

# From Business Process Design to Electronic Business Engineering

Arnout Bruins, Paul Oude Luttighuis, Maarten Steen  
Telematica Instituut, P.O. Box 589, 7500 AN Enschede, The Netherlands  
{bruins, oudeluttighuis, m.steen} @telin.nl

Developing e-business solutions is a complicated task. It involves many different disciplines and requires knowledge of e-business technologies as well as business processes. In this paper we describe our models and concepts for e-business engineering. We illustrate this with a fictitious case.

## 1. Introduction

Most work on business process modelling and engineering has been driven by the enabling power of information technology to change business processes. Evolving electronic business technologies stretch over the boundaries of individual organisations and cover entire markets, chains, and networks of companies. Still, main efforts in electronic business research are either concerned with technology or with the strategic, commercial, and marketing implications. Attention is lacking to issues of actually (re-) engineering business(es) along with the introduction of electronic business (Biemans, et al 99). Classical Business Process Engineering methods, concepts, tools, and techniques (as described in Ould 95, Davenport 90, Venkatraman 95) insufficiently capture the complexity of electronic-business engineering (EBE). In particular, they generally fail to:

- capture cross-company business processes and transactions,
- capture relevant higher-level concepts, such as roles, responsibilities, chains, markets, etcetera,
- capture dynamic changes in business processes,
- rapidly translate into software engineering because traditional methods are time consuming, and
- make use of business process model components which results in reinventing the wheel.

Exceptions may be found for some of these, but no existing EBE approach covers all, or even most, of these characteristics. This paper presents ongoing work on EBE, as carried out by the Giga Transaction Services project. In particular, it focuses on the concepts used in modelling networked enterprises and illustrates these with an example.

In this paper we describe the models and concepts we use for modelling Networked Enterprises, the transaction scenarios, and the detailed procedures that can be automated. Which models are necessary, what is the connection between the models and which concepts are needed to build the models, are questions raised which we try to answer. In paragraph 2 we describe the Rapid Service Development Methodology (RSD), this is the overall framework. In paragraph 3 we describe the concepts for Networked Enterprise Modelling, such as the actors, roles and functions. Next we describe our formalism for modelling inter-organisational business processes and transaction scenarios. Finally we give some directions for implementation.

## 2. Rapid Service Development

The RSD methodology defines an integrated framework for business-driven design of transaction services. Generally, the development of such services is highly complex, as it involves many different aspects ranging from high-level strategic business concerns to low-level protocol definitions. In order to deal with this complexity, the 'separation of concerns' principle is applied. In total, the RSD framework distinguishes seven different aspect areas, called *cornerstones*, from which models and specifications can be made.

The cornerstones of the RSD are structured along two dimensions, as illustrated in Figure 1. Firstly, we distinguish between business-oriented models (on the left) and technology or system-oriented models (on the right). Secondly,

models can vary in scope or granularity. Along this axis, they range from high-level, broad scope, coarse grained, little detailed models (at the top) to low-level, narrow scope, small grained, highly detailed models (at the bottom). The ultimate goal is not only a technological solution (system realisation), but one that is integrated with the realisation of the (re-) engineered business models as well.

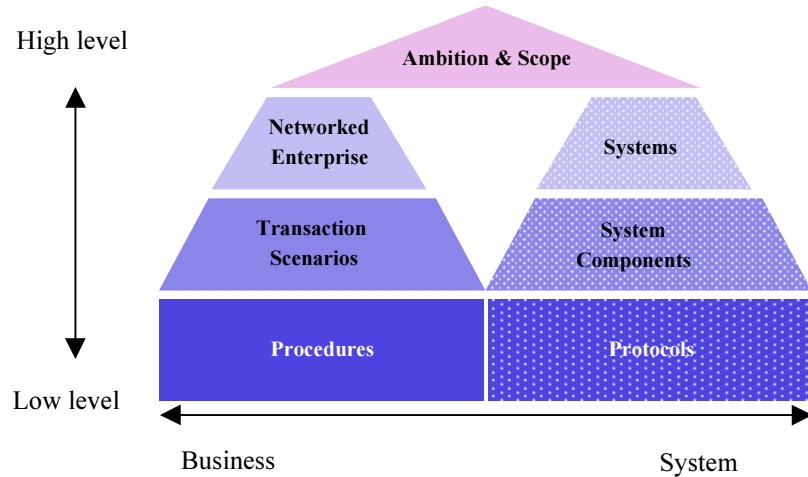


Figure 1: The RSD framework

On the business-oriented side, we model and design the way organisations co-operate in a networked enterprise. To this end, we use

