HELSINKI SCHOOL OF ECONOMICS (HSE)
Department of Management

A STANDARD DATA MODEL FOR PRODUCT DEVELOPMENT DOCUMENTS

B2B Integration of Document Management Systems Using RosettaNet

Information Systems Science
Master's thesis
Katrine Jokinen 24546-1
Fall 2003

Approved by the Head of the Department of Management _____/_____ 200__ , and

awarded the grade _____

_____

ABSTRACT

Companies conduct more and more product development in collaborative networks. This enables them to concentrate on their core competencies. The collaboration introduces a need for new processes and information technology tools to support it. The companies need to share product design data and project management documents in their product development networks.

This thesis provides one solution to facilitate the collaboration. This is done by automating the exchange of product development documents through document systems integration. Only exchanging documents would not solve the problem, though. In addition, document metadata (information about the document) has to be sent to the partner. Otherwise it would be impossible to know who needs the document, and what for.

RosettaNet is an independent, non-profit consortium of more than 500 information technology, electronic components, semiconductor manufacturing, and telecommunications companies. It works to collaboratively create, implement and promote open Internet-based e-business process standards for the electronic sharing of business information. Currently the standard is mainly used in the order-delivery process and forecasting. Standards for collaborative engineering processes have been planned, but are not released yet, and cannot thus be utilized.

This research analyzes three case companies' internal data models and four document management systems' data models to find the common concepts in them. Then these concepts are mapped with RosettaNet concepts to find what the use of the standard requires. These common concepts form a standard data model, which provides a common way to describe the documents exchanged in a product development network. This way the companies will not have to define the exchanged metadata separately with each one of their partners, thus making it easier and faster to add new companies to the collaboration network.

The research also includes building a prototype, which shows that document management systems integration can be done with RosettaNet.

Keywords: document management, product data management, metadata, data model, information systems integration, RosettaNet

PREFACE

This thesis was written as part of the NetData research project, which belongs to the Product Data Management Group (PDMG) of the Software Business and Engineering Institute (SoberIT) at the Helsinki University of Technology (HUT). The project was partially funded by the National Technology Agency (TEKES).

I would like to thank the project manager of NetData, Jukka Borgman, who has taken his time to read what I have written from the first plan to the final version of this thesis. He has provided me with feedback, new ideas, encouragement, - and push when needed. I also want to express my gratitude to the instructor of the thesis, professor Tomi Dahlberg, and researcher Paavo Kotinurmi for their constructive comments and suggestions.

The prototype was implemented partly as student assignments in programming and software project courses at HUT. I would like to thank Elina Jormanainen, Heli Juntunen, Kalle Korpijoki, Jaakko Kotimäki, Harri Pylkkänen, Samuli Ruuskanen, Nikodemus Siivola, Tony Söderudd, and Jarkko Viinamäki for their effort in the implementation. I would also like to thank Hannu Laesvuori and Mikko Kaila from NetData staff and researcher Hannu Peltonen for their help with the implementation.

Espoo, August 22nd 2003

Katrine Jokinen

TABLE OF CONTENTS

ABBREVIATIONS

API	Application Programming Interface
AML	Approved Manufacturer List
B2B	Business-to-Business
BOM	Bill of Material
CAD	Computer Aided Design
COM	Component Object Model
DMS	Document Management System
DTD	Document Type Definition
EAI	Enterprise Application Integration
ebXML	Electronic Business using XML
EC	Engineering Change
ECO	Engineering Change Order
ECR	Engineering Change Request
EDI	Electronic Data Interchange
EDM	Electronic Document Management
EDMS	Engineering Data Management System
EMS	Electronics Manufacturing Services
ERP	Enterprise Resource Planning
FTP	File Transfer Protocol
GTIN	Global Trade Identification Number
HTTP	HyperText Transfer Protocol
HUT	Helsinki University of Technology
ISBN	International Standard Book Number
ISSN	International Standard Serial Number
IT	Information Technology
JPG	Joint Photographic Experts Group
ODA	Open Document Architecture
ODIF	Open Document Interchange Format
ODM	Original Design Manufacturing
ODMA	Open Document Management API
OEM	Original Equipment Manufacturer

PINJA	Product Data Management INterface in JAva
PIP	(RosettaNet) Partner Interface Process
PD	Product Development
PDF	Portable Document Format
PDM	Product Data Management
PDMG	Product Data Management Group
PDMS	Product Data Management System
PDX	Product Data eXchange
RNBD	RosettaNet Business Dictionary
RNIF	RosettaNet Implementation Framework
RNTD	RosettaNet Technical Dictionary
SGML	Standard Generalized Markup Language
SOAP	Simple Object Access Protocol
TPA	Trading Partner Agreement
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
WWW	World Wide Web
XML	eXtensible Markup Language
XSLT	eXtensible Sylesheet Language Transformation

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Product life cycles of consumer electronics products get shorter and the products more complicated. As a consequence, the lead-time (or time-to-market) in product development (PD) projects becomes more important. Companies want to concentrate on their core competences in PD; therefore, there is a trend to conduct it in collaborative networks. One example of this is electronics manufacturing services company Elcoteq's recent acquisition of telematics products company Benefon's research and development operations in order to enable Elcoteq to offer technology and PD services and original design manufacturing (ODM) services to its customers (Benefon 2002). The same applies for mechanical engineering industry: Metso Paper has established partner networks for the development of paper machines (Tekniikka & Talous 2003, 17).

The collaboration introduces new problems, though: how to efficiently work together with the partner companies? There is a need for new processes and information technology tools to support the collaboration. The companies need to share product design data and project management documents in their PD networks.

1.1 BACKGROUND

Borgman & Sulonen (2003) have studied preliminary design data exchange in four PD projects in one company network. They found lack of inter-company transparency and inter-company process, in addition to a lack of inter-company document management to be the main problems with the *iteration*s, i.e. engineering design change processes. The whole product and its design process at the customer are not really visible to the supplier and the suppliers do not know the maturity of design data. They might have to change the design of their parts when the design of other parts changes. Information about these changes would help to minimize this rework that the exchange of preliminary design data causes. This information would also minimize the impact to project lead-time, which would improve the lead-time controllability of a PD project.

The engineering design iterations caused 167 to 252 calendar days of rework to the supplier in the projects studied by Borgman & Sulonen (2003, 5). An average injection mould was scheduled to be manufactured in 100-150 days. Controlling product development is very difficult and one main reason for this is the impact of preliminary design data exchange.

Therefore, the researchers find it is essential to be able to manage the design data documents in product development networks.

Design data is stored in documents, which are usually kept in a document management system (DMS). All companies in the network need to have correct versions of the documents at all times. The documents can be delivered in different ways: they can be sent as attachments by e-mail, companies can provide their partners with web user interfaces (UI) to their DMSs, or the systems can be integrated on a system-to-system level. The delivery process should be as flexible as possible, but as stiff as needed: too loose a process makes it susceptible to errors, but too fixed a process can be difficult to implement and thus make it time consuming to add new partners to the network.

Paasivaara & Lassenius (2001) have conducted a case study on communication in a PD network of four Finnish consumer electronics companies. Their research shows that project managers act as information gatekeepers and weekly project meetings are the main tool for change management. Memos of these meetings are the main source of information, and e-mail its distribution mechanism in the network. The study introduces problems like over reliance on gatekeepers, lack of defined methods for communication, and lack of inter-company process understanding.

To have project managers act as gatekeepers becomes a problem if they are absent (on a business trip, ill, on vacation etc.). Even if they are at work, they might forget to e-mail the documents to their partners. Offering a web UI to one network member's DMS makes sure everyone has access to the correct document versions. There are other problems with this solution, though. Small suppliers might have plenty of big customers, each providing a DMS to their partners. The suppliers' personnel would have to learn how to use a number of different systems. In addition, customers do not want to let suppliers use their systems, because they are afraid that the suppliers would gain access to confidential information at the same time (Paasivaara & Lassenius 2001).

System-to-system integration is the fastest and most reliable of the document delivery solutions presented. Automating the process saves time, lessens manual work, and helps to avoid problems that arise when there are people in the middle (e.g. when they are absent or forget to forward the documents). The people interviewed by Paasivaara & Lassenius (2001)

want to have their proprietary information physically located inside their own company. This is also achieved with system-to-system integration, as documents are only copied to the partners' systems and thus also remain in the document owner's firm.

1.2 RESEARCH PROBLEM AND GOALS

Having stated the need for DMS integration, there are also certain problems that arise with it and need to be dealt with. Firstly, many e-business standards for system-to-system integration have been developed (e.g. EDI, RosettaNet, ebXML), but they mainly deal with the order-delivery process. The standards have not yet been successfully implemented in the collaborative product design process. The companies need a standard for the process, though. If they construct point-to-point integration with each of their partners it will be time and resource consuming to take new partners into the network. The companies will have to negotiate all the details of the integration with each partner separately. PD networks change from project to project, so adding new partners to the network needs to be fast and simple.

The partners have often already built some form of business-to-business (B2B) application integration, e.g. in forecasting or the order-delivery process. Adding DMS integration to this already existing infrastructure would not necessarily require much additional investment. For example, if standard messaging is already in use with some partners in forecasting, adding new collaborative engineering processes will only require configuring the system.

In DMS integration there are actually two things that need to be shared between the companies: the document files and *metadata.* The latter describe the document, its structure and relationships. The metadata are needed to save the document to the partner's system so that the people who need it can easily find it, and have all the relevant information of it at hand. This information is presented as a *data model* of the document. It consists of the document attributes (e.g. name, type, author), structure, life-cycle status and relationships.

A second problem with the integration is that the internal data models in different companies have diverse sets of attributes, derived from the respective company's needs. Different DMS's also have diverse data models. There is a need to have a standard data model of the product development documents to be shared. This way the companies will not have to define the exchanged metadata separately with each one of their partners, thus making it easier to add new companies to the collaboration network.

A third problem is the mapping of this standard data model to an existing information exchange standard. This way it will become available for all possible collaboration partners. Companies that are not yet members in a network can prepare their information systems for future collaboration according to public standards. Another issue is the document file (or files) that needs to be delivered with the corresponding metadata. The handling of attachment files is not always well thought out in the e-business standards.

