2012

Using Enterprise Ontology as a basis for Requirements for Cross-Organizationally Usable Applications

Marien Krouwel
Capgemini / UA, marienkrouwel@gmail.com

Martin Op 'T Land
Capgemini NL / AMS / TU Delft, martin.optland@capgemini.com

USING ENTERPRISE ONTOLOGY AS A BASIS FOR REQUIREMENTS FOR CROSS-ORGANIZATIONALLY USABLE APPLICATIONS

Krouwel, Marien, marien.krouwel@capgemini.com
Capgemini Netherlands, P.O. Box 2575, 3500 GN Utrecht, the Netherlands;
University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium.

Op ‘t Land, Martin, martin.optland@capgemini.com
Capgemini Netherlands, P.O. Box 2575, 3500 GN Utrecht, the Netherlands;
Antwerp Management School, Sint-Jacobsmarkt 9-13, 2000 Antwerp;
Delft University of Technology, P.O. Box 5031, 2600 GA Delft, the Netherlands.

Abstract

Developing cross-organizationally usable applications is becoming increasingly important. However, actually re-using business applications has been hindered in practice often by implicit assumptions about organizational and IT implementation. We propose a method to gather requirements for cross-organizationally usable applications that, taking Design and Engineering Methodology for Organizations (DEMO) as starting point, a) is easy to communicate, b) can discern differences between the organizations, and c) has an attractive Return On Modeling Effort (ROME). When testing this method in a large real-life case study, it also appeared to be possible to a) systematically design the `unhappy' flow as well, b) define application architecture principles for supporting organizational flexibility, and c) systematically design screens based on DEMO models, irrespective of internal or external use. Finally the method greatly objectified the discussions between all stakeholders involved.

Keywords: Enterprise Ontology, requirements engineering, interaction design, cross-organizational, software flexibility, DEMO

1 INTRODUCTION

Developing cross-organizationally usable applications is becoming increasingly important. Especially in (international) cooperations, ranging from open-ended without any agreements on one side to one with a high degree of sector agreements and standards on the other side, re-using applications is becoming more and more popular. Being able to develop an application once and use it in several organizations is, obviously, one of the main benefits to be achieved.

Re-using applications among multiple organizations is a great concept in theory and has been quite successful in the context of inter-organizational reuse of technical infrastructure: modern operating systems, database management systems, component infrastructures, web servers, web browsers, etc., are examples of infrastructural applications that can be reused by organizations (Bosch, 2004). Re-using business applications, however, has been hindered in practice often by implicit assumptions about organizational and IT implementation. How often do we not see similar applications are developed multiple times, just because of small differences in the requirements? And how often do we not see organizations have to adapt to an implemented package, e.g., an ERP system, as it is not possible to adapt the package to meet the organization's specific needs?

We propose a method with DEMO (Dietz, 2006) as starting point to gather requirements for cross-organizationally usable applications. The method a) is easy to communicate, b) can discern differences between the organizations, and c) has an attractive Return On Modeling Effort (ROME). By using interaction designs to elicit the requirements and validate organization models, within a relatively short period, great insight can be gained in the similarities and differences between organizations and the consequences for supporting applications.

The method was tested in a large real-life case study at an International Public Private Organization (IPPO; 7,000 employees in 30 countries), which supports common concerns of National Offices granting patents. Currently, these National Offices have their own way of working, including their own supporting IT, loosely based on what once were copies of the same IT system: several offices have adapted the system in terms of business rules, supported processes, and integration with CMS and DMS. Goal of the project was to find the differences between three National Offices to see whether these offices could be supported by one IT system.

The remainder of this paper is structured as follows. The research design in section 2 outlines the problem statement, introduces the way of thinking and embeds this in a way of working. Next we describe the actual intervention, of which section 4 discusses the results. Finally, section 5 provides the conclusions, as well as directions for further research.

2 RESEARCH DESIGN

2.1 Problem statement

Our research program aims at investigating how to engineer applications for *comparable enterprises*. We define enterprise as a goal-oriented cooperative of people. Enterprises we consider comparable when offering the same kind(s) of products or services. For example, we consider health care insurance companies comparable, as they all provide health care insurance, although the exact products delivered may vary. Let us introduce the Generic System Development Process (GSDP) (Dietz, 2008) in order to explain this in more detail.

By system development is meant the bringing about of a new system or of changes to an existing system. The system development process comprises all activities that have to be performed to arrive at an implemented system. In any system development process two systems are involved: the Object System (OS), the system to be developed, and the Using System (US), the system that will use the functions or services of the OS. The complete process, as shown in Fig. 1, consists of three phases: design, engineering, and implementation.

The design phase is further split up into *function design* and *construction design*. Function design starts from the construction of the US and ends with the function of the OS. The result is a functional (black box, see Dietz (2006; 2008)) model clarifying the behavior of the OS in terms of (functional) relationships between input and output of the OS. The functional model is mainly driven by *functional requirements*. Construction design starts with the specified function of the OS and ends with the construction of the OS. Construction design bridges the mental gap between function and construction, establishing a correspondence between systems of different categories. Construction design is driven by *construction requirements*, also known as `non-functionals'.

