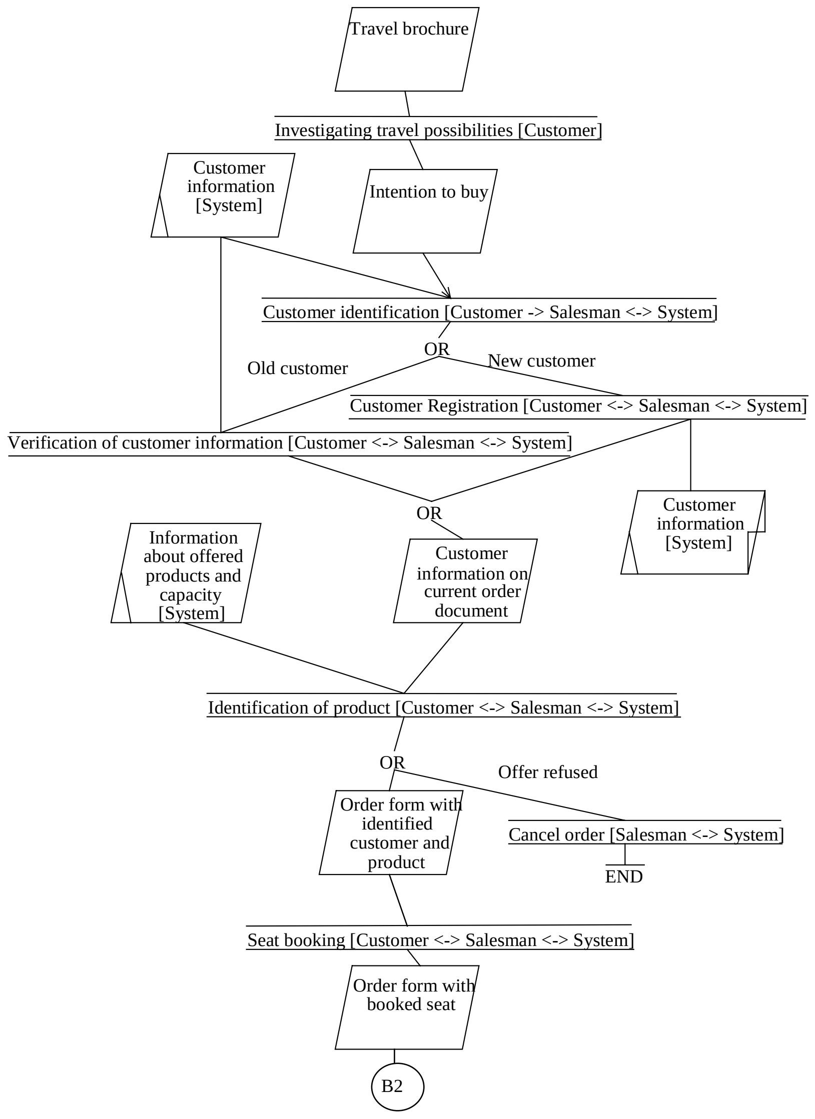
References

1. Austin J L (1962). How to do things with words, Oxford University press.
2. Blaha M and Premerlani W (1998). Object-oriented modeling and design for database applications. Prentice hall, Inc. Englewood Cliffs, New Yersey.
3. Boehm B W (1988). A spiral model of software development and enhancement. IEEE Computer, 21 (5), 61-72.
4. Booch G (1994). Object-oriented analysis and design with applications. 2:nd ed. Benjamin/Cummings.
5. Budde R, Kautz K, Kuhlenkamp K, Züllighoven H (1992). Prototyping: An approach to evolutionary system development. Springer-Verlag.
6. Cameron J (1989). JSP & JSD: The Jackson approach to software development. IEEE computer society press.
7. Chen P (1976). The entity-relationship model–toward a unified view of data. ACM TODS 1, No. 1 mars 1976.
8. Coad P, Yourdon E (1991). Object-oriented analysis. Object international, Inc. Published by Prentice Hall, Inc. Englewood Cliffs, New Yersey.
9. Denning P J, Medina-Mora R (1995). Completing the loops, Interfaces
10. Dignum F, J Dietz (Eds 1997). Communication modelling - the Language Action Perspective. Proceedings of the 2nd International workshop on Communication modelling, Computer science Reports, Eindhoven University of Tecnology, http://www.win.tue.nl/win/cs
11. Dignum F, J Dietz, Verharen E, Weigand H (Eds 1996). Communication modelling - the Language Action Perspective. Proceedings of the 1st International workshop on Communication modelling, Electronic Workshops in Computing, Springer Verlag.
12. Flores F, Ludlow J (1980). Doing and speaking in the office, in Fick & Sprague (Eds, 1980) Decision support systems: Issues and challenges, Pergamon Press
13. Goldkuhl G (1992). Contextual activity modelling of information systems, in Proceedings of "3rd int working Conference on Dynamic Modelling of information systems", Noordwijkerhout
14. Goldkuhl G (1996). Generic business frameworks and action modelling, In proceedings of conference Communication modelling - Language/Action Perspective´96, Dignum et al (1996)
15. Goldkuhl G, Röstlinger, R (1988). Förändringsanalys (Change analysis). In swedish. Studentlitteratur, Lund, Sweden.
16. Goldkuhl G, Cronholm, S (1993). Customizeble CASE environments: A framework for design and evaluation. COPE'IT'93/NORDDATA, Copenhagen.
17. Goldkuhl G, Lind M, Seigerot U (1997). Method integration as a learning process. Research report, Linköping University, Sweden.
18. Goldkuhl G, Lyytinen K (1982). A language action view of information systems, In Ginzberg & Ross (Eds, 1982) Proceedings of 3rd International Conference on informations systems, Ann Arbor
19. Goldkuhl G, Röstlinger A (1984). The legitimacy of information systems development - a need for change analysis, in Proceedings of IFIP Conference Human-Computer Interaction, London