Kotinurmi et al. (2003a) have compared and analyzed 15 standard frameworks for exchanging design data between companies. The comparison of the frameworks is described in section 4.3. The researchers find RosettaNet to be the most suitable framework for B2B application integration in the PD network studied, and based on this analysis it was selected to be used in the NetData[1] research project. Therefore, RosettaNet will also be used in this study. The standard is introduced in section 4.4.

The problems in DMS integration I have introduced lead to the research problem this thesis aims to find a solution to. The problem can be summed up in the following problem statement:

Is it possible to develop a standard data model for product development documents in business-to-business document management system integration with RosettaNet?

This question can be divided into the following research questions:

1. What metadata of PD documents needs to be shared in collaborative design networks?
    a. What are the common attributes in different companies internal data models?
    b. What are the common concepts in different document management systems?
    c. Which of these common concepts need to be transferred with every document and which are optional?
2. Can document management systems be integrated with the RosettaNet standard?
    a. How do the DMS concepts map to the concepts in RosettaNet?
    b. Can documents be transferred from one system to another with RosettaNet?

[1] http://www.soberit.hut.fi/netdata/

The major contribution of this work will be to construct and evaluate a data model for sharing PD documents in collaborative design networks. Four document management systems and three companies' internal data models for PD documents are analyzed to find the common concepts and attributes needed for transferring documents. The research data comes from case companies participating in the NetData research project. The data is described in chapter 5.1.

The thesis will also consider the pros and cons of a standard data model. The data model will be implemented in a prototype environment, and it will then be evaluated in terms of these benefits and drawbacks.

1.3 PREVIOUS RESEARCH

There has been plenty of research on project management and PD, but little attention has been paid to document management – document generation, distribution, and changes – in that context. Another less explored issue is communication in PD through document control. (Amami & Beghini 2000, 7)

The same rarity of empirical studies goes for organizational document metadata: its existence, creation, and use in organizations. Theoretical approaches to metadata in non-traditional settings are not easy to find either. (Murphy 1998, 271)

Salminen et al. (1996) have conducted a case study on standardizing legislative documents. They decide to follow the idea of structured documents, which is a different approach to the one in this research. For structured documents, the logical structure is defined like a database schema in the design phase. This helps in automating the handling of the documents, like publishing them on the World Wide Web (WWW). The researchers find that SGML (Standard Generalized Markup Language), a declarative markup language for describing document structure, includes no means for describing the notions and concepts needed in document management, though.

Päivärinta et al. (2002) have studied different document metadata standards. These findings are summarized in chapter 4. Also, Burnett et al. (1999) have analyzed metadata standards. Based on the analysis of these standards Päivärinta et al. (2002) and their target organization's experts developed an enterprise-wide metadata specification for documents. Their study's context is to integrate document management systems after a merger, and the specification is

intended to be used with all documents, not only PD documents. Therefore, their 48 metadata element definition is too effusive for the use in collaborative design networks.

Päivärinta & Tyrväinen (1998) have also developed a framework for analyzing organizational document genres. Karjalainen et al. (2000, 1) define *document genres* as "typified, enacted and shared purposes and forms of documents occurring in recurrent situations … in the organization under analysis." Tyrväinen & Päivärinta (1999) have used the framework they developed previously as a tool to identify and evaluate 11 organizational document genres of an industrial organization. Their results show that universal definitions for a document are not sufficient in an organizational context when designing DMSs. They find that an analysis of document genres would enable the development of a particular genre and the related technology and processes.

Karjalainen et al. (2000) have conducted action research to identify and analyze metadata, which is related to genres of organizational communication. They did not attempt to analyze metadata at the document level, though. The metadata they identified on genres was related to describing the users, producers and the use frequency of genres; describing the current technological implementations of a genre; and about processing needs of a genre. Therefore, their work cannot be used as such in this context.

1.4 SCOPE

The context of this research is networked PD process. More specifically, document management between companies in such a process. Document management can be thought of as a subset of Product Data Management (PDM). This relationship is discussed in chapter 2 and can be seen in Figure 2. This thesis concentrates on the document management part of such systems; PDM functionality and related concepts, such as product structures, are out of the scope of the research.

The description of the collaborative engineering process and the definition of new processes to support it are not included in this thesis, although they are a part of the NetData research project. The data model developed in this thesis is intended to be used in such a process, though.

The implementation of the prototype described in this thesis will consist of the communication between the document management system and the integration server. Inter-organizational communication between the servers in two distinct companies is left out of this discussion. Therefore, issues related to messaging as defined in the RosettaNet Implementation Framework (RNIF) and the associated security and data transmission issues are not a part of the thesis.

The prototype includes interfaces to a document management system and an integration server for RosettaNet messaging. These systems are typically found in large companies, whereas small firms might not have either of them. Therefore, the results are intended to be applicable in large organizations, but can also be used on relevant parts in small and medium-sized enterprises.

1.5 METHODOLOGY

Based on a literature review, Kontio (2001, 14-15) concludes that software engineering has a strong need for empirical research; it has been a research field that focuses on building and developing models, with less attention paid to empirical evaluation of these constructs. With this in mind, the goal of this thesis is to construct a model and validate it empirically with a software prototype, even though extensive empirical research cannot be conducted in the scope of a master's thesis.

The research is divided into four phases according to Glass' classification (1995, 4): informational, prepositional, analytical and evaluative phase. A research project does not need to utilize all of these steps, which is often the case in real studies: there is much more proposing and analyzing than there is evaluation (Glass 1995, 4-5). This is also the situtation in this study – the emphasis is on developing the construct. It will be evaluated further in future research, although some validation is done during this research as well.

1.5.1 Informational phase - Understanding the problem

The research on existing document management systems and standards is conducted as a literature review. In addition, the data models and existing systems of the partner companies are studied.

The research data consists of interview notes and organizational documents. Information is gathered in discussions with the representatives of the target organizations, as well as from the related documents, such as instruction manuals for document management systems and guidelines for internal data models.

1.5.2 Prepositional phase - Solution

The data model is created based on the literature review and research data. The design, implementation, and testing of the software prototype is conducted as a team with the NetData project staff and two student groups from programming and software project courses at HUT.

1.5.3 Analytical phase

In the analytical phase the developed solution will be evaluated and improved according to the results of exploratory trials in a prototype environment and discussions with company representatives and document management experts. The prepositional and analytical phases will be partly intertwined, as the solution will be further developed according to the feedback gathered. The development is conducted in an iterative manner.

1.5.4 Evaluative phase - Validation

In the evaluative phase the construct will be tested and evaluated. Both the prototype and the data model shall be evaluated with the partner companies. Due to the limited scope of a master's thesis more thorough empirical evaluation cannot be conducted. Arrangements for further evaluation will be discussed in chapter 9.

1.6 TERMINOLOGY

Metadata

Metadata is information about information. In the context of document management, metadata refers to information about a document, e.g. its author, version, state, and creation date. Metadata is important in searching for the right information (i.e. the correct document). Both metadata and the content it describes can exist without their counterparts. (Saarela 1999, 14-15).

Metadata can be explicit or implicit. The former approach means that someone makes explicit statements about the document, for example, fills in a form. The latter approach refers to

automated metadata generation through examining the object and using heuristics to determine metadata. An example of this is using pattern recognition systems to determine the subject of a picture, or using the author's login identification to the computer system. (Saarela 1999, 16,31)

Document

In this thesis documents are defined according to Anttila (2001, 2) and Peltonen et al. (2002, 47) as a combination of metadata and the representation files. The files can be drawings, plain text files or in any other format, even paper copies. All documents have metadata, at least some attributes that describe it. There does not need to be a file representing the document and, on the other hand, there can be more than one file belonging to one single document. In the case of paper copies a document management system only includes a reference to the representation, whereas digital representation files are also stored in the system (Peltonen 2000, 48).

Data model

Peltonen (2000, 22) defines a data model as "a definition of the basic concepts of an information system by means of objects and relationships between objects." He also distinguishes between *metamodels* and *company models*. The former are more general, whereas the latter are company specific. Metamodels are used when defining the company models. They define the general concepts, whereas company models define concepts for each firm's particular needs. The concepts are defined by item types, their class hierarchy, attributes and relationships between the items. (Peltonen 2000, 22)

1.7 STRUCTURE OF THE THESIS

The first chapter of the thesis has presented the background of the research, the research problems, objectives and scope, as well as the methodology used. Chapter two introduces the context of the research: collaborative design. It also covers other background for the study. Chapter three provides an introduction to document management on the parts related to product development documents. Chapter four introduces document standards: document metadata standards, document structure standards, and B2B e-commerce standards, from which RosettaNet is described in more detail.

Chapter five presents the research data and the development of the standard data model. Chapter six describes the standard data model, and chapter seven presents the prototype system. Both the data model and the prototype are evaluated in chapter eight, which also includes discussion. Chapter nine contains conclusions of the research and ideas and plans for future work.

2 BACKGROUND

The context of this research is product development networks. This chapter describes the subject, from traditional PD to collaborative design. The concept of engineering changes (ECs) is especially important in this context, as it creates a need to exchange many PD documents. ECs are introduced in section 2.2. Section 2.3 describes document management in a PD network. Finally, systems integration is introduced.

2.1 FROM TRADITIONAL PD TO COLLABORATIVE DESIGN

A traditional new product development process has been described, for example, by Cooper (1993) with the Cooper's Stage-Gate™ model, and by Ulrich & Eppinger (2000) with their Generic NPD model. The process starts with a concept of a new product. It often comes from marketing or research and development teams. The idea is the basis for the system-level and detail design, which create the product's physical dimensions, performance information, and materials specifications. This is followed by testing and refinement. After the design is ready, it is transferred to manufacturing, which develops a prototype of the product according to the design. If the prototype is satisfactory, manufacturing begins the ramp up to production at full volume. The promotion and selling of the product is the responsibility of the marketing department.

This process is essentially sequential. Each function follows the previous, as the product and information is passed on to them. The downstream tasks wait for their predecessors to complete their tasks before beginning to work, even though some of this work could be done simultaneously. This results in the process taking longer. There is also little or no discussion on product or process improvements between the functions. In addition, giving the development team information on manufacturing's current production capabilities, strengths, and weaknesses early enough allows the developers to consider this when they can still do something about it. (McDermott & Handfield 2000, 38-39)

Changes later in the product life cycle (e.g. during production) cause a lot of rework, costs and delays in the project. Closer co-operation between the different departments helps to avoid these late changes. (Helms 2002, 52)

Zirger & Hartley (1996, 150) studied 12 different strategies' effect on product development time. They find team structure and management variables to have the greatest impact on PD times. The four techniques that accelerate PD are: increasing the functions represented on the team, reducing the number of projects to which team members are assigned and placing a high priority on time as a project goal. Overlapping development activities was also a significant characteristic of the development process: it reduces overall time as the activities are conducted simultaneously, but also cuts time in project review preparation, presentation, and resulting ECs as more functional representation is present.