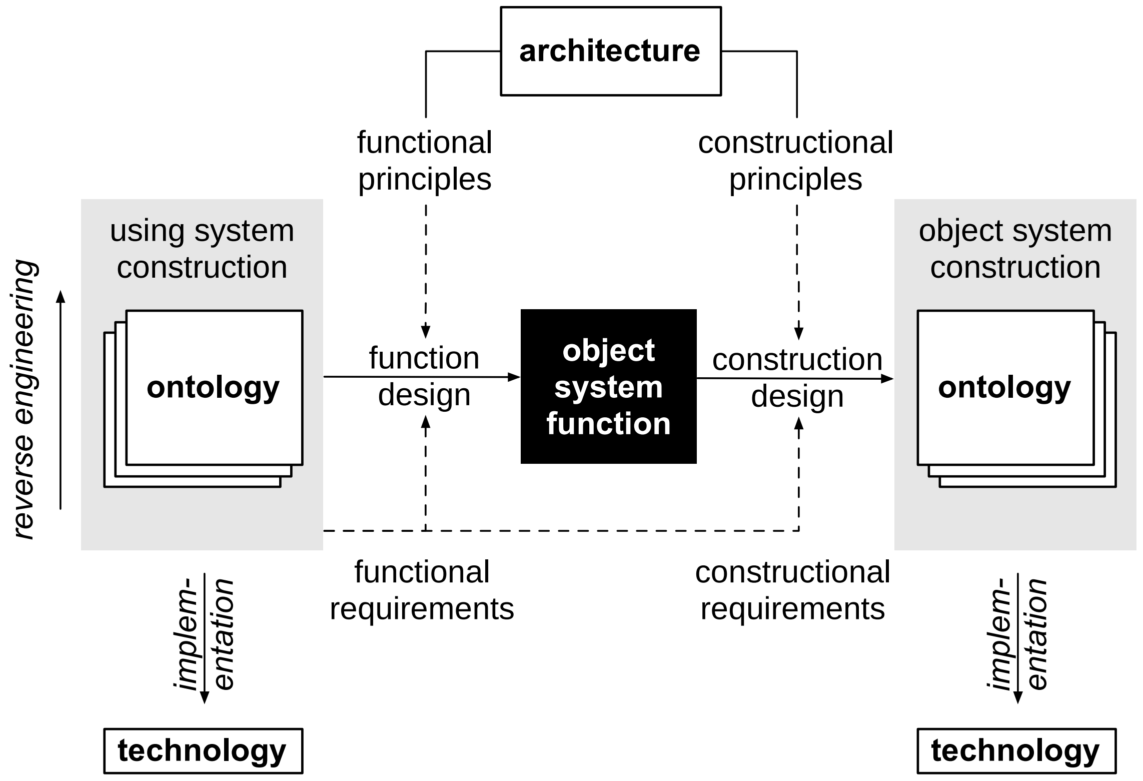


Figure 1. The Generic System Development Process (Dietz, 2008)

Engineering is the process in which a number of subsequently more detailed constructional (white box, see Dietz (2006; 2008)) models are produced such that every model is fully derivable from the previous one(s) and the available requirements. Engineering starts from the *ontological* model, a model of the construction that is completely independent of the way in which it is implemented, and ends with the implementation model. By *implementation* is understood the assignment of technological means to the elements in the implementation model, so that the system can be put into operation.

In order to balance and safeguard the interests, concerns and objectives of all stakeholders, broader than just the parties involved in US and OS, the design freedom of a class of object systems has to be restricted. Following Dietz (2008), we define *architecture 1)* conceptually as a normative restriction of design freedom and *2)* operationally as a consistent and coherent set of design principles that embody general requirements, where these general requirements hold for a class of systems[1]. Expected benefits of such a governing based on architecture are a/o improved integration, adaptability, and agility.

In terms of GSDP, we can now operationally define enterprises as comparable when their businesses share the same function model. Different organizations sharing the same function model might have different ontologies. Even when they share the same ontology, the implementations can still differ.

[1] See Dietz (2008) for a comparison with this definition to other definitions, such as IEEE 1471-2000

Our ideal would be that one application could support comparable enterprises, possibly with different ontologies and possibly with different implementations.

So, we are looking for a way to gather requirements for cross-organizationally usable applications, which *a)* is easy to communicate, *b)* can discern differences on the ontological and implementation level, and *c)* with an attractive ROME.

Significant improvements in communication for eliciting requirements can be achieved by the use of prototyping (Nuseibeh and Easterbrook, 2000; Mannio and Nikula, 2001). Often, the requirements found are static, reflecting the current implementation of the organization and not the ontology. Therefore this strategy can endanger future changes in the organization and re-use in other organizations.

Op 't Land et al (2009) showed that within six weeks the essentials of the Cargo business processes of the two merging parts of Air France and KLM could be clarified in a way which allowed comparison of different organizational and IT implementation of these processes. However, the method applied – using DEMO's Construction Model (CM), event traces and organization and IT mapping – did not yet design new applications.

Mulder (2006) has captured the essentials of 22 complaints boards of the Dutch Stichting Geschillencommissies Consumentenzaken (Dispute Settlement Boards for Consumer Affairs) using DEMO. It was found the 22 boards share the same ontology, but have different implementations. Based on the results, a workflow system was implemented, based on the ontology while also supporting the implementation differences. Mulder however does not show how the implementation differences were captured, nor does he show how the gap from business analysis to information system engineering was bridged.

As DEMO has shown to offer a quick way to find ontological similarities and differences, and to clearly distinguish those from implementation, we think DEMO can be used for gathering requirements for cross-organizationally usable applications. However, when using DEMO, two areas of attention arise:

- as DEMO only focuses on the ontology of an enterprise, we will need additional ways to find and express implementation details;
- studies on the adoption of DEMO show that attention is needed to communicate the models with individuals at any level in the organization (Sattari Khavas, 2010).

So we are looking for a method that combines the ability to discern the ontology and implementation and the ROME of DEMO, and the communicability and detail level of prototyping.

2.2 Proposed way of thinking

As DEMO has shown to have a good ROME for identifying processes and essential differences (Mulder, 2006; Op 't Land, 2007; Op 't Land et al, 2009), DEMO will be used as starting point of our execution. From the Performance in Social Action (PSI- or ψ-) theory, DEMO's underlying theory, the following axioms are relevant for our research.