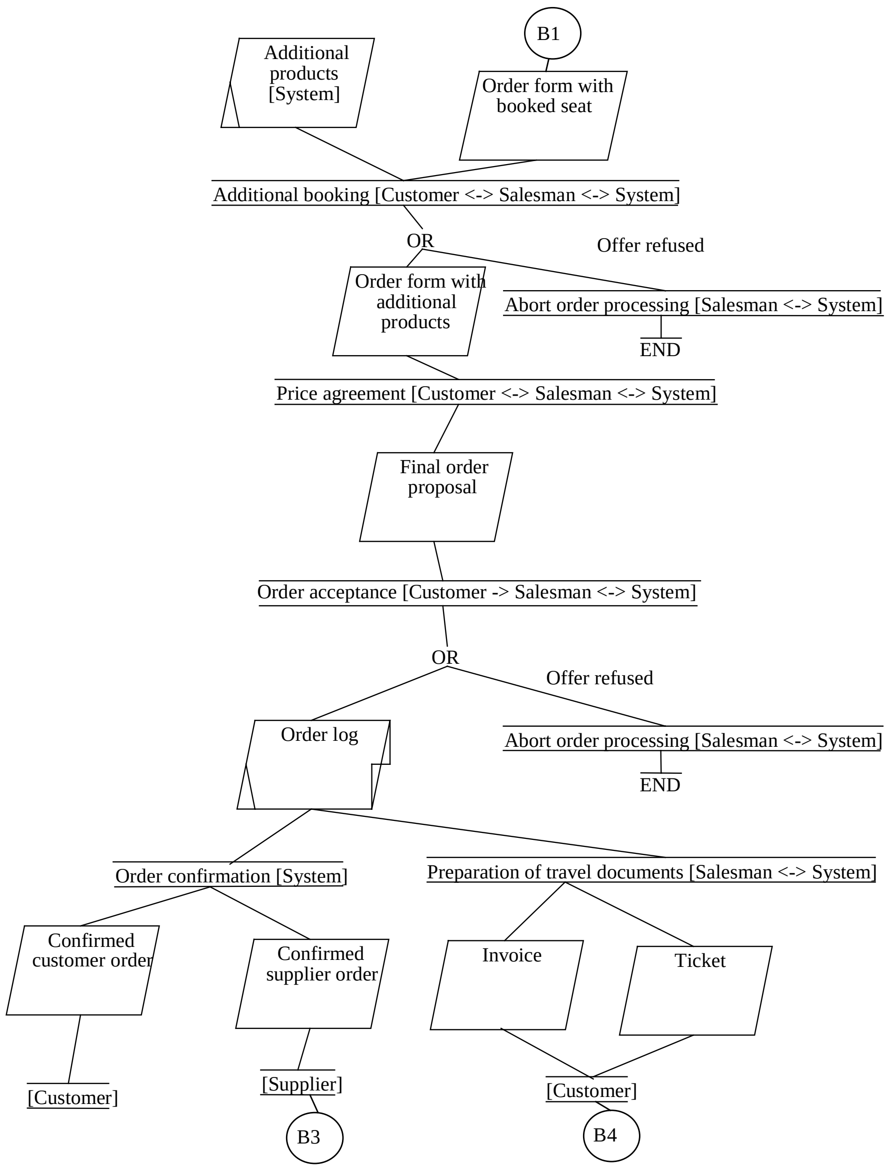
20. Goldkuhl G, Röstlinger A (1993). Joint elicitation of problems: An important aspect of change analysis, in Avison D et al (1993, Eds) Human, organizational and social dimensions of Information systems development, North-Holland/IFIP w.g. 8.2
21. Habermas J (1984). The theory of communicative action 1. Reason and the rationalization of society, Beacon Press
22. Hammer M, Champy J (1993). Reengineering the corporation. A manifesto for business revolution, Nicholas Brealey, London
23. Jackson M A (1983). System development. Prentice Hall, Inc. Englewood Cliffs, New Yersey.
24. Jacobson I, Cristerson M, Jonsson P, Övergaard G (1992). Object-oriented software engineering: A use case driven approach. ACM-press.
25. Jacobson I, Ericsson M, Jacobson A (1995). The object advantage–business process reengineering with object technology. ACM-press.
26. Lundeberg M, Goldkuhl G, Nilsson A G (1981). Information systems development: A systematic approach. Prentice Hall, Inc. Englewood Cliffs, New Yersey.
27. Martin J (1989). Information engineering–Introduction. Prentice Hall, Inc. Englewood Cliffs, New Yersey.
28. McConnel S (1996). Rapid development. Microsoft press.
29. Porter ME (1985). Competitive advantage. Creating and sustaining superior performance, Free Press, New York