McDermott & Handfield (2000) conducted an empirical study of six radical new product development projects in six case companies on their PD practices. On the basis of this study they propose concurrent engineering especially suitable for well-defined technologies in meeting time-to-market, cost and market penetration objectives. They find the serial approach more appropriate for developing breakthrough products with new technologies.

Zirger & Hartley (1996, 145) point out, that even though greater number of overlapped activities is expected to decrease development time, it also significantly increases information processing requirements for the team. An example they give is the need for design engineers to incorporate downstream constraints from manufacturing, which increases the amount of information and uncertainty the teams must address. Overlapping activities has become established best practice of product design processes (Loch & Terwiesch 1998, 1032).

Concurrent engineering was initially a term used to describe the overlap of product and process engineering processes. It was later extended to also cover overlapping other phases of the product life cycle (e.g. operational use and service). This integration is created using multidisciplinary teams with representatives from different phases of the product life cycle. (Helms 2002, 51)

McDermott & Handfield (2000, 43-44) reviewed literature on supplier integration to new product development. They find a variety of benefits in this: including suppliers on project teams allows for a smoother, less expensive and speedier development process. In addition, communication and information exchange reduce delays and add information and expertise regarding new ideas and technology, and helps to identify potential problems so they can be resolved up front. Supplier integration also allows considering accessibility of parts early on,

in addition to providing outsourcing possibilities that reduce the internal complexity of projects and provides extra personnel to shorten the critical path for the projects. Taking suppliers into the team improves the supplier relationship, which leads to commitment and a smoother working relationship.

Collaborative design takes concurrent engineering one step further, by extending the functions taking part in PD teams to partner companies' functions. It is organized between supply chain partners, and used for new product development and designing new revisions to old products. The collaboration with supply chain partners takes advantage of the design to manufacturing capabilities of the outsourced manufacturer, which reduces the lead-time and costs of production. (Cartwright, 2002)

2.2 ENGINEERING CHANGES

The design information evolves over time. Concurrent engineering forces the design team to work with preliminary document versions, since they cannot waste time waiting for the documents to reach their final state. This causes problems, as changes to the documents affect the design and manufacture of other components. Loch and Terwiesch (1998, 1034) define preliminary information as design information that is precise from the start, but which is then modified repeatedly in the *upstream* activity as the design evolves. The modifications are incorporated *downstream* through ECs. The iterations can also result from finding an error in a downstream activity, or through concurrent changes in the common assumptions.

The authors' later work (Terwiesch & Loch 1999b) has shown, that the ECs virtually always become more difficult to implement the later they occur. They postulate that all changes are communicated in meetings between product and process engineers, and downstream will not become aware of any new ECs until the next meeting (Loch and Terwiesch 1998, 1036).

Engineering changes are not just a cost and time delay factor in a PD project. They can bring cost savings and performance improvements into the project. Still, many engineering change orders (ECOs), long ECO processing times and especially late ECOs bring high costs to a development project. A long response time causes late implementation of the ECO, which increases change costs for tools and interfacing components. In addition, a long response time means also having many problems open simultaneously, which causes problems in coordinating the change efforts. With CAD (Computer Aided Design) programs many

engineers can work with the same data simultaneously, and coordination becomes more important. Finally, there is also a risk that the conditions that required a change at the beginning of the process have changed during the approval process of the ECO. Thus, by the time the ECO is approved, it might already be outdated. (Terwiesch & Loch 1999b, 161-162)

As a consequence, there has been plenty of research on reducing the negative effects of ECOs. Terwiesch & Loch (1999b, 161-162) have examined this literature and classified strategies proposed to reduce the negative consequences of ECOs. They call the classification "Four Principles of ECO Management". These principles are: 1) avoid unnecessary changes, 2) reduce the negative impacts of an ECO, 3) detect ECOs early, and 4) speed up the ECO process.

Terwiesch & Loch (1999a, 456) summarize findings from their previous work (Loch & Terwiesch 1998): "The more uncertain the upstream activity, the more ECs will occur during the project." This means that the uncertainty in a development project can be described by the amount of ECs in it. They also show statistically how uncertainty resolution influences the effectiveness of overlap. Their findings show, that overlap gains increase with fewer and earlier ECs. The gains have to be weighed against the rework delay resulting from the use of preliminary information by the downstream task (Loch & Terwiesch 1998, 1043).

2.3 DOCUMENT MANAGEMENT IN A PD NETWORK

The organizations taking part in the collaboration all have their own practices, systems, and methods, with which they are used to operate. The partners are often located in various geographical locations. They will have to adapt their organizational structure, workflow, and information systems to the ones of the collaboration partners. (Chen et al. 1998, 2)

The engineers who work in different locations use different systems and computer platforms linked via the Internet. Domazet et al. (2000, 1-2) identify their needs for sharing design information: the engineers have to be able to view parts designed by other team members, design assigned components or sub-assemblies under constraints specified, analyze, discuss and modify design solutions, propagate design modifications as soon as possible, and review and verify design solution in different phases of a development process. This information is usually stored in documents.

Banerjee & Kumar (2002, 97) list business objectives that can be addressed by electronic business document interchange. The first item on their list is quick response to changing product demand and new product introduction. They also list operational goals to achieve the business objectives. These include improved cross-functional communication, tighter integration and improved communication with strategic trading partners, and globalizing work to make 24-hour workdays.

Lublinsky (2002, 38-39) sees B2B integration as coordinating information among organizations and their information systems. He lists the key principles of business success: faster time-to-market with new products and services and lower production costs. Communication breakdowns lose time and money in production delays or budget overruns, whereas information sharing allows for better understanding of market needs and trends, earlier initial orders, better production planning, and higher margins. Collaboration over the Internet through integrated systems can automate business processes.

In order to effectively collaborate, the document management processes have to be extended from individual companies to the network of collaborative partners. Linking the original equipment manufacturers' (OEM) information systems with those of their suppliers is a precondition for rapid product introduction and cost efficiencies through outsourcing. Integration of design and production functions even within companies is still a difficult task, though. The inefficiencies of the current outsourcing model result in costs and product delays. (Cartwright, 2002)

Collaborative design processes are unstructured and dynamic in nature, so in addition to integrating the document management systems, the workflow definitions have to be modifiable. Exception handling has to be flexible as well, since unpredictable situations occur often and have to be resolved successfully in collaborative engineering processes. The activities should be initiated by events from the collaborative engineering environment. (Domazet et al. 2000, 2)

An example of document management in a PD network consists of a customer, a design supplier, and a manufacturing supplier[2]. Each of the partners has their own DMS, and document exchange is done by e-mail. This situation is illustrated in Figure 1.

Figure 1 An example of communication in a product development network

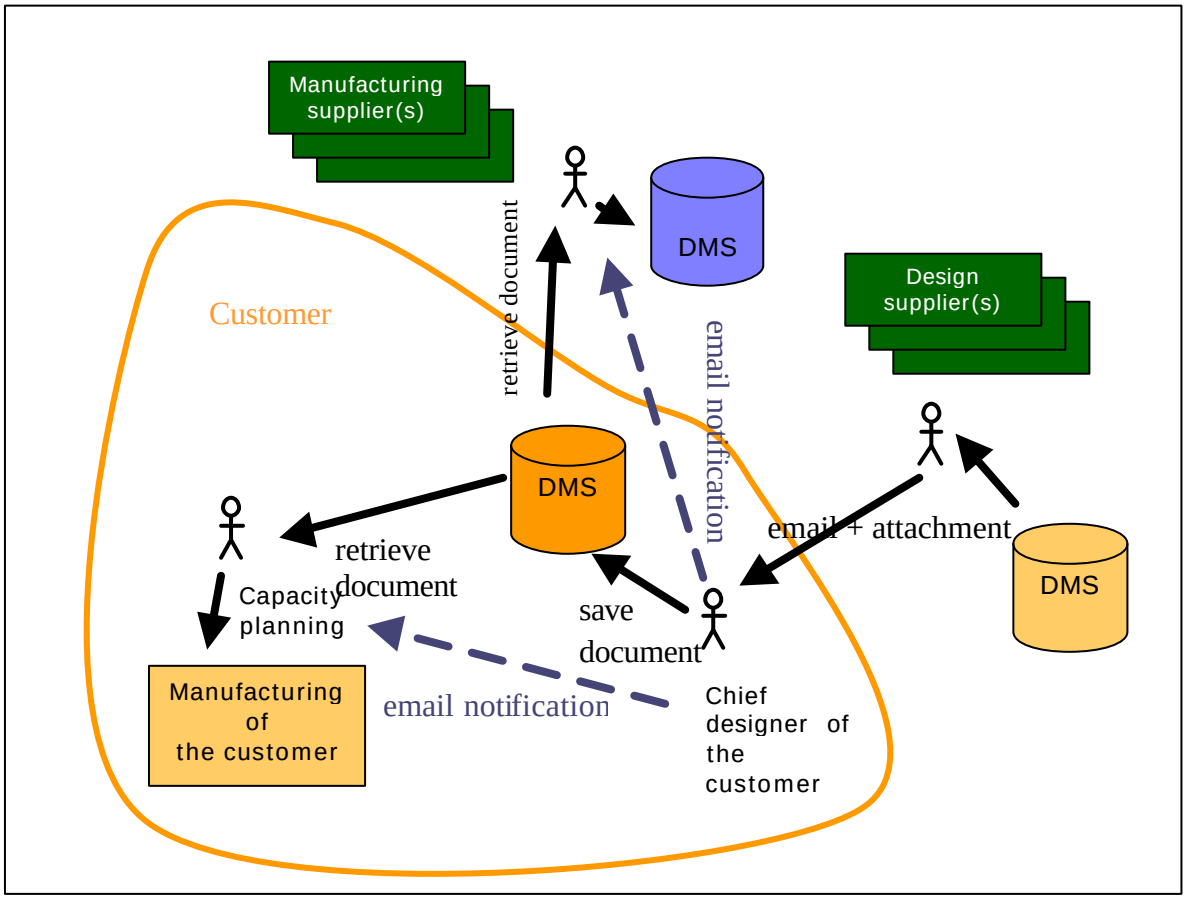


Figure provided by Jukka Borgman

A typical document delivery process begins when an engineer in the design supplier company modifies the current design of one component. This change needs to be communicated to the customer and the manufacturing supplier quickly, as the manufacturer is about to start machining of the component according to an old drawing. Also, the manufacturing of the customer needs to be informed, because their components interact with the modified one.