- Operation Axiom: People in an organization (subjects) perform two kinds of acts: *Production acts* (P-acts) and *Coordination acts* (C-acts). We abstract from the subjects in order to concentrate on the different actor roles they fulfill. An *actor* is a subject fulfilling an actor role. By performing P-acts they contribute to the purpose or the mission of the enterprise. By performing C-acts they enter into and comply with mutual commitments about P-acts.
- Transaction Axiom: C-acts and P-acts occur in generic socionomic patterns, called *transactions*. These patterns always involve two actor roles (initiator and executor) and are aimed at achieving a particular result. The basis transaction pattern for one transaction consists of five steps, namely request, promise, execute, state and accept. In the request step the initiator of the transaction requests the production of a P-fact, in the promise step the executor of the transaction promises to produce the P-fact, in the execution step (P-act) the executor produces the P-fact, in the state step the executor presents the P-fact to the initiator and in the accept step the initiator accepts the P-

fact. The initiator does not know about the P-fact being produced until the executor has presented (stated) the result. Apart from these 4 types of `success' C-acts, 7 other types of C-acts exists (Table 1). A C-act can be performed in three different ways: explicitly (by some observable act), implicitly (performing another C-act also counts for performing the C-act), or tacitly (there is no observable act that counts as performing the C-act).

- Distinction Axiom: three distinct human abilities play a role in the operation of actors, called *performa*, *informa* and *forma*. The forma ability concerns the being able to handle data and documents, so e.g. to copy, transport and store documents. The informa ability regards intellectual capacity, the ability to reason, to compute or derive new facts from existing ones. The performa ability concerns the ability of human beings to produce original new things, i.e. facts that cannot be derived from existing facts. Examples of those facts are decisions and judgments.

success	problems	canceling
request	quit	cancel
promise	decline	refuse
state	stop	allow
accept	reject	

Table 1. The 11 types of coordination acts (C-acts).

DEMO states that a complete enterprise ontology is a model of the B-organization, consisting of four aspect models:

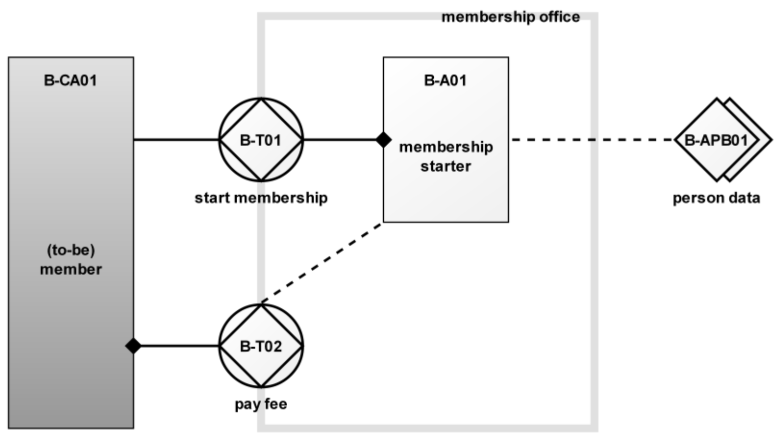
- The CM specifies the composition, the environment and the structure of the organization. It contains the identified *transaction types*, the associated *actor roles* as well as the links to *production* or *coordination banks* where the production and coordination facts are stored.
- The Process Model (PM) details each single transaction type of the CM according to the universal transaction pattern. In addition, the initiating and waiting relationships between transactions are shown.
- The Action Model (AM) specifies the imperatively formulated *business rules* that serve as guidelines for the actors in dealing with their agenda. It contains one or more action rules for every agendum type.
- The Fact Model (FM) specifies the *object classes*, the *fact types* and the declarative formulations of the *business rules*.

We will now briefly introduce the required concepts and the diagram of the DEMO CM, by using a fictitious membership office (Fig. 2). A member-to-be (B-CA01) can request a new membership at the membership office. Such a request will be dealt with by actor B-A01 (called executor, marked by a small black diamond on the edge of the actor role box). The execution of transaction T01 results in a fact in reality, viz., a new membership. In executing transaction T01, actor B-A01 needs to know about ongoing and past transactions T02 (payments); this information link between actor B-A01 and (the fact bank of) transaction type T02 is indicated by a dashed line. In the fact bank of each transaction we find both the production facts (e.g., membership has been started) and the coordination facts (e.g., `requested', `promised', `stated', `accepted') of the instances of that transaction type.

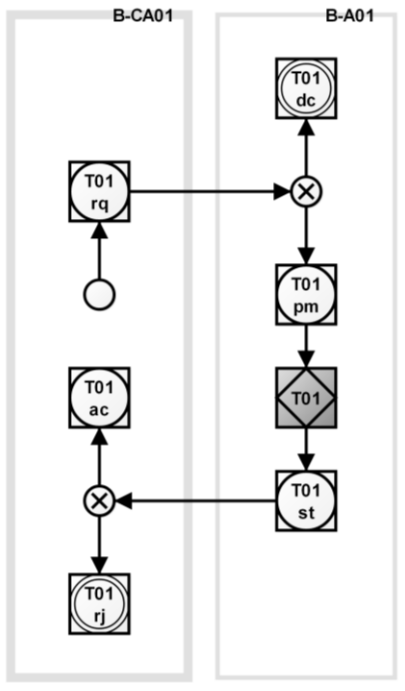
2.3 Proposed way of working

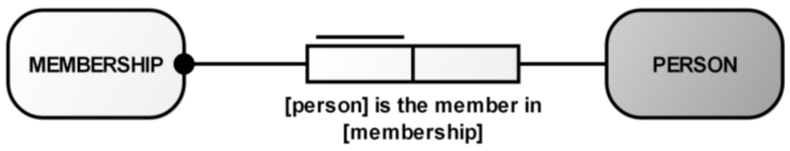
We will now elaborate a way of working for gathering requirements for cross-organizationally usable applications, building on the way of thinking outlined above. The way of working should result in a validated set of business and application requirements, making explicit the similarities and differences between the enterprises. Main principles behind our way of working were:

- open discussion;
- involvement of employees from different positions (management, subject matter expert, legal);
- the models created are understandable and/or falsifiable by understandable views for everyone.