- **Networked enterprise models**, which give a high-level view of the co-operation in terms of the actors involved, the roles they fulfil and their relationships, but also in terms of the business functions they perform and the flows between these.
- **Transaction scenarios**, which describe the inter-organisational business processes and transactions that achieve the co-operation, and
- Specifications of the **procedures**, which describe in more detail the interactions and message formats for conducting transactions between organisations.

On the system-oriented side, we model and design the technology that supports the co-operation between organisations in a networked enterprise. To this end, we use

- **System descriptions**, which describe the architecture and the composition of transaction systems,
- **Component specifications**, which describe the functionality of well-encapsulated, reusable pieces of software, and
- **Protocol and code specifications**, which describe in detail the protocols used for communication between components, as well as the internal behavior of these components where necessary.

**Ambition and scope** can be formulated both in terms of business goals and in terms of technology. The crossover from business oriented modelling and design to system oriented modelling and design is somewhere between procedures and protocols. Here the division between the two ends of the spectrum gets blurred. At the higher levels of abstraction, there is a more pronounced gap (also visualised in Figure 1) between business modelling and technology-oriented modelling. In this position paper we do not go into the details of all cornerstones. We focus on the ones that are most relevant to the scope of the workshop, i.e. networked enterprises and transaction scenarios.

Hence, where many approaches cover only parts of the issue, the RSD provides an integrated framework covering:

- both business and technology;
- both high level and low level models;
- flexibility by allowing multiple strategies, we call these path ways.

### 3. Concepts for Networked Enterprise Modelling

The central concept in the RSD approach to modeling inter-organisational cooperation is that of a ‘Networked Enterprise’. Networked Enterprises from the unity of analysis and specification and are captured in Networked Enterprise Models (NEM’s). Supply chains, electronic markets, auctions and virtual enterprises are all examples of Networked Enterprises. In general, a Networked Enterprise is any undertaking that involves two or more interacting parties. The concept is close to that of a business model as defined by Timmers (1999). The Networked Enterprise Model describes the main structure of a networked enterprise. The NEM describes the actors involved, the roles they play, the functions they carry out, and the information, goods and money flows among them. We use actor, role and the function diagrams to capture NEM’s.

In addition, we model transaction scenarios and inter-organizational business processes. These diagrams are described in more detail in paragraph 3.2.

We illustrate our method with a fictitious case called Net Profit. Net Profit is a car insurance company that works through agents. Net Profit is confronted with a loss of clients. After a client survey, Net Profit concluded that one of the main causes is the lengthy and complex process of settling damage claims. When car damage occurs, a complicated process starts, including many interactions between parties involved. Net Profit wants to improve the way damage claims are settled using new technological means to communicate with customers and other involved parties. Net Profit therefore aims at designing this process on the basis of the following principles:

1. Information exchange is done electronically where possible.
2. All information exchange between Net Profit and insurant (the insured person) will be via the agents.
3. Agents are responsible for the soundness and completeness of claims.

#### 3.1 Actors, roles and functions

A Networked Enterprise Model (NEM) describes the rough line of a business chain and contains the high-level information needed for subsequently designing transaction scenarios, transaction services, and systems. Within the Networked Enterprise Model we have three elements of analysis: Actors (the organisational entities), Roles (the responsibilities) and Functions (behaviour). These are modelled in respectively the actor, the role and the function diagrams.

Actors are the concrete parties involved in a networked enterprise, an actor can initiate and participate in behaviour. We recognise four actors in this case. The first one is – of course – Net Profit; the second are the agents who are the intermediaries between Net Profit and the insurants. The customers are the insurants this is the 3<sup>rd</sup> group of actors, the last group of actors are the garage companies who fix the broken cars. The double lines in figure 2 indicates that more than one actor is involved, the asterisk gives the number of actors (with the star means an unspecified number). When an actor is replicated it indicates a group of instances, we call this role players.

Between the actors we recognise Channels. A channel represents an interconnection between actors for the exchange of goods or information. Channels are named after the medium.