[2] Example provided by Jukka Borgman. It is based on his research (Borgman & Sulonen 2003), but not an exact case situation.

At the moment the EC is communicated as follows. The designer sends the modified document to the customer's project manager. He then saves the document in their DMS and informs their own manufacturing of the new document by e-mail. He also sends the document as an e-mail attachment to the manufacturing supplier's contact person, who then adds it to their DMS.

This kind of an arrangement causes problems, as the correct version of the document does not always reach the right people on time. The three main reasons for this have been described by Kotinurmi et al. (2003a, 2). The first reason is that the document is not sent to all the concerned designers in the network Secondly, the distribution is done by e-mail from project manager to project manager, which causes problems if one of them is absent at the time or just forgets to forward the document to other relevant parties. The third reason is that e-mail is not a very secure or efficient way for transferring large files as attachments (a CAD drawing's size can typically be 1-100 MB). Therefore, special data-transfer directories for each partner are set up and as files need to be transferred they are copied to this directory. The document is then automatically encrypted and transferred to the partner company by ftp (file transfer protocol). This is not a good solution, though. As the files transferred do not include any metadata the name of the file is the only way to distinguish the documents from each other. Some files cannot be delivered to the right people, as it can be impossible to determine to which project or component they belong.

These problems lead to working with old versions of design documents, which in turn causes delays and rework. Therefore, a more efficient solution has to be developed. Kotinurmi et al. (2003a, 4) have identified information technology requirements for document management in networked PD. These requirements are of two categories: process and messaging related. The processes specify the activities that are to be carried out in a specific order. The messaging instructions specify the format and contents of the *message*s exchanged by the partners. The document delivery message contents include delivery information (e.g. sender and receiver), document metadata, and the document file(s) as attachments. The messaging instructions also cover the packaging and encrypting of the messages, and responding to network problems. Messaging also makes sure the document delivery is *non-repudiated*, i.e. the sender cannot later deny having sent the message and the receiver cannot deny having received it. This can be an important legal issue. Both processes and messaging have been addressed in standard frameworks, which are further discussed in section 4.3.

2.4 SYSTEMS INTEGRATION

E-business requires integrated business processes within, across, and between organizations. The organizations often have *legacy systems*, i.e. old, often monolithic information systems that continue to be used because of the cost of replacing or modifying them. The integration of business processes requires integrating these legacy systems, as well as business functions, application programming interfaces (APIs), and databases across departmental and corporate boundaries. The old processes should not be automated; instead, networks of new, highly efficient virtual organizations should be encouraged. (Yang & Papazoglou 2000, 40)

These organizations must quickly adapt to change to remain competitive. The life cycles of virtual manufacturing organizations can span from several years to just a few months or even weeks. Information systems cannot be designed with tightly integrated approaches (e.g. federated information management systems); instead, the integration levels must be flexible. Rapidly changing environment does not leave time for traditional approaches. (Chen et al. 1998, 3)

Our interviews in the case companies revealed that the suppliers for one company can change as often as weekly, and there can be hundreds of them. One of our case companies states having about 300 suppliers. Adding a point-to-point connection to a supplier can take three months. A PD process can take the same time, which makes this kind of set up time unacceptable.[3]

The business processes must be reconfigured according to changing market conditions. The business processes evolve too quickly for linking also evolving applications across organizational boundaries with established approaches. The day-to-day business cannot be interrupted when responding to the changes in the environment. Instead, enterprise models have to be extensible to accommodate for changes. (Yang & Papazoglou 2000, 41-43)

Business process reengineering is both an opportunity and a prerequisite for electronic document management (EDM). Business processes, like the product development process, have to be analyzed to find out how they will be affected by EDM. Interdisciplinary

[3] The name of the company and the person are not revealed here due to confidentiality reasons.

cooperation is required to provide an integrated technology infrastructure for EDM. (Meier & Sprague 1996, 59)

EAI (Enterprise Application Integration) deals with integrating existing information systems within an organization. This usually involves heterogeneous legacy systems, which already makes it a difficult task. To support inter-organizational processes, the same work has to be done for heterogeneous information systems between organizations, which can be an even more challenging task. This is referred to as B2B application integration. (Hasselbring 2000, 33-34)

The business processes should not have to be changed to fit an applications' functionality. The information architecture should be such that it can be aligned with the business organization. (Hasselbring 2000, 34)

Message formats and contents have to be standardized so that the applications understand the data provided by other applications (Hasselbring 2000, 35). This is not a trivial task, as Chen et al. (1998, 2) note that this has not been achieved even within key components in virtual manufacturing organizations. They point to engineering analysis and design, in which the various tools are not fully integrated, due to the incompatibility among existing standards and different vendors' products. Effort is needed to guarantee a smooth information flow between the components.

2.5 SUMMARY

This chapter has provided a short presentation of the evolution of product development, engineering changes, document management in PD networks, and systems integration. The traditional way of conducting PD is not fast and efficient enough. The repeated ECs require an iterative process, with multidisciplinary teams. The members of the team do not work within the same company, and they work in different geographical locations. Product information is usually stored in documents, and it needs to be shared between the design team members.

As concluded in this chapter, the collaborative new product development process has substantial benefits of DMS integration between the partners. The integration saves time and reduces costs, as the engineers have the right documents at the right time. Systems integration

is not a trivial task, though. Different companies have their legacy systems, and since the teams are dynamic in nature, there is a need for a fast procedure to add new members to a team. Therefore, point-to-point integration is not a good enough solution for the problem, but a standard solution needs to be developed.

The next chapter describes document management in more detail. It presents the concepts and processes relevant to product development documents. These issues are important when developing the standard data model for PD documents.

3 DOCUMENT MANAGEMENT

Document management (also electronic, or enterprise document management - EDM) is a function, which enables documents to be managed as an information resource instead of a collection of files (Sprague 1995, 39). The document management functions include, among others, storage of the documents, maintenance of their metadata (see 1.6), version control, access control, and document life cycle management. Without document management it would become impossible to find the right documents and their correct versions in a company. One of the case companies in this research had already in 1994 over one million different technical documents in one of its divisions. Information systems help in document management, but even more important is the process associated with creating and modifying documents or managing their life cycle status.

In the following sections I will first discuss the benefits of EDM. In section 3.2 I will present the relationship between document management and product data management. In section 3.3 the basic concepts and processes in document management are introduced. Document metadata models and basic data modeling concepts are discussed in the last two sections.

3.1 BENEFITS OF DOCUMENT MANAGEMENT

Meier & Sprague (1996, 57-58) list the opportunities and benefits of electronic document management: firstly, it supports both internal and external communication. Secondly, it acts as a business process vehicle and an enabler of business process reengineering. Finally, electronic document management aids in achieving an expanded organizational memory with better access for all relevant parties. Document management is also a large factor determining project performance. When companies want to shorten product development time they need to consider their document management activities, which define the fast-cycling capabilities to introduce products to the market (Amami & Beghini 2000, 7).

Sprague (1995, 32) categorizes the business value of documents first in documents as a source of revenue (e.g. in publishing companies), and second in supporting organizational performance as a mechanism for organizational communication, as a vehicle for business process, and as a major component of organizational memory.

Amami & Beghini (2000, 9) classify the value of document management in supporting a PD process in three groups: a mechanism of communication and management of information flows; a vehicle for product development quality; and a linkage among project planning and control, product structure, procedures, and physical data storage. Although their article only discusses project management within a company, their findings can be extended to collaboration networks. Actually, the value of document management even increases, as the number of partners, geographical distance and difference in information systems increases.

3.2 DOCUMENT MANAGEMENT AND PDM

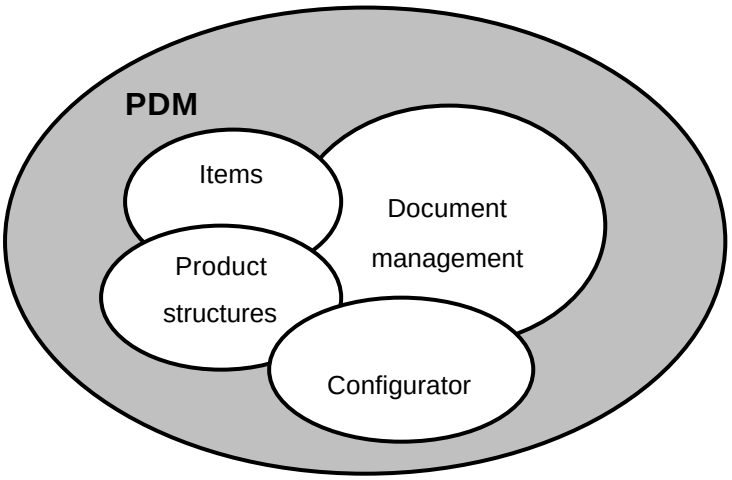
Product data can be defined as all data related to products. This is a rather broad definition, as according to it almost all data in a manufacturing company can be classified as product data. Usually PDM refers to engineering data in product development. (Peltonen 2000, 18)

Product data is largely represented as documents created with specific tools, such as CAD tools or text processors (Peltonen 2000, 47). PD documents can be managed with a product data management system (PDMS). In addition to document management such systems also manage items, product structures, as well as their relationships and changes in them. (Anttila 2001, 90)

Some products are *configurable*, i.e. they can be customized according to customer requirements. The products are designed to fulfil a set of similar customer requirements through combining previously designed components. *Configuration management* can refer to such configurable products, but the term is also used for non-configurable products; for example, the configurations can refer to consequtive revisions of a mass product. Configuration management can also be a part of a PDMS. (Peltonen et al. 2002, 79-81)

Document management is only a part of PDM, although a major part. The basic functions of PDM can be seen in Figure 2. In this thesis the emphasis is on document management as a part of PDM.