(a) DEMO CM of a membership office





(c) DEMO FM of a membership office

(b) DEMO PM of T01 of a membership office

Figure 2. DEMO models of a fictitious membership office

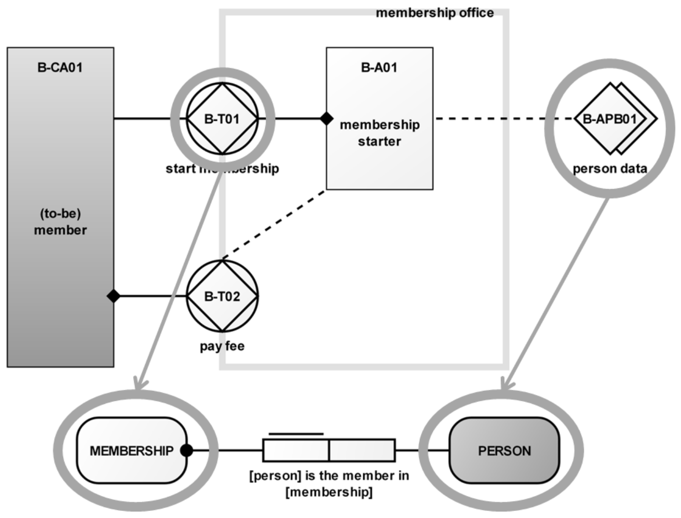
First, a draft DEMO CM, PM, and FM is created using existing materials such as process models and IT implementation (reverse engineering). Because of the reasons mentioned in 2.1, we will use mock-up screens to communicate and validate parts of the DEMO models. We will create one draft screen for each C- act (Dietz, 2003; Huysmans, Bellens, van Nuffel and Ven, 2010; Krouwel and Op ‘t Land, 2011). As stated by Dietz in his CRISP-model (2006, p.128), the being executed of a P-act can only be known by performing the corresponding state act; so we will not create a screen for performing P-acts.

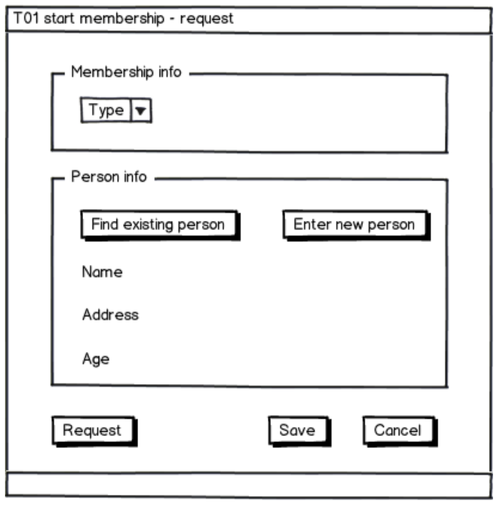
On a screen information can be entered, information will be needed, and a decision could be taken, depending on the act and its related business rules, information links and objects instantiated. Let us explain how to determine the contents of a screen by two examples using the fictitious membership office in Fig. 2.

Suppose a transaction of type T01 is being requested. It is of no relevance whether it is the member-to-be itself that enters the data on some (on line) form, or whether it is an administrative employee who enters the data on behalf of the member-to-be. The data needed for the membership concerns some personal data (Fig. 3a). That data must be entered if the person is not yet known in the system (Fig. 3b) – when the membership office uses the personal data as known by the government (residential registration office), we can safely regard every person to be known.

After being requested, actor B-A01 now needs to decide on a promise or decline. In order to do so, it needs to evaluate a business rule. For that, B-A01 will need information about the requested membership but also about one or more transactions of type T03 (Fig. 4). Note that we added buttons for saving and canceling entered data, enabling interruption of entering data while not yet performing the act.

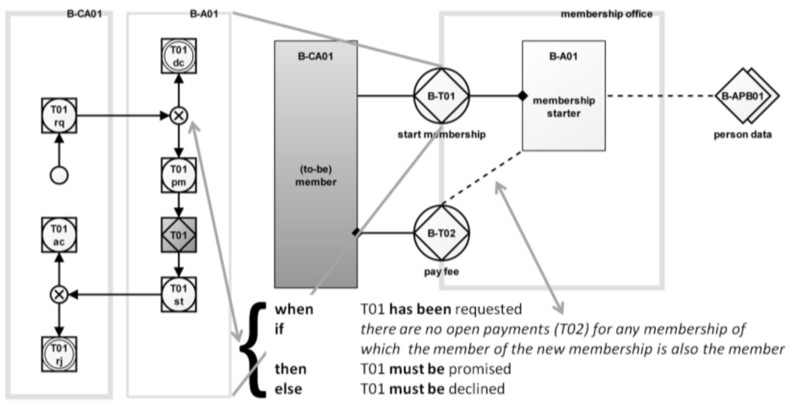
The draft models and mock-up screens will be used for validation of requirements while also enriching them with implementation details, using the following steps, performed for each business process, consisting of a causally or conditionally cohering set of transactions, in the CM.