30. Preece J, Rogers Y, Sharp H, Benyon D (1994). Human Computer Interaction. Addison-Wesley.
31. Rumbaugh J (1995). What is a method? Technical Paper, Rational software, Inc.
32. Rumbaugh J, Blaha, M, Premerlani W, Eddy F, Lorensen W (1991). Object-oriented modeling and design. Prentice hall, Inc. Englewood Cliffs, New Yersey.
33. Searle J R (1969). Speech acts. An essay in the philosophy of language, Cambridge University Press, London.
34. Smith M F (1991). Software prototyping: adoption, practise and management. McGraw-Hill.
35. Snoeck M and Dedene G (1998). Existence dependency–the key to semantic integrity between static and behavioural aspects of object types. Research report. Katholieke universiteit Leuven, Belgium. To appear in IEEE software.
36. Sommerville I (1992). Software engineering. 4th ed. Addison-Wesley Publishing Company Inc.
37. Vemulapalli C (1995). A Use Case FAQ. Advanced Software Technologies Group, WorldCom Inc. Contribution to the first Workshop on Use Cases/OOPSLA '95.
38. Verharen E (1997). A language-action perspective on the design of cooperative information agents, Ph D thesis, KUB, Tilburg
39. Winograd T, Flores F (1986). Understanding computers and cognition: A new foundation for design, Ablex, Norwood
40. Yourdon, E (1989). Modern structured analysis. Prentice Hall, Inc. Englewood Cliffs, New Yersey.