Figure 2 Product Data Management



Source: Anttila 2001, 91

There are also plain document management systems, but they do not provide the same functionality as PDMSs. Usually, the DMSs focus on the office environment and are linked to office software, whereas PDM systems' focus is on product development. Therefore, PDM systems offer integrations to CAD systems and possibilities to manage CAD files. (Helms 2002, 29)

The links from DMSs (and PDMSs) to different applications allow the user to open, modify and then save a document to the DMS from the application they are using (e.g. word processor). This can be achieved, for example, with ODMA[4] (Open Document Management API), an industry standard interface for managing documents. (Anttila 2000, 161)

[4] http://www.infonuovo.com/odma/

3.3 CONCEPTS AND PROCESSES

Helms (2002, 22) summarizes the main document management functions and sub-functions as:

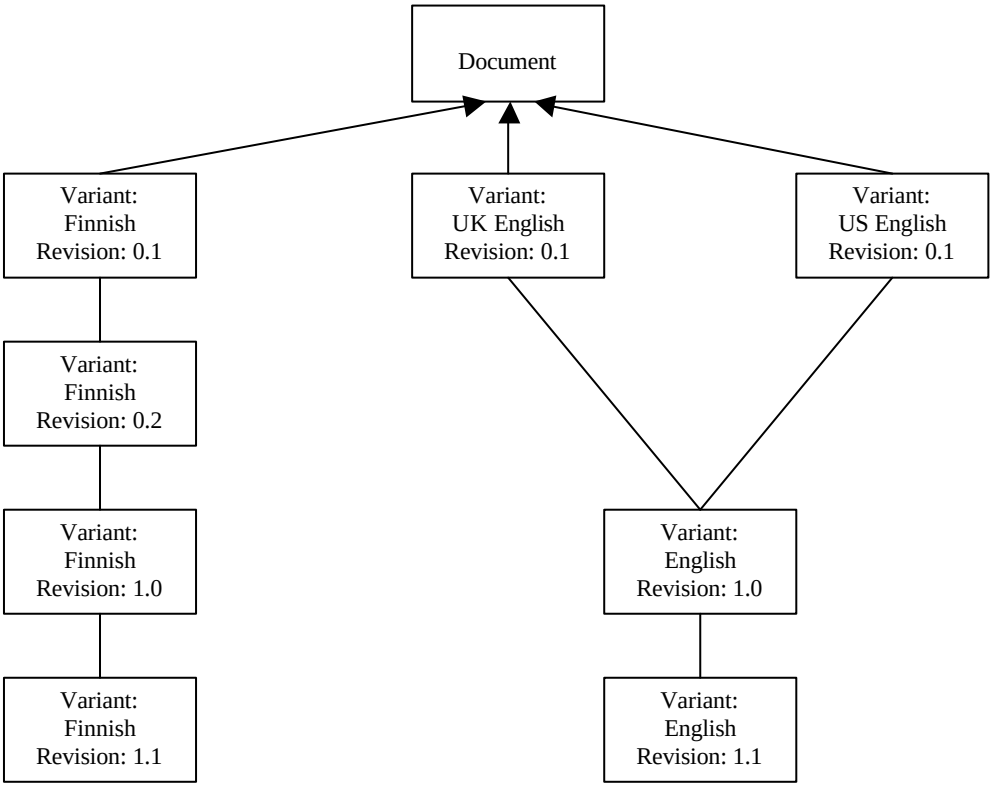
- Document Repository management
  - Vault management
  - Document identification & numbering
  - Document viewing and mark-up
  - Scanning and imaging
- Document Structure management
  - Document structure modeling
  - Document structure viewing
  - Document change impact analysis
- Document life cycle management
  - Document version control
  - Document status and release control
  - Document baselines

Document repository management functions are important to document management within a DMS. The latter two are more relevant in DMS integration. These two concepts are discussed in more detail in the following.

3.3.1 Document structure

Document structure includes the document's *versions* and their organization into *variants* and *revisions* (Peltonen et al. 2002, 55). The terms version, variant and revision are somewhat overlapping and their meaning varies. I will use the terms the same way Peltonen (2000, 50) does: revisions represent the evolution of a document through successive stages. Variants are parallel alternatives of the document, for example, different language versions. Both revisions and variants are called versions and they can form various structures with branching and merging. An example of this is illustrated in Figure 3.

Figure 3 Example of revisions and variants of a document



Keeping track of different document versions enables the users to have access to document history and the changes made to the document. They can go back to work with earlier versions of the document if needed. (Anttila 2001, 37)

Documents are often made of several *subdocuments*, which also belong to the document structure. A simple example from Peltonen (2000, 48) is a text file that contains a figure. The figure can be stored as a different document or as a separate file within the same document. In such a case, the text file contains only a link to the figure file. Another example from product development is a CAD drawing that has several pages. Each page can be thought of as a subdocument of the drawing.

The document versions have *representations*, which are the files with the document content. There can be primary and secondary representations. For example, CAD files are usually in a format that can only be viewed with the program they were created with. Therefore, there can be a secondary representation (in, e.g., PDF or JPG format) that allows for viewing the file without the specific CAD program.

Helms (2002, 21) uses the term document structure to mean relationships between documents and document versions. He has identified different types of relationships: hierarchical and lateral, and internal and external. Hierarchical relationships define the different subdocuments the document is made of (e.g. a manual includes a user manual and a maintenance manual), whereas the lateral, or logical relationships describe relationships between documents on an equal level (e.g. a particular version of calculation that belongs to a particular version of a drawing). Relations between documents are called external relationships. Both hierarchical and lateral relationships belong to this category. Internal relationships are relations within a file. A link to a figure file in a text file from the previous example would fall into this category. (Helms 2002, 21-22)

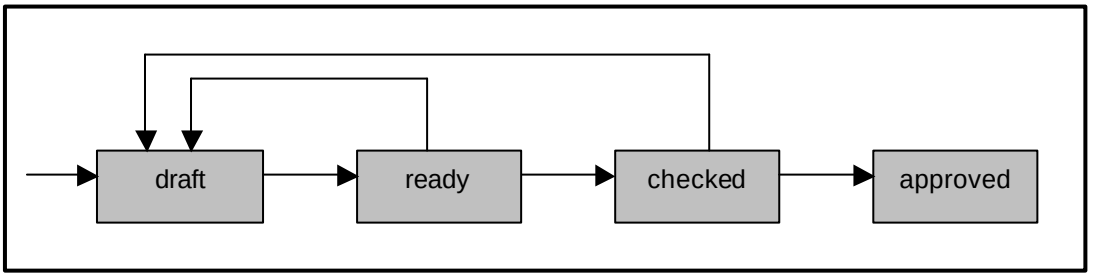
All these relationships should be represented in the document management system. Document structures are especially important in the product design context. An EC causes changes to a document. Document structures help in defining the impact of the change, as they show the related documents that might need to be changed accordingly. (Helms 2002, 22)

3.3.2 Document life cycle

All documents have a life cycle. The cycle begins when the document is created and ends when it is removed from the DMS or its status is changed to indicate that the document is no longer active. The life cycle can include various versions (variants and revisions) and statuses. Document life cycle states describe its evolution. The states and the allowed transitions from one state to another can be described with a *state chart or state transition diagram*. These state charts can be attached to a document or to a document version. Figure 4 shows an example of a state graph for document versions in a document management system (EDMS – Engineering Data Management System) (Peltonen 2000, 120). These state charts vary from company to company.

The states can be used to support the release process of a document. The authorization to access, modify or delete a document can be defined for each state. Release procedures are typically used to increase the quality of the documents, as errors are detected and improvements suggested in the process. Such a process consists of at least two activities: review/check and authorization & freeze. In Figure 4 the first state (*draft*) is for editing the document, the following two (*ready, checked*) for review/check and the last (*approved*) for authorization & freeze. (Helms 2002, 22, 44)

Figure 4 Sample EDMS state graph for document versions



Source: Peltonen 2000, 120

The state graph in Figure 4 includes loops, which mean that the version can go back to *draft* if, for example, the approver finds mistakes in it. Helms (2002, 47) suggests this should not be possible as it makes it impossible to monitor the history of the document[5]. He thinks document versions should always be promoted to a higher status and never return to a lower status; loops in the state graph should be replaced by creating a new version each time it is needed to go back to a lower status.

This way of working might lead to so many versions, though, that it would be impossible to follow the evolution of the document. Therefore, some loops in the state transition diagrams are acceptable. Following the state graph of Figure 4 after the version has been approved, it cannot be modified further. If the document needs to be modified a new version has to be created. Thus the history of changes in the document becomes visible, but the changes to a document version during its approval process do not force a new version to be created.

According to Helms (2002, 22) status and version control can also be applied to document structures. This way it would be possible to see which documents have been parts of a document structure over time. He calls the stored document structure a *document baseline.*

3.3.3 Product development document life cycle

Helms (2002, 64-69) has developed a model, which describes the PD process as a document creation and release process: before the document is released it is edited in *creative iterations*. These iterations usually result in many possible solutions, from which the best alternative is selected and released as the final document. If the document is released before the creative

[5] Some systems include a history file, in which case the loops do not affect the use of history information.

process ends, there can be changes to the released document, which in turn result in rework and delays. The iterations cause multiple document versions to be created, what makes the product development more complex, as it is difficult to predict the length of the iterations.

The rework caused by iterations can be categorized as intradomain or interdomain. The first applies to documents in the same domain with a precedence relationship. For example, a part does not fit into an assembly after a change in the assembly drawing. Interdomain iteration has to do with documents from different functional domains. For example, it can be impossible to design a part that has the functional capabilities described in the functional domain. Therefore, the functional description has to be changed, which means creating a new version of the released document. (Helms 2002, 69)

Helms (2002, 70-72) suggests a well-defined release and version control process for documents as a working practice to deal with iterations in a controlled manner, as it is impossible to prevent them from taking place. Normally an engineer would start working with a new document after its predecessors are released. Instead of this, Helms suggests the predecessors to be released preliminary while the authors are still specifying them. This preliminary release should be shown in the document version state, as the engineer using it has no guarantee that there will be no more changes to it. He adds that the preliminary release should only be for a specific purpose to a specific audience.

Working with preliminary releases in networked PD requires interorganizational document management processes and accurate exchange of document versions. The engineers in the partner company need to know whether the document version is preliminary or released, and they need to get the latest versions of the document on time and without a lot of extra work.