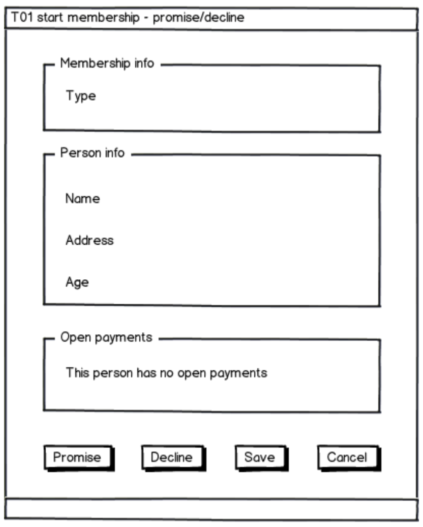


(a) Links between DEMO CM and FM for T01/rq *(b) Resulting mock-up screen for T01/rq*

Figure 3. Relevant DEMO models and resulting mock-up screen for T01/rq



when T01 **has been** requested
if *there are no open payments (T02) for any membership of which the member of the new membership is also the member*
then T01 **must be** promised
else T01 **must be** declined



This person has no open payments

(a) Links between DEMO CM, PM and AM for T01/pm *(b) Resulting mock-up screen for T01/pm*

Figure 4. Relevant DEMO models and resulting mock-up screen for T01/pm

Step 1: Identify and validate the essential activities (transactions) necessary for bringing about the business process. As the CM is independent of working order, the way in which it is executed by human beings, and the way in which it is supported by IT, it should be possible to find a limited set of essential activities needed for executing the business process.

Step 2: Identify and validate the essential steps (acts) and the strong dependencies, i.e., the minimal order of working, by discussing the related part of the PM. As the transaction pattern provides not only the happy flow, the requirements also include the `unhappy' flow.

Step 3: For each act, determine whether it is performed explicitly, implicitly, or tacitly. For explicit acts that have to be supported by an application, the application must provide one or more screens for showing and entering data. For other acts, no screens will be necessary.

Step 4: Discuss the screen flow in order to validate the links in the PM. This will also result in a preferred order of working, consciously restricting the freedom still left by dependencies between acts as denoted in DEMO's PM (Step 2).

Step 5: Discuss the screen contents:

- o which decision is made;
- o what data is entered;
- o what data is needed.

The requirement for the data entered should reflect by the FM. Some data might be reference data, e.g., a list of countries, fees, etc. It's worthwhile to discuss whether it should be possible to manage these reference data with additional screens or (business) activities. With the requirement for the data needed, the information links in the CM can be validated. The requirement for the decision made ensures the screen is used for the right step (act); the decision should reflect the paths in the transaction pattern – i.e., after T01 has been requested, an employer sees some information, adds some information and decides `promise' or `decline'.

Step 6: Discuss business rules per screen applicable. This will provide the action rules (AM) and ensures the data needed for making a decision is present.

Step 7: For each screen, discuss which department or functionary type enters the data. This will make explicit how authorization, delegation, and actor role assignment is arranged.

At the end, a consistency check between the models should be performed. When activities are re-used, e.g., payment transactions, variations could have been introduced in the requirements for the data entered, needed, or the actor role assignment, depending on the business process at hand. The audience should be confronted with these differences, making explicit why these differences are there or concluding there are in fact no differences and the execution should be the same, no matter which triggered the activity.

Although it might be practicable to perform the steps at every enterprise individually, one could also choose to perform the steps in one session involving employees of all enterprises at once.

3 THE APPROACH IN PRACTICE

3.1 Background

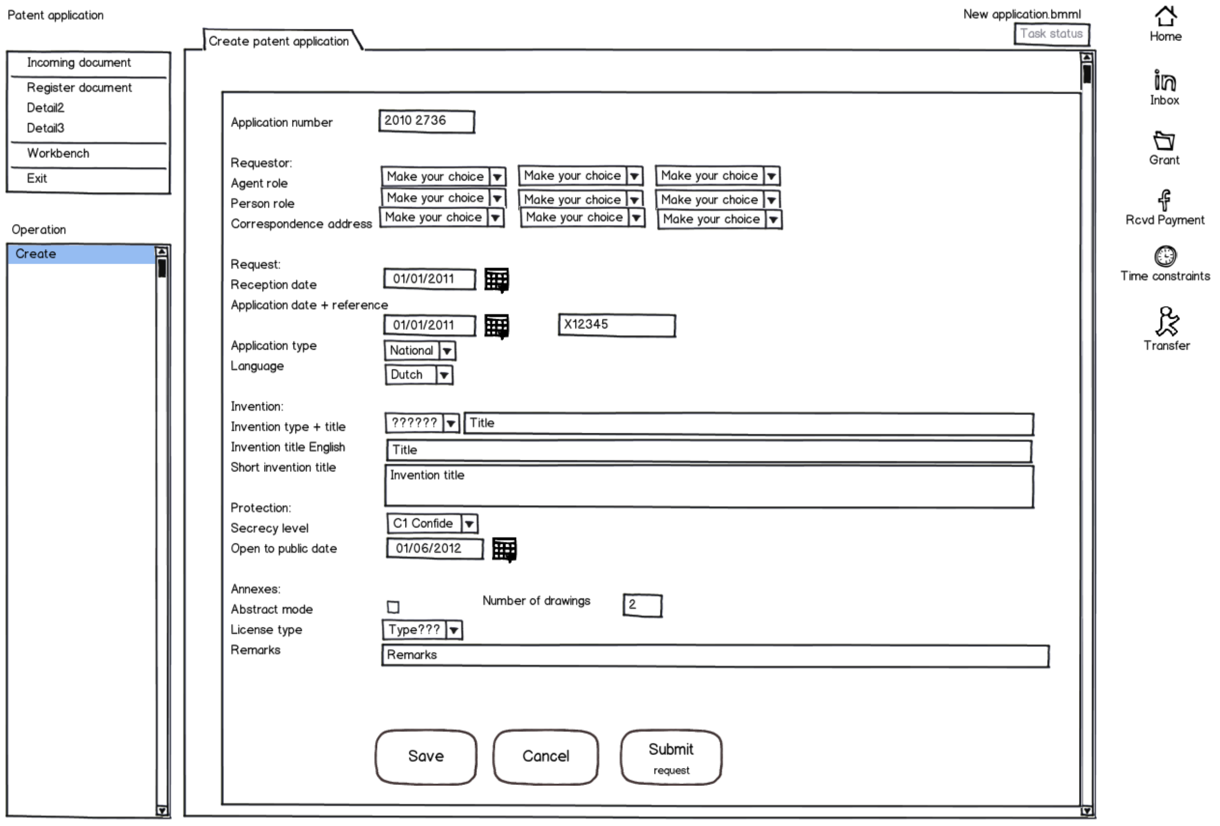
Directed by IPPO, from 2002 to 2005 an application was developed for the National Offices for supporting its business. This application was highly adaptable in order to conform to national laws. Every country configured its own version.