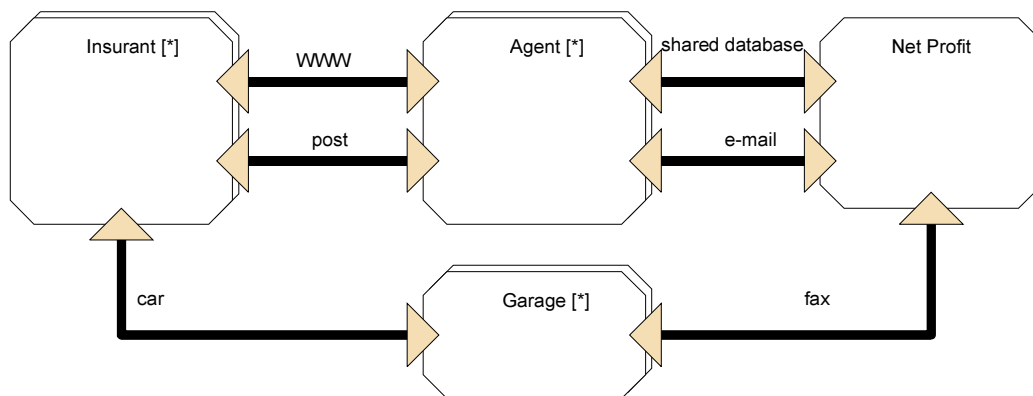


Figure 2 —Actors involved in Net Profit and the channels between them.

Each actor fulfils one or more roles. A role represents a collection of responsibilities that may be fulfilled by one or more actors. The garage, for example, has the role of assessing the damage and the role of repairer, Net Profit has the role of insurer. Roles allow for abstracting from concrete parties. In the Net Profit case there is a one-to-one mapping of actors to roles.

We identify the functions performed by the roles. Functions represent a logical grouping of activity. Examples of functions are payment and distribution. A function can usually be assigned to a single role. In that case, the function represents a task, the execution of which is a responsibility of that role. Figure 3 can be interpreted as follows: the insurant has car damage, next he brings the car to the garage and gives the agent a claim report. The agent transforms the claim report to a claim, and sends the claim to the insurer. The garage assesses the car and sends the damage form to the insurer. The insurer assesses the damage report and the claim and communicates the result to the agent, who - in his turn - informs the insurant. Between the functions we recognise flows. A flow is an abstraction of a sequence (in time) of transfers of information or goods between roles and functions. A flow has a direction from one function towards another function; each flow has a name.

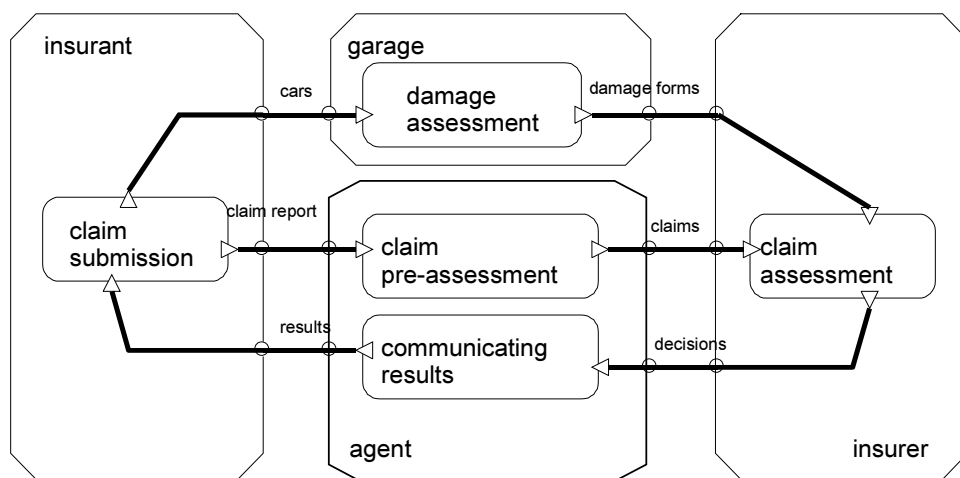


Figure 3 — Function diagram Net Profit.

Where actor diagrams are a traditional ingredient of other approaches, function diagrams are not. In RSD however, they are crucial, because they:

- provide the level at which responsibilities are assigned to the actors;
- are the level at which behavioral components are identified.