3.4 DOCUMENT METADATA MODELS

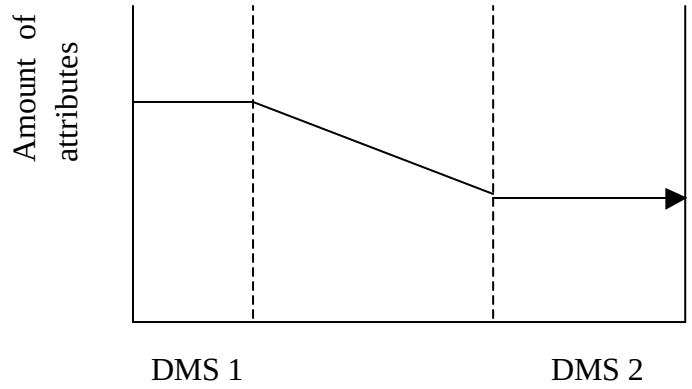
Documents are described with metadata elements in document management systems. Metadata is information about the document, such as its name, author etc. There are usually different attribute sets for each *document type* (e.g. drawing, ECO, project document) (Helms 2002, 21). A data model of a document also includes relationships between the metadata elements and constraints for them. For example, one document can have many versions, but a version can have only one predecessor. The rapid increase in the amount of digital documents in organizations suggests that the range, amount, and quality of metadata deserve

investigation (Murphy 1998, 272). The more informative the metadata is, the easier it is to find the right document the first time, thus saving time and nerves of the DMS users.

Päivärinta et al. (2002) find that the knowledge embedded in organizations' existing information systems and the tacit experience of the domain experts is important in providing practical viewpoints to organizational document metadata. They find it important to optimize the assembly of elements into a set large enough to be useful, but small enough to be used. The researchers mention the challenge of convincing the users of a DMS of the importance of metadata and motivating them to obediently fill in a metadata form for documents. Some attributes are usually added to the document automatically (e.g. the creation time and the creator of the document), but others need to be added by the user.

Another issue with the size of the element set is comparing the power of expression with the difficulty of implementation. On the one hand, a greater power of expression gives more possibilities for the use, as it allows for different kinds of searches in the DMS. On the other hand, it will be difficult to transfer documents from one system to another, if there are too many attributes. If there is a difference in the amount of attributes, it is possible to take a subset of the bigger set and save it to the system with a smaller set. This is illustrated in Figure 5. The other way around it will not work: if some attributes are missing from the smaller set they cannot be automatically filled in in the bigger set. In automating the addition of the missing attributes some heuristics or default values can be used, but this leads to the attributes not being accurate or the same as in the other system.

Figure 5 Attribute amount level



One of the company representatives interviewed on the subject said he has seen data models with up to 350 attributes, and finds them totally useless. On his opinion, the data model should be as simple as possible.

3.5 DATA MODELING CONCEPTS

Object-oriented analysis methods seem to be the most suitable methods for developing a standard data model for documents. These methods have been developed for identifying the elements of a problem area and for understanding and explaining how they interact with each other (Salminen et al. 1996, 76). The data model is thus described with basic object-oriented modeling concepts: objects and types, attributes, and relations (Peltonen 2000, 24).

3.5.1 Objects and types

The objects in the data model include documents, projects, and companies. These objects have different types. For example, document types include “drawing”, “project document” etc. These types are organized in a type hierarchy, in which the subtype *inherits* from the supertype. For example, all document types (i.e., subtypes of “document”) inherit the common document attributes. In addition to those, there can be type specific attributes[6].

3.5.2 Attributes

The objects are described with attributes; data about the object. For instance, document object can be described with attributes like “name”, “author”, “creation date”, etc. The set of attributes for an object depends on its type; for example, a document of type “change request” could have an attribute “reason for change”, which would not make sense for a document of type “project document”. An attribute’s definition includes its name and value type. As an example, the value type of “creation date” could be “date”, and that of “name” could be “a string of at most 20 characters”. The value can also be assigned from a particular value list, and the value itself can be a list of individual values. Attributes can be compulsory or optional. The former are required to have a value for each object instance of the respective type, the latter can be left out if they are not needed. (Peltonen 2000, 28-30)

[6] In the standard data model there are no document type specific attributes, but in most of the companies’ internal data models and DMSs this is the case. See discussion on this issue in chapter 6.

3.5.3 Relations

Different objects can be related to each other with various relations. An object and its versions are related to each other. Also, documents can be related to products and/or projects. The cardinality of the relation specifies the number of participants in it. For example, a document can have many or no versions, but a version is always related to one document. The cardinality is presented in brackets [] with numbers or an asterisk (*), which means zero or more. The cardinality of the previous example is [1]-[0..*]. (Peltonen 2000, 33)

3.6 SUMMARY

This chapter has described the value of document management, its basic concepts and processes, as well as document metadata models and data modeling concepts. Inter-organizational document management enables the PD process to be reengineered. Document management supports the communication between the design engineers and the other members of the virtual PD team in different companies. Product data is usually stored in a PDMS, which includes document management functionality. Other PDMS functions are out of the scope of this thesis.

The relevant concepts for DMS integration are document structure and life cycle. These include document versions, subdocuments, representations, status, and other metadata, which can be described with document metadata models. The basic building blocks of the models are objects and types, attributes, and relations. These concepts are used in developing the standard data model, which is described in section 5.2. They are also important when discussing different document standards, which also include document metadata. These standards are the subject of the next chapter.

4 DOCUMENT STANDARDS

The most important organizational support challenges in electronic document management are business process reengineering, interdisciplinary cooperation, and intra- and interorganizational document standardization, which includes standardization of both document formats and contents (Meier & Sprague 1996, 59). Different standards address different aspects of documents. The most important standards concerning this thesis are document metadata standards, but document structure standards and e-commerce frameworks need to be thought of as well. These standards are presented in the following. The last section of the chapter presents the standard chosen for our integration solution: RosettaNet.

4.1 DOCUMENT METADATA STANDARDS

There are plenty of standards for document metadata, but they have been developed mostly for public archives and information collections. One of the best known, Dublin Core[7], was developed by the library community for resource discovery on the web. It is one of the most widely adopted standards for describing document contents, but remains limited due to its simplicity and generality (Jokela 2001, 21). It could be thought of as a candidate for organizational document metadata structure, but it maps inconsistently to an organizational environment: some of the elements seem to have organizational equivalents, but others show greater variation or do not appear at all (Murphy 1998, 273). Especially the lack of important elements in the organizational context makes the use of these standards difficult, if not impossible.

Päivärinta et al. (2002) have studied and analyzed 19 of these standards in their research. The analyzed standards include ODMA (see section 3.2) and Dublin Core. The researchers report a case of defining document metadata in an engineering company with 2600 employees. They used 18 metadata standards[8] to create a baseline for the organizational metadata. Elements that appear in ten or more metadata standards form the core elements of their definition. The independent research by Burnett et al. (1999) found an almost identical set of core elements. These elements are listed in Table 1.

[7] http://dublincore.org
[8] UNIMARC was chosen to represent the family of MARC standards

Table 1 The core elements of the 18 metadata standards

Element name	Description
Availability	Access rights, use constraints or any other information available for document retrival (e.g. time period when the document is valid).
Creator	Creator, author, designer or other person responsible for the document content.
Date	Date of creation or publication of the document.
Description	Textual description of the document content (e.g. abstract, table of contents).
Format	File format of the document (e.g. doc, pdf, tiff, dwg).
Identifier	Unique identifier of the document (e.g. URL, ISBN, ISSN).
Keywords	Keywords describing the document content. Thesauri may be used.
Language	Language of the document.
Location	Location (physical or logical) of the document (e.g. URL).
Notes	Notes and comments about the document content, usage etc.
Organization	Organization of the creator or organization, which is responsible of the document content.
Publisher	Publisher of the document.
Relation	Sources to which the document is based on; references of the document; relations to other documents and objects (e.g. document is part of a collection).
Subject	Subject of the document.
Title	Title of the document.
Type of resource	Type of the document (e.g. invoice, report, or memo)

Source: Päivärinta et al. (2002, 3)

After defining the baseline, the researchers analyzed the organizational factors affecting the metadata requirements. In the final definition there were 48 metadata elements. (Päivärinta et al. 2002)

The researchers conclude that the traditional standards do not necessary correspond to the organizational environment. Therefore, organization-specific metadata elements need to be defined in that context. The standards stressing mostly bibliographic metadata should not be adopted in organizations as such. (Päivärinta et al. 2002, 7-8)

4.2 DOCUMENT STRUCTURE STANDARDS

In addition to document metadata standards there are standards for defining document structures. These include ODA (Open Document Architecture) and SGML. The former is intended for transferring documents between document management systems. The latter is based on a declarative markup, and is used in different application areas to mark metadata in documents. The metadata is marked with tags that describe the elements, e.g.

`<title>Document management</title>`. However, it leaves the definitions of the tags to the users; in the previous example the users would first have to define what they mean by "title". (Salminen et al. 1996, 75)

ODA is an internationally standardized (ISO 8613) information architecture for compound documents consisting of text, images, and geometric graphics. It provides a common architecture for sharing information when collaborating via computer networks. The standard defines interchange formats, concepts to represent the structure of the information in a document, and the meaning of a set of formatting parameters. There are two different interchange formats that can be used to interchange constituents of a document among systems: the Open Document Interchange Format (ODIF) an Open Document Language. The former employs Abstract Syntax Notation One (ASN.1), the latter employs SGML to define a representation of the ODA constituents. (Fandert et al. 1992, 729, 734, 736)

The ODA standard is from 1989, and in 1999 there were still no document processing systems with an ODA capability. The market for ODA is not very active, and the adoption of the standard has been slow. (NHS Information Authority, 1999)

SGML is also an ISO standard (ISO 8879). It is not sufficient on its own for DMS integration, as it does not define the metadata elements exchanged. SGML can be used in combination with other standards, though. XML (eXtensible Markup Language)[9] is derived from SGML, and it is widely used for data exchange. For example, RosettaNet messages are in XML format.

4.3 B2B E-COMMERCE STANDARDS

E-business *frameworks* have been developed to enable companies to communicate over the Internet. These frameworks include standards and specifications, which define various things like contents of the messages, communication processes, security, packaging and error handling capabilities for messaging. These features aid in achieving general system interoperability.