As the National Offices sometimes hand-over requests to other offices (for obtaining a International patent, operationalized as several patents in a set of countries), in 2009 the idea came up to connect the systems to exchange data. However, in 2010 an independent feasibility study identified 54 problems on both the business – e.g., missing functionality, difficult interaction design – and technical level – e.g., failure, bad database design, missing documentation. These problems were mainly caused already in the design phase of the system. First, the design was based on an older version reflecting a limited set of requirements. Secondly, the National Offices were not allowed to provide input for the design phase. And thirdly, as a consequence of the bad design and the degree of customization, every National Office could create its own version of the system, supplemented by other systems – e.g., DMS, CMS.

In order to identify the needs of the offices, a study was performed to identify the requirements for three National Offices, making explicit the similarities and differences between them. The focus was on identifying the as-is situation at the business level, i.e., the main processes and the way in which they should be supported. Identifying optimization opportunities at the business level (to-be) were out of scope.

3.2 Preparation

In the first phase of the project in each country a one-day workshop was held to create the CM and interviews were held to create the PM, resulting in three similar CMs and PMs. A FM was created by reverse engineering the existing database model, which was similar in all three countries. For every C-act a mock-up screen (Fig. 5) was created using a rapid wire-framing tool.



Application number: 2010 2736

Requestor:
Agent role: Make your choice | Make your choice | Make your choice
Person role: Make your choice | Make your choice | Make your choice
Correspondence address: Make your choice | Make your choice | Make your choice

Request:
Reception date: 01/01/2011
Application date + reference: 01/01/2011 X12345
Application type: National
Language: Dutch

Invention:
Invention type + title: ?????? Title
Invention title English: Title
Short invention title: Invention title

Protection:
Secrecy level: C1 Confide
Open to public date: 01/06/2012

Annexes:
Abstract mode: ☐ Number of drawings: 2
License type: Type???
Remarks: Remarks

Figure 5. Example mockup screen for new application

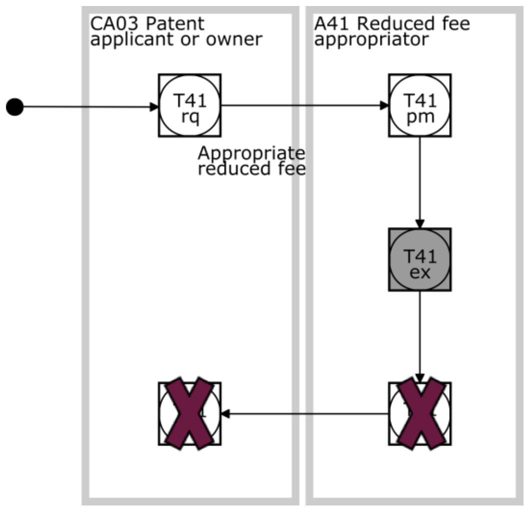
3.3 Workshops

In each country, workshops were organized in which the steps mentioned above were executed. For each step, large posters and slides were used in an interactive way (Fig. 6a). Per country, in three days all business processes were discussed elaborately with domain experts, IT directors, administrative, financial, and legal officers, and the sponsors of the project. During the workshops, two team members made notes of the discussed topics, including relevant implementation details. After each 3-day session, the results were summarized and the notes were added to the DEMO models. To minimize interpretation issues, the notes and models were sent to the National Offices for providing feedback. In the second and third workshops insights gained from the previous one(s) were taken along and discussed, making the differences between the countries explicit. After finishing the workshops in the three countries, we started identifying the similarities and differences between the DEMO models and other materials. A set of DEMO models which fitted all three National Offices was created, detailing which activities were (not) applicable per country.

4 RESULTS (CASE LEVEL)

In this section we will give the results on the case level. Per step we will discuss the similarities and differences found as well as some other findings.

- A CM was created consisting of 70 transactions and 21 information links (partly shown in Fig. 7). Although there are almost no international regulations about the granting of patents, the models are similar to a large extent. In some countries some business processes or individual transactions simply did not exist (indicated by a gray diamond in Fig. 7). For example, in country A *check military interest* (T04) is mandatory by law, in country C it's optional, depending on the content of the patent requested, while in country B military interest is never checked. Also, in country B neither substantive examination nor a search is performed, as a patent is always granted.
- Although the preconditions for any activity are the same for every country, the working order differed a lot. For example, while one country performed the search at the beginning to prevent unnecessary work, another country performs it at the end because the costs are high. This implies that different screen flows better fit the needs of the different organizations.
- Whether an act is performed implicitly or explicitly is to a large extent the same for every country (Fig. 6b). However, differences were also detected, especially in the transactions that are both initiated and executed within the organization boundary. For example, in country A it is not possible to *decline* any internally initiated transaction. Also, while in country A and B an implicit *accept* holds until an explicit *reject* is performed, in country C the *accept* has to be performed explicitly, otherwise a *reject* is assumed. This implies not every screen is used in every country.
- The data entered at each screen (if applicable) for every country is the same. Also, the information needed (indicated by information links in the CM) is the same. The screens were very helpful as they brought the models alive.
- The business rules were not made explicit and no conclusions can be drawn on that aspect.
- It was discovered the functionary type to actor role mapping differed a lot per office. Discussions about the fulfillment of actor roles for one country resulted in introducing a new functionary type, fulfilling actor role B-A01 as coordinator of the grant process; by fulfilling this role explicitly, the control in coordination increased, shortening throughput times.
- As already indicated by some examples above, the sharing of information of other countries helped offices and the essential processes and activities, as well as finding new business improvement opportunities.