Series	**ACTION DIAGRAM**		**Appendix A**
REFSQ'98			
Creator	**Date**	**Version**	**Ref. code**
Goldkuhl, Ågerfalk	2.27.98	1	B1

Conserns: Order phase 1

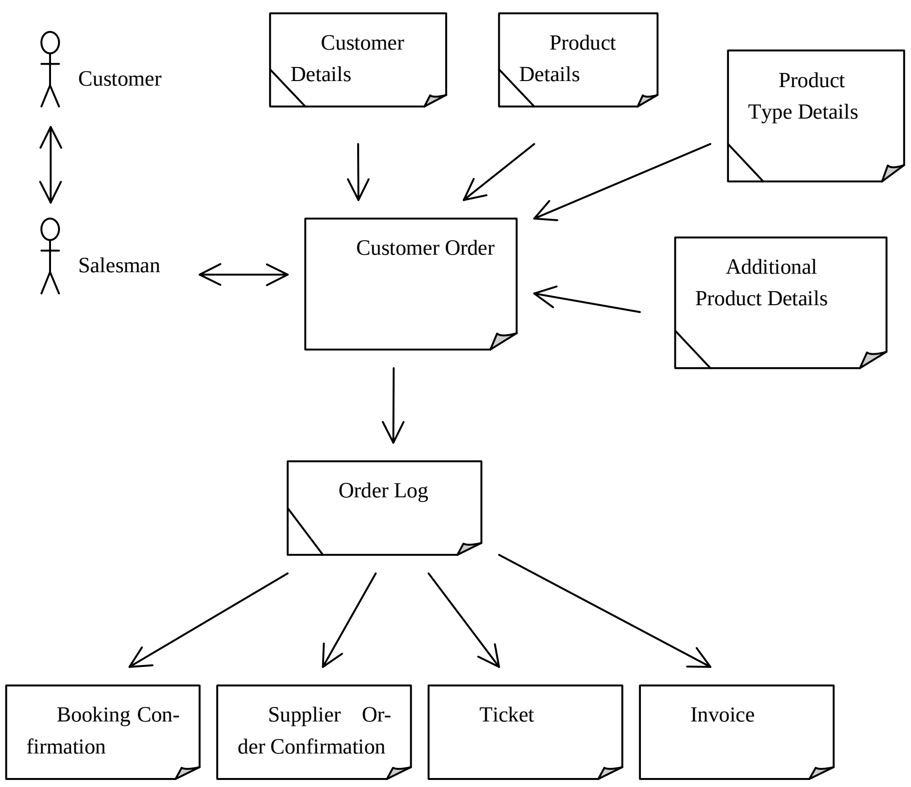

Travel brochure
Investigating travel possibilities [Customer]
Customer information [System]
Intention to buy
Customer identification [Customer -> Salesman <-> System]
OR
Old customer
New customer
Customer Registration [Customer <-> Salesman <-> System]
Verification of customer information [Customer <-> Salesman <-> System]
OR
Customer information [System]
Information about offered products and capacity [System]
Customer information on current order document
Identification of product [Customer <-> Salesman <-> System]
OR
Offer refused
Order form with identified customer and product
Cancel order [Salesman <-> System]
END
Seat booking [Customer <-> Salesman <-> System]
Order form with booked seat
B2

Series	**ACTION DIAGRAM**		
REFSQ'98			
Creator	**Date**	**Version**	**Ref. code**
Goldkuhl, Ågerfalk	2.27.98	1	B2
Conserns: Customer order			


B1
Additional products [System]
Order form with booked seat
Additional booking [Customer <-> Salesman <-> System]
OR
Offer refused
Order form with additional products
Abort order processing [Salesman <-> System]
END
Price agreement [Customer <-> Salesman <-> System]
Final order proposal
Order acceptance [Customer -> Salesman <-> System]
OR
Offer refused
Order log
Abort order processing [Salesman <-> System]
END
Order confirmation [System]
Preparation of travel documents [Salesman <-> System]
Confirmed customer order
Confirmed supplier order
Invoice
Ticket
[Customer]
[Supplier]
[Customer]
B3
B4