[9] http://www.w3.org/XML/

Different frameworks have their benefits and shortcomings, which have been analyzed by Kotinurmi et al. (2003a). They compared and analyzed 15 standard frameworks for exchanging design data between companies. The five most promising frameworks were ebXML (Electronic Business using XML), RosettaNet, OAGIS (Open Applications Group Integration Specifications), PDX (Product Data eXchange) and STEP (Standard for the Exchange of Product Model Data).

Kotinurmi et al. (2003a) define a comparison framework, which identifies the requirements for design document management in networked PD. These requirements are:

- *Processes* that specify activities that are to be carried out in a given order. Standard frameworks specify processes, participant roles in communication and the messages involved. For instance, when a message should be sent and how they are answered.
- PD related *messages*, which specify the allowable format and contents of the message documents exchanged commonly understood between the collaboration partners.
- *Messaging* that specifies secure communication when exchanging the messages over Internet. Messaging specifies the packaging, encrypting and responding to basic network problems, such as a message lost in the delivery. Messaging makes sure that the design documents delivery is non-repudiated.
- Industrial support and usage that are obviously important for the companies involved

They find RosettaNet to be the most suitable framework to use as it covers all the requirements listed. The first requirement is met with the specifications of about 110 processes for information exchange. Secondly, RosettaNet includes DTDs (Document Type Definition), which define the format for the messages, and dictionaries that describe the contents of the different elements. The third requirement is covered with the RosettaNet Implementation Framework (RNIF), which defines how the messages are packaged, signed and sent securely over the Internet.

Finally, RosettaNet is already in use in many companies, e.g. in forecasting and order-delivery processes. Therefore, it will be easier to add new PD processes that use existing infrastructure, such as integration servers for RosettaNet messaging. As large companies push for the standard, small suppliers will have to start using it. An example in Finland is Nokia, which is a member in RosettaNet Executive board. Nokia has announced its commitment to

RosettaNet and aims to have 40 % of its purchasing volume involved in RosettaNet by the end of 2002 (Nokia 2002). This forces Nokia's suppliers to also acquire the needed infrastructure and implement the standard. The standard is described in more detail in the next section.

4.4 ROSETTANET

RosettaNet is an independent, non-profit consortium of more than 500 information technology, electronic components, semiconductor manufacturing, and telecommunications companies. It works to collaboratively create, implement and promote open Internet-based e-business process standards for the electronic sharing of business information. So far RosettaNet has mainly concentrated on supply chain integration. The consortium was founded in June 1998. (RosettaNet 2003)

RosettaNet partner companies participate in the standard's development process. They define and agree on common processes, and develop the XML-based standards to support these processes. The partners provide technical input and human resources for RosettaNet project teams, and commit to implementing the standards in their respective companies. (RosettaNet 2003)

RosettaNet includes data dictionaries, an implementation framework (RNIF), and XML-based business message schemas and process specifications (Partner Interface Processes – PIPs). RosettaNet Business Dictionary (RNBD) and Technical Dictionary (RNTD) provide a common vocabulary for conducting e-business. The business dictionary includes properties used in basic business activities between trading partners. These business properties are made of other business data entities, and fundamental business data entities. For example, a physical address is made up of address lines, a postal code, and a country code. The fundamental business data entities cannot be divided into further components; for example a postal code.

The technical dictionary provides properties for defining products and services. For example, an "integrated services digital network terminal adapter" is defined as: "A device that adapts a computer to a digital ISDN line. Like a modem, it plugs into the serial port of the computer or into an expansion slot."

The RNIF specifies the packaging, routing, and transport of PIP messages and acknowledgments of these messages between trading partners' servers. PIPs are the most important part of the standard concerning this thesis, and they are described in the following section. (RosettaNet 2003)

4.4.1 Partner Interface Processes

PIPs are definitions of business processes between trading partners. They are divided into eight clusters, numbered from 0 to 7. The clusters are further divided into segments noted by letters. Individual PIPs are found within the segments. The eight clusters, the segments in cluster 2 and the PIPs in segment 2C are listed in Table 2. (RosettaNet 2003)

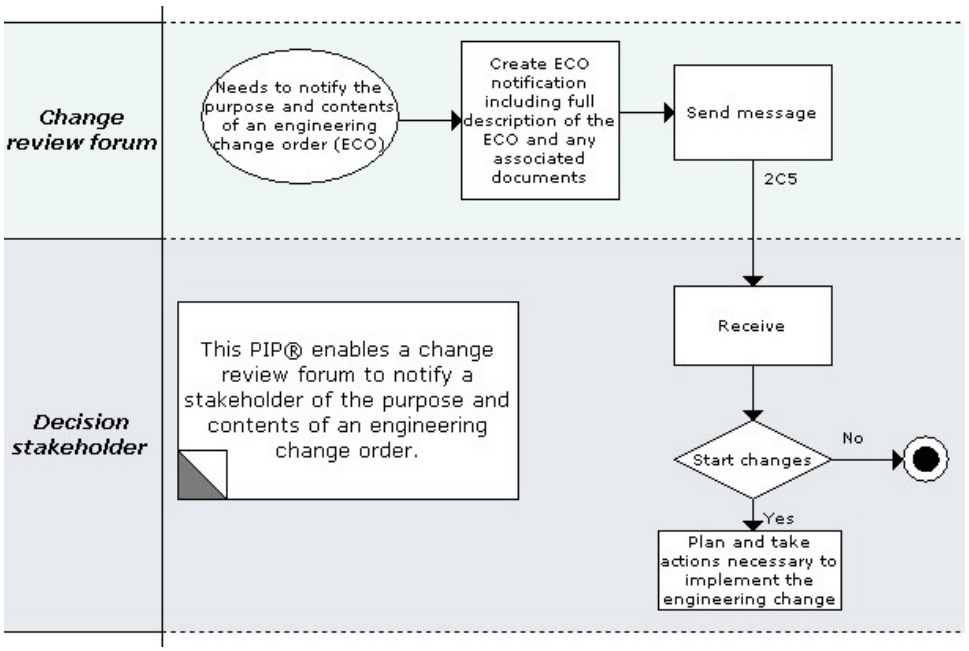
Table 2 PIP Clusters, Cluster 2, and segment 2C

Clusters	
Cluster 0:	RosettaNet Support
Cluster 1:	Partner Product and Service Review
Cluster 2:	Product Information
Cluster 3:	Order Management
Cluster 4:	Inventory Management
Cluster 5:	Marketing Information Management
Cluster 6:	Service and Support
Cluster 7:	Manufacturing
Cluster 2	
Segment 2A:	Preparation for Distribution
Segment 2B:	Product Change Notification
Segment 2C:	Product Design Information
Segment 2D:	Collaborative Design & Engineering
Segment 2C	
PIP2C1:	Distribute Engineering Change Status
PIP2C2:	Request Engineering Change
PIP2C3:	Distribute Engineering Change Response
PIP2C4:	Request Engineering Change Approval
PIP2C5:	Notify of Engineering Change Order
PIP2C6:	Notify of Engineering Change Implementation Plan
PIP2C7:	Request Bill Of Material
PIP2C8:	Notify of Bill Of Material
PIP2C9:	Request Approved Manufacturer List
PIP2C10:	Notify of Approved Manufacturer List

Source: RosettaNet 2003

Each PIP includes a vocabulary and an XML-based message dialog. Individual PIPs include a *message guideline*, a specification and a DTD. The message guideline describes the elements of the DTD and specifies their allowed values. The specification defines the business process the PIP is developed for, the start and end states, partner role descriptions, business process activity controls, the PIP business data, network component design, and business transaction dialog. Figure 6 illustrates the process of PIP2C5: Notify of Engineering Change Order. (RosettaNet 2003)

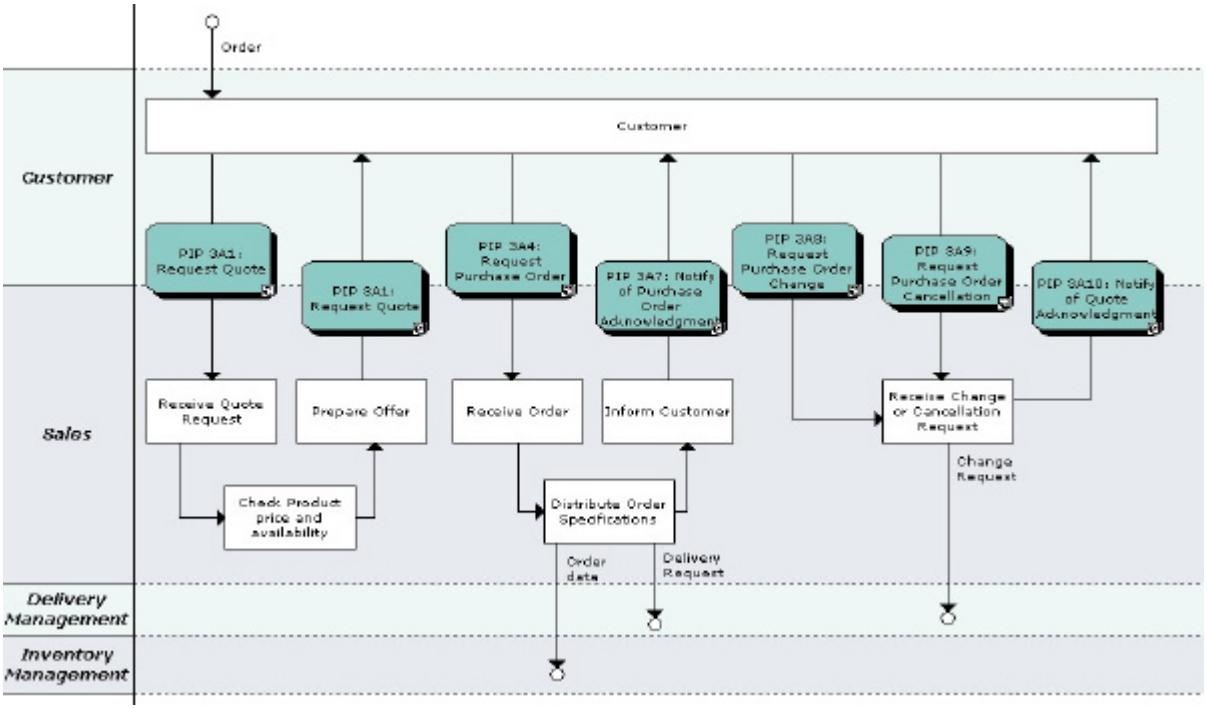
Figure 6 PIP2C5: Notify of Engineering Change Order



Source: RosettaNet 2001

Some PIPs include only one action, i.e. sending one message to a partner. The partner acknowledges receiving the message, which ends the process. An example of such a PIP is 2C5 in Figure 6. There are also two-action PIPs that require an action from the receiver, in addition to the acknowledgment signal. For example, PIP3A1 Request Quote includes two actions: QuoteRequestAction and QuoteConfirmationAction. It is also possible to include many PIPs in a workflow. An example of this is order processing, which could include five PIPs. This example is illustrated in Figure 7. The figure also shows the seller company's private process with the white rectangles. The public process is illustrated with the PIPs in the rounded shadowed rectangles. (RosettaNet 2003)

Figure 7 Order processing



Source: RosettaNet 2003

PIPs are developed continuously, which means, that there are different versions of PIPs used in different implementations. Therefore, it is important to define the version of each PIP used in the systems integration and always use the same one with the same partner. This definition can be done in the TPA (Trading Partner Agreement), which is an agreement between two companies, and addresses the legal issues in electronic information exchange. It defines security issues, like digital signatures; communication issues, such as server addresses and PIPs used; and it includes a non-disclosure agreement.