(a) Example of the capturing of implicit and explicit acts

Transaction	C-act	Country		
		A	B	C
T01 – grant patent	rq	E	E	E
	pm	E	E	E
	st	I	E	E
	ac	I	I	I
	dc	E	E	E
	rj	E	E	E
T07 – examine formally	rq	E	E	E
	pm	I	I	I
	st	E	E	E
	ac	I	I	E
	dc	-	E	E
	rj	E	E	I

(b) Part of results of implicit (I)/explicit (E) analysis

Figure 6. Way of capturing and results of implicit/explicit analysis

Based on the findings outlined above, two (application) architecture principles were defined:

- The IT system should be able to support different orders of working (as a result of Step 4);
- The IT system should be able to deal with different actor role to functionary type mappings (as a result of Step 7).

As the method helped in objectifying the discussions, the results are supported by all offices. We estimate the differences between the organizations to be that small, that it is possible to create one application to support all National Offices.

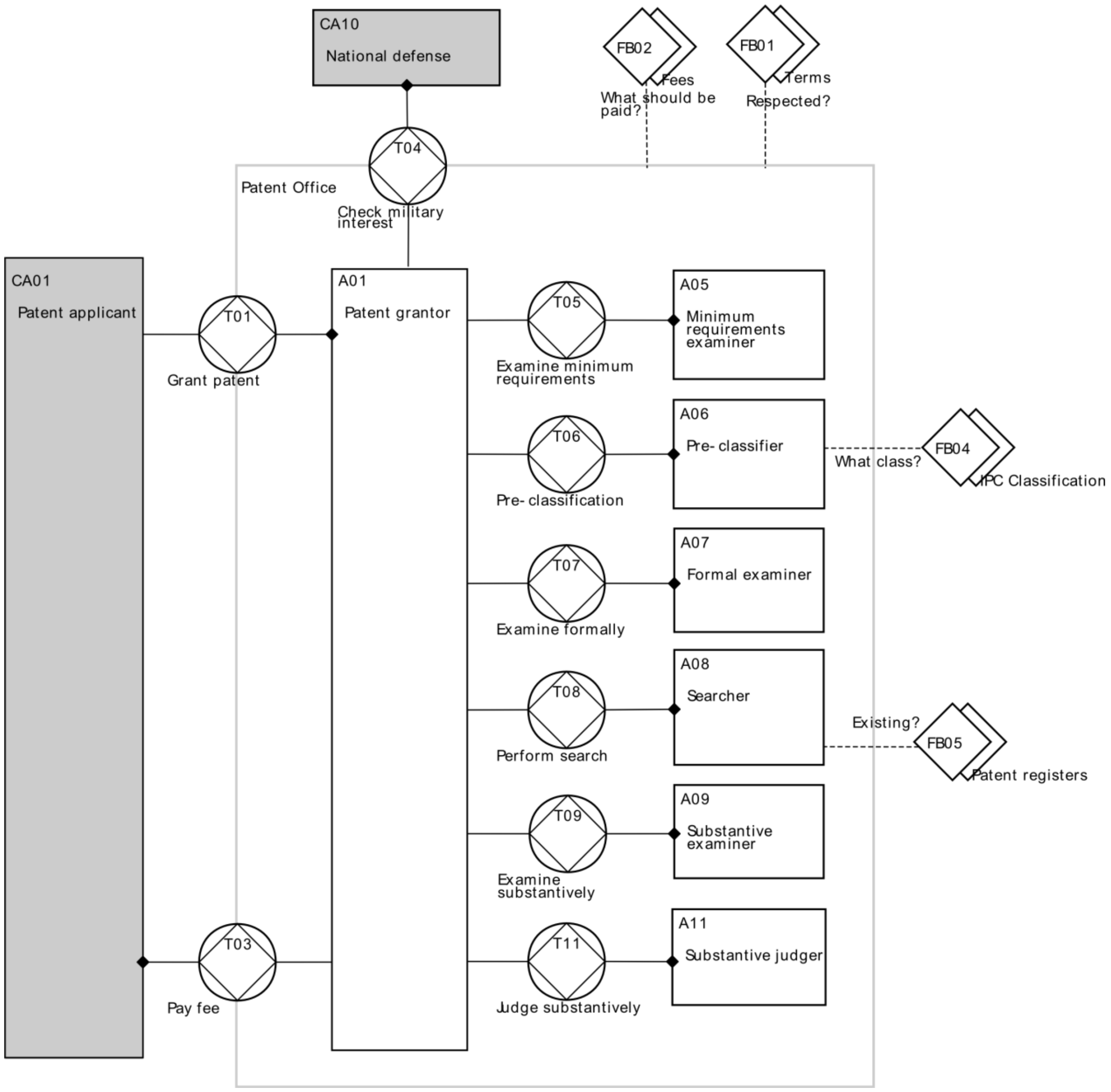


Figure 6. DEMO CM of patent granting

5 CONCLUSION AND DISCUSSION (RESEARCH LEVEL)

We were looking for a method to gather requirements for cross-organizationally usable applications, which a) is easy to communicate, b) can discern differences between the organizations, and c) with an attractive ROME. In the workshops, interaction designs and mock-up screens were used for visualizing the DEMO models, representing the function of the system. As stated by the participants, the use of interactions design and mock-up was a great way of communicating and validating the, sometimes rather abstract, models, as they `brought alive the system'.

The use of DEMO has shown its benefits in finding essential differences, often resulting from regulations applicable, e.g., the military interest check enforced by law in country A but not applicable in country B. The implicit/explicit and actor role assignment analysis, as well as discussions about the screen flows, have offered great insights in the organizations, at both the ontological and implementation level. DEMO helped to distinguish the ontological differences from differences in implementation details.