### 3.2 From inter-organisational processes to transaction scenarios

In paragraph 3.1 the involved actors, the roles and the functions are modelled. The next step is to model the business processes in more detail. A (business) process is a delineated piece of behaviour, leading to the accomplishment of some result. This relates to concepts as workflow and procedure. A process is composed of a trigger, actions, interactions, transactions, transfers, and causality relations. Triggers and actions are internal to roles and functions. Where a process crosses a role or function boundaries, only interactions, transactions and transfers may occur. The overall claim submission and settlement process for Net Profit is depicted in figure 4.

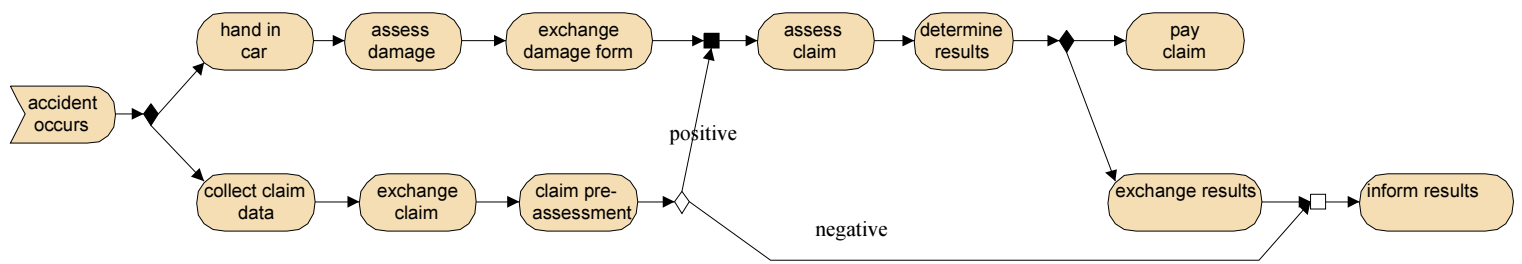


Figure 4 — Overall process Net Profit.

A business process always starts with a trigger. A trigger is an action with no cause; it models an event that signals the start of a process. In our example the trigger is the occurrence of an accident. After the trigger one or more actions are performed. An action is an abstraction of an activity in the real world. Some actions are executed after another action is executed: a causal relation. A causal relation defines the conditions for the occurrence of an action. Within the chain of actions we recognise the ‘AND split’, the ‘OR split’, the ‘AND join’, and the ‘OR join’. After an ‘AND split’, all following actions will be performed, while with the ‘OR split’, only one of the following actions will be performed depending on a condition. In our example there is an ‘AND split’ direct after the trigger, and there is an ‘OR split’ after ‘claim pre-assessment’, with a positive or negative condition. For performing an action after the ‘AND join’ all leading actions need to be performed, with the ‘OR join’ only one of the leading actions need to be performed to continue the process. Before ‘assess claim’ there is an ‘AND join’, and before ‘inform results’ is an ‘OR join’. It is possible to model other concepts as; completion time, process costs, and responsibility. A tool called Testbed Studio supports this part of our method; the modelling language is called Amber. Amber is described in more detail in Eertink (99).

Actions can be assigned to functions, roles and actors. Based on this assignment, transaction scenarios can be generated that depict how the process is distributed (see figure 5). This makes it possible to assign the actions to, for example, roles. See Figure 5. This can be read as follows: ‘the insured person fills out a claim. The agent files the claim and pre-assesses it on the basis of basic rules, without knowing the garage’s damage assessment data. In case the pre-assessment yields a negative result, the insurer is informed immediately. Otherwise, the insurer carries out an assessment on the basis of the garage’s damage assessment and the claim. The agent triggers the insurer for assessing the claim. The insurer reaches a decision and informs the agent, who informs the customer. Finally the insurer pays the corresponding amount to the insured’. The activities that are shared between roles are called transactions. The parallelogram named ‘repairs’, and ‘claim file’ are artefacts. An artefact represents a logical or physical passive object.