Series	**DOCUMENT ACTION DIAGRAM**		
REFSQ'98			
Creator	**Date**	**Version**	**Ref. code**
Goldkuhl, Ågerfalk	2.27.98	1	DA1

Conserns: Customer order

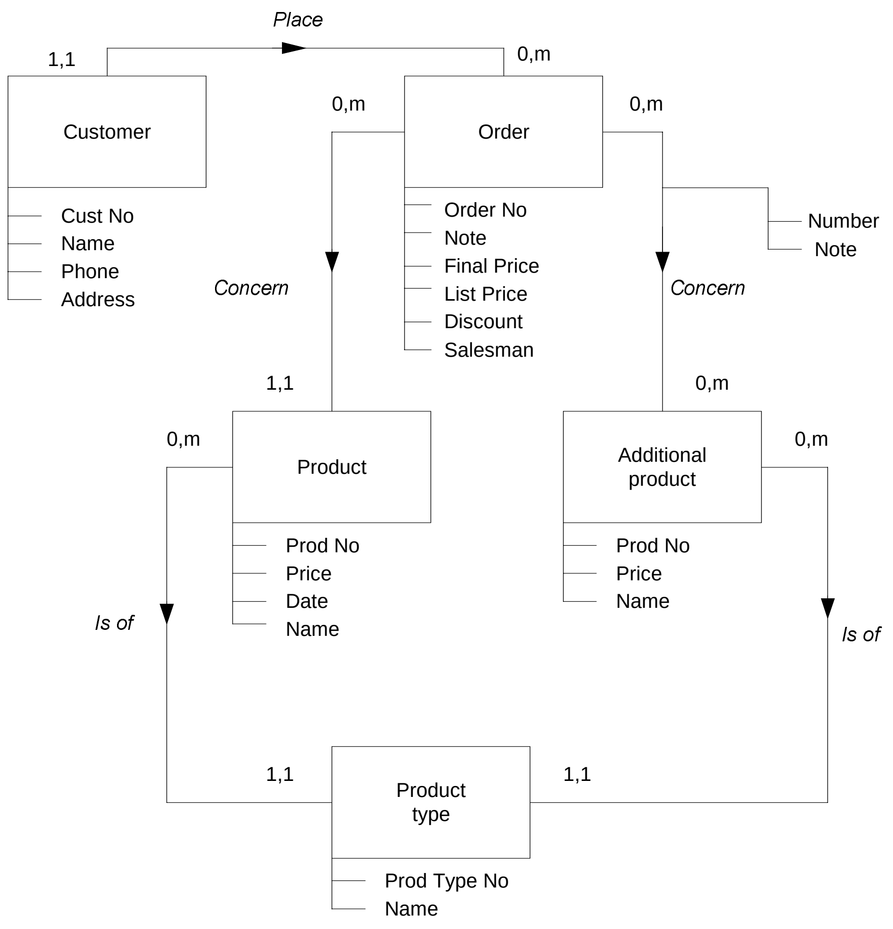


Series **CONCEPT DIAGRAM**

REFSQ'98

Creator	**Date**	**Version**	**Ref. code**
Goldkuhl, Ågerfalk	2.27.98	1	CD1

Conserns: Customer order



Series	**DOCUMENT DEFINITION**		
REFSQ'98			
Creator	**Date**	**Version**	**Ref. code**
Goldkuhl, Ågerfalk	2.27.98	1	DD1

Conserns: Customer order (order form)

Kind of document	Interactive screen document / paper document
Communicator(s)	Customer and salesman
Registrar	Salesman
Action mode	Business contract between customer and supplier
Content	Customer information, product information, seats, additional products, prices
Communication effect	Maintain order register and thereby enable creation and retrieval of business documents needed to fulfil the deal.