To achieve the results mentioned, an investment of 200 man days (128 by Capgemini, 72 by involvement of IPPO and National Offices) was needed.

In applying our method, we also found other interesting benefits. First, as the DEMO transaction pattern includes the `unhappy' flow, it is automatically a systematic part of the requirements. Second, we were able to define (application) architecture principles. Third, we have shown to be able to systematically design screens and screen flows, based on DEMO models. Finally the method greatly objectified the discussions between all stakeholders involved, which confirms Op ‘t Land et al (2009).

As DEMO models do not reflect implementation, in designing screens and screen flows inevitably implementation details are introduced. For example, we considered external fact banks to be part of the application as well. One could also decide to use external data sources such as the database of the residential registration office. It is interesting to see how application architecture can support such changes.

Using the DEMO Distinction Axiom, it is now possible to design the screens once, independent of the user of the screen, i.e., whether the user is an internal employee or an external customer or supplier. The screens themselves are actually used for performing an (in)formative act, fully determined by the ontological act. The performer of the (in)formative act can but need not be the same as the performer of the ontological. For example, the screen used for requesting a membership (Fig. 3b) is for entering the data necessary for the membership as determined in the ontological act, i.e., member name, birth day, etc. The same screen can either be used by a member-to-be requesting the membership on line, or by an employee entering the data after a letter of request has been received, or by the employee listening to the member-to-be passing by in the office.

Our case demonstrates that comparable enterprises, such as National Offices sharing the same function model, can have different ontological models, as the different enterprises are subject to different national laws and regulations. This also implies that DEMO models will change as these laws and regulations change over time. Since ontological changes often have a deep impact on supporting applications, it is important to increase the insight in other aspects that can influence the ontological models of an enterprise.

Our case also demonstrates that even when enterprises share the same ontological model, their organizational implementation can differ significantly, also over time. We have shown two (application) architecture principles that contribute to the possibility that one application can support different organizational implementations of the same ontology. In computer science, principles have been formulated to ensure flexibility at the application level, with regard to an anticipated set of changes (Mannaert and Verelst, 2009). More research is needed to find additional principles that are necessary and even sufficient, to ensure that differences and changes in (specific) organizational implementations for the same (generic) ontological model can be supported by one IT system, at design-time or at run-time.

To embed this method for requirements engineering in a complete system development cycle, both linear and agile/SCRUM case studies are needed, using earlier experiences in applying DEMO as starting point for system development in an agile environment (Op ‘t Land et al, 2011).

Finally, we want to encourage researchers and practitioners to use or incorporate other modeling techniques in the method.

References

Bosch, J.: On the development of software product-family components. In: Proceedings of the 3rd International Conference on Software Product Lines (SPLC), Springer LNCS (2004)

Dietz, J.: Deriving use cases from business process models. In Song, I.Y., Liddle, S., Ling, T.W., Scheuermann, P., eds.: Conceptual Modeling – ER 2003. Volume 2813 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2003) 131-143

Dietz, J.L.G.: Enterprise Ontology – Theory and methodology. Springer (2006)

Dietz, J.L.G.: Architecture: Building strategy into design. Sdu Uitgevers bv, The Hague, The Netherlands (2008)

Huysmans, P., Bellens, D., van Nuffel, D., Ven, K.: Aligning the Constructs of Enterprise Ontology and Normalized Systems. In Aalst, Will and Mylopoulos, John and Sadeh, Norman M. and Shaw, Michael J. and Szyperski, Clemens and Albani, Antonia and Dietz, Jan L. G., ed.: Sixth International CIAO! Workshop on the Fifth International Conference on Design Science Research in Information Systems and Technology (DESRIST 2010), Springer Berlin Heidelberg (June 2010)

Krouwel, M.R., Op 't Land, M.: Combining DEMO and Normalized Systems for Developing Agile Enterprise Information Systems. In Albani, A., Dietz, J.L.G., Verelst, J., eds.: Lecture Notes in Business Information Processing. Volume 79., Springer-Verlag Berlin Heidelberg (2011) 31-45

Mannaert, H., Verelst, J.: Normalized systems: re-creating information technology based on laws for software evolvability. Koppa, Kermt, Belgium (2009)

Mannio, M., Nikula, U.: Requirements elicitation using a combination of prototypes and scenarios. In: WER. (2001) 283-296

Mulder, J.B.F.: Rapid Enterprise Design. PhD thesis, Delft University of Technology (2006)

Nuseibeh, B., Easterbrook, S.: Requirements engineering: a roadmap. In: Proceedings of the Conference on The Future of Software Engineering. ICSE '00, New York, NY, USA, ACM (2000) 35-46

Op 't Land, M.: Toward Evidence Based Splitting of Organizations. In Ralyté, J., Brinkkemper, S., Henderson-Sellers, B., eds.: Proceedings of the IFIP WG 8.1 Working Conference on Situational Method Engineering, Fundamentals and Experiences. Volume IFIP 244., Geneva, Switzerland, Springer-Verlag Berlin Heidelberg (2007) 328-342

Op ’t Land, M., Zwitzer, H., Ensink, P. and Lebel, Q. Towards a Fast Enterprise Ontology Based Method for Post Merger Integration. In Proceedings of the 2009 ACM symposium on Applied Computing (SAC- ACM2009) (2009) 245–252

Op 't Land, M., Krouwel, M.R., van Dipten, E., Verelst, J.: Exploring Normalized Systems Potential for Dutch MoD's Agility – A Proof of Concept on Flexibility, Time-to-Market, Productivity and Quality. In Harmsen, F., Grahlmann, K., Proper, E., eds.: PRET. Volume 89 of Lecture Notes in Business Information Processing., Springer (2011) 110-121

Sattari Khavas, S.: The adoption of DEMO in practice. Master's thesis, TU Delft (2010)