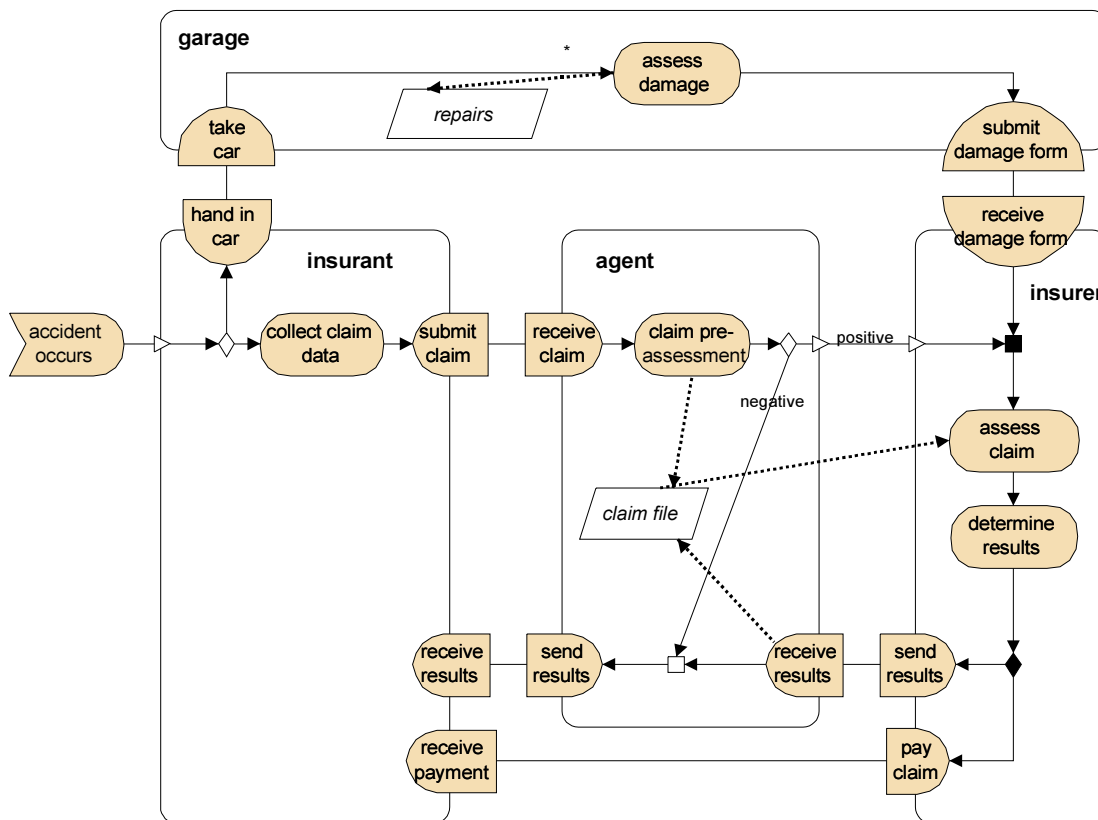


Figure 5 — Transaction Scenario

### 3.3 Towards Implementation

In order to proceed from the transaction scenarios of figure 5 towards implementation, tasks and services need to be more detailed. An example is the transaction to submit a claim. In figure 6 is the detailed procedure description. The procedure can be mapped onto implementation components. The precise way in which this should be done is matter of further investigation. A mapping of the detailed procedure descriptions onto UML (Booch 95) use case (Jacobsen 92) descriptions is envisaged. We implemented the claim submission procedure for Net Profit with the IBM WebSphere; an IBM DB2 SQL database stores all data relevant to the claim submission, where JDBC supports the connection between the application server and the database server.

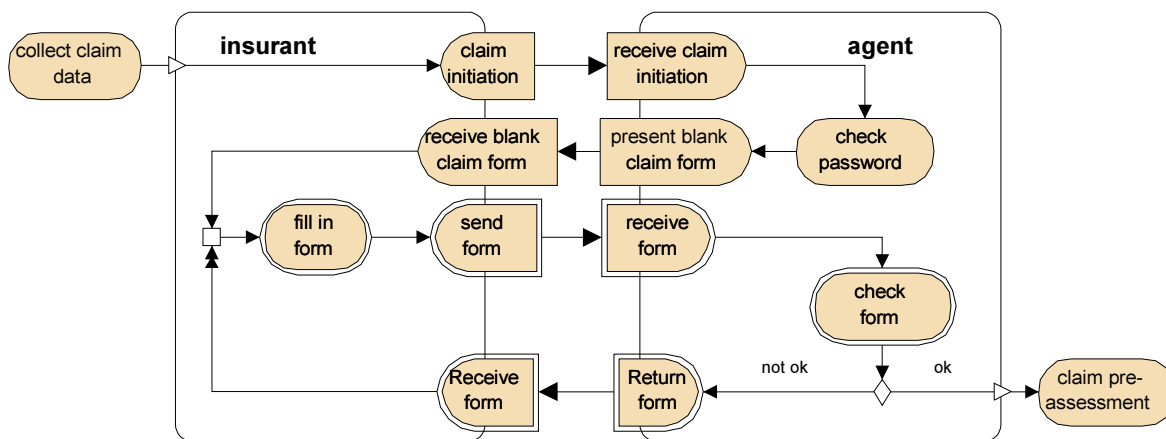


Figure 6 — Procedure for Claim submission.

Although the ideas have not completely crystallized so far, the coupling between the business and the technology cornerstones will be one of the most important innovations in RSD.

#### 4. Conclusion

The paper discusses an approach to modeling business processes in the situation where several organisations participate. The RSD methodology is a formal procedure that breaks the whole business process into several sub-processes each of which is handled by a particular participant. The procedure generates a set of transactions between those sub-processes. The procedure consists of three parts:

1. A Networked Enterprise model, which consists of a network definition of the participants and channels of communication between them and a model of the functions the participants are responsible for with the communication channels between them;
2. A process definition as a flow of activities and a map assigning particular activities to particular participants.
3. The detailed refinement in procedures which can be used for supporting system implementation.

Further work will be to study the coupling to UML. UML is a good language for software modeling, but lacks suitable concepts and semantics for modeling inter-organisation business processes. That is why we have developed a dedicated language for modeling business processes in an inter-organizational setting, featuring concepts such as trigger, action, interaction, transaction, transfer and causality.

We believe that the concepts introduced in this paper are relevant to modelling inter-organisational business processes. Our first experiences show that the concepts used in this paper are useful, but they need to be validated in further work. Currently we are conducting a number of pilot studies, e.g. in the flower industry, in the port of Rotterdam, and in the vegetables industry.

We thank Wil Janssen, Alko Smit, Bob Hulsebosch and the Giga Transaction Services team for their input.

**Biemans, F., W. Janssen, P. Oude Luttighuis, H. Schaffers, P. Van der Stappen, 'Business driven design of Telematics Services', Wognum, N., Thoben, K.-D., Pawar, K. S., (Eds.), Proceedings of ICE '99, the 5th international conference on concurrent engineering: The concurrent enterprise in operation, The Hague, The Netherlands, 1999. Nottingham, UK: Centre for Concurrent Enterprising of the University of Nottingham, 1999, pp. 337-344**

**Booch, G., J. Rumbaugh, 'Unified method for Object-Oriented Development', Rational Software Corporation, 1995**

**Davenport**, T., J.E. Short, 'The new industrial engineering: information technology and business process redesign', Sloan Management Review, Summer, 1990, pp. 309-330

**Eertink**, H., W. Janssen, P. Oude Luttighuis, W. Teeuw, C. Vissers, 'A business process design language', In: Wing, J. M., Woodcock, J., & Davies, J. (Eds.), FM'99 – Formal methods: World congress on formal methods in the development of computer systems, Toulouse, France, September 1999; Proceedings, Vol. I. Lecture notes in computer science: 1708. Berlin ; Heidelberg: Springer, 1999, pp. 76-95

**Janssen**, W., M. Steen, 'Rapid Service Development: an integral approach to e-business engineering', whitepaper, submitted to the workshop on Web Engineering WWW9

**Jacobsen** I., M. Christerson, P. J. G. Overgaard, 'Object oriented software engineering, a use case driven approach', Addison-Wesley, 1992

**Ould**, M.A. 'Business Processes: Modelling and analysis for re-engineering and improvement', John Wiley & Sons, Chichester, 1995

**Timmers**, P., 'Electronic Commerce, Strategies and Models for Business-to-Business Trading', John Wiley and Sons Ltd, England, 1999.

**Venkatraman**, N., "IT-enabled business transformation: from automation to business scope redefinition", Sloan Management Review, Fall 1995, pp. 32-42