RosettaNet PIPs can be criticized for being a little ambiguous. Trading partners have to agree on which attributes to use in which context, as the dictionaries are not univocal. It would actually be possible to send all document attributes as an attachment to a PIP message, only including the mandatory fields in the service content, which normally carries the metadata. Also, there is a problem with the use of DTDs. Some of the elements are described as alternatives in the message guideline: one of them has to be included in the PIP message. Still, the DTD defines both elements as optional.

4.4.2 Documents in RosettaNet

Document files are handled as attachments to PIP messages in RosettaNet. For example, Figure 6 shows the creation of PIP2C5, which includes attaching associated documents. However, the PIP business document, Engineering Change Order Notification, includes all the information to enable the creation of an implementation plan: ECO number, affected component identifiers, ECO status, and other information. PIP2C5 DTD includes only a few attributes for attachments. These are listed in Figure 8, and the related guideline information from PIP2C5 message guideline in Table 3. Table 4 describes the business properties related to these elements. Table 5 lists the business data entities and Table 6 the fundamental business data entities related to these elements. Altogether the message guideline includes 318 elements and their descriptions. Due to space limitations they are not all listed here. (RosettaNet 2001)

PIPs also include elements called “fromRole” and “toRole”, which require information of the partner sending the message and receiving the message. This information includes contact information (name, e-mail address, facsimile number, and telephone number), a global partner role classification code, and partner description (business description: global business identifier and global supply chain code, and a global partner classification code (e.g. distributor, OEM)). These are common elements to all PIP messages.

Segment 2D Collaborative Design and Engineering is listed on RosettaNet homepages, but it does not include PIPs yet. The PIPs in Segment 2C are all waiting for validation, so they are not yet implemented either[10]. Even though the PIPs do not include sufficient support for DMS integration, it is possible to use RosettaNet in this context. Actually, we can put all document metadata in one free form text field of the message. In addition, most of the other elements are optional, so we do not need to have this data at hand. This is not an elegant solution, though. Therefore, our aim is to get the data model as a part of the segment 2D PIPs as they are developed. This happens through the industrial partners of the NetData research project. HUT is also a member of RosettaNet, but only the industrial members who provide resources for the organization can really influence the development of new PIPs.

[10] Situation 22.7.2003

Figure 8 Attachment related elements in PIP2C5 DTD

```
<!ELEMENT AttachmentMarkUp (
          attachmentMarkUpRowNew? ,
          attachmentMarkUpRowOld? ,
          GlobalMarkUpTypeCode ) >

<!ELEMENT attachmentMarkUpRowNew
            ( Attachment ) >

<!ELEMENT Attachment (
          description? ,
          GlobalAttachmentDescriptionCode? ,
          GlobalMimeTypeQualifierCode ,
          UniversalResourceIdentifier ) >

<!ELEMENT GlobalAttachmentDescriptionCode
           ( #PCDATA ) >

<!ELEMENT GlobalMimeTypeQualifierCode
           ( #PCDATA ) >

<!ELEMENT UniversalResourceIdentifier
           ( #PCDATA ) >

<!ELEMENT attachmentMarkUpRowOld
            ( Attachment ) >

<!ELEMENT GlobalMarkUpTypeCode
           ( #PCDATA ) >

<!ELEMENT description
           ( FreeFormText ) >

<!ELEMENT FreeFormText
           ( #PCDATA ) >
```

Source: RosettaNet 2001

Table 3 Guideline information for attachment related elements in PIP2C5 Message Guideline

AttachmentMarkUp
Constraint: If GlobalMarkUpTypeCode = 'Add' then attachmentMarkUpRowNew is required and attachmentMarkUpRowOld should not be included. If GlobalMarkUpTypeCode = 'Modify' then attachmentMarkUpRowNew and attachmentMarkUpRowOld are both required. If GlobalMarkUpTypeCode = 'Delete' then attachmentMarkUpRowOld is required and attachmentMarkUpRowNew should not be included.
GlobalMarkUpTypeCode
Entity Instances Add: Add mark up data instances. Modify: Modify mark up data instances. Delete: Delete mark up data instances. No Change: No change to mark up instances
GlobalAttachmentDescriptionCode
Entity Instances Assembly drawings Block diagrams BOM: Bill of Material CAD information: Computer Aided Design Information Schematics Test instructions Assembly/fabrication instructions Quality data: Yield data Sample plan: Quantity of samples in a sample plan Logistics: For example: packaging, carrier requirements, etc. Blueprints
UniversalResourceIdentifier
Constraint: This value MUST follow the Content-ID reference syntax per RFC 2111 and MUST refer to the MIME Content-ID of the attachment.
User Notes: Reference to the content ID of the attached document.
description.FreeFormText
User Notes: This field may be used to supply the file name of the attachment.
GlobalMimeTypeQualifierCode
Entity Instances application/vnd.noblenet-sealer application/vnd.noblenet-directory application/prs.nprend application/vnd.webturbo etc.

Source: RosettaNet (2001)

Table 4 Attachment related business properties in PIP2C5

Name	Definition
attachmentMarkUpRowNew	Describes specifications for an Attachment after a proposed engineering change.
attachmentMarkUpRowOld	Describes specifications for an Attachment prior to a proposed engineering change.
description	A description of a process, thing, action, etc.

Source: RosettaNet (2001)

Table 5 Attachment related business data entities in PIP2C5

Name	Definition
MarkUp	The collection of business properties that describe the nature and type of revisions to be made in an engineering change.
AttachmentMarkUp	The collection of business properties that describe the nature and type of revisions to be made to an Attachment.
Attachment	A business property that describes additional information regarding a product or process.
EngineeringChangeDocumentIdentification	The collection of business properties that describe unique references used to identify documents during the lifetime of the Engineering Change cycle.

Source: RosettaNet (2001)

Table 6 Attachment related fundamental business data entities in PIP2C5

Name	Definition	Data Type	Min	Max	Representation
FreeFormText	Unformatted text.	String	1		
GlobalMarkUpTypeCode	Code identifying the type of mark up.	String	1		
GlobalAttachmentDescriptionCode	Code identifying the type of product attachment, e.g. assembly instructions, block diagrams, test requirements, etc.	String	1		
GlobalMimeTypeQualifierCode	The MIME type. Refer to http://www.iana.org for a list of types.	String	1		
UniversalResourceIdentifier	A network-centric identifier that provides the identity of a resource.	String	1		

Source: RosettaNet 2001

4.4.3 RosettaNet and PDX

PDX is a multipart standard (IPC-2570 series), which focuses on communication of product content information between OEMs, electronics manufacturing services (EMS) providers, and component suppliers. It is based on XML, and it provides a way to describe product content (Bill of Material (BOM), Approved Manufacturer Lists (AML), ECRs, ECOs, and Deviations). It is anticipated that these descriptions will be added to RosettaNet standard for exchanging manufacturing information. (IPC-2571 2001, 1-3)

4.5 SUMMARY

There are a vast amount of standards for documents. These standards have been developed for different purposes, and some of them complement each other. The document metadata standards can be used as a baseline for defining PD document metadata, but they are not sufficient on their own, as they have not been developed for business documents. Document structure standards provide a way to describe document structure. The most important of them in this context is XML, which many other standards, e.g. RosettaNet, use. E-business

frameworks are developed for communication over the Internet. RosettaNet is the most promising of them in the context of this research, and it is consequently used as a starting point for the study.

The RosettaNet standard is under development, and it does not currently include PIPs for PD document exchange. In addition, attachments do not receive much attention in PIP specifications. Through collaboration with the industrial partners of the NetData research project we have a chance to influence the development of these PIPs.
The first part of the thesis has introduced the context of collaborative design, and the needs it creates for document management in partner networks. Also, different aspects of document management and various document standards have been introduced.

The most efficient way to share PD documents is by DMS integration. This requires a common data model for the documents exchanged. This model has to include information of the document's structure and life cycle. Also, common attributes from different companies' internal data models should be included, to make it easier to use the documents in the receiving companies. The core elements of 18 document metadata standards will be used as a starting point for the standard data model. Case companies internal data models, DMS data models, and RosettaNet PIP elements will form the research data, which is used to develop the standard model. We will also have to consider other requirements the use of RosettaNet sets for the standard data model. The data will be modeled using object-oriented concepts: objects and types, attributes, and relationships. Finally, we will build a prototype for DMS integration as a proof of concept.

The next chapters will present the research data and describe the development of the standard data model. The model is described in chapter 6, and the prototype in chapter 7. They are both evaluated in chapter 8, which also includes discussion. Chapter 9 includes conclusions and ideas for future research.

5 RESEARCH DATA AND METHODOLOGY

The previous chapters have described the need for a standard data model for PD documents. Document standardization includes defining the notions and concepts for document management, as well as definitions for document structures. This should be based on a thorough analysis of the current documents and document management practices in an organization. On the basis of this, new document structures and document management practices can be developed. (Salminen et al. 1996, 72)

We use companies' internal data models to represent the current documents they use. DMSs give an idea of document management practices in the organizations using them, as their configurations reflect these practices. I will next shortly introduce the companies participating in the research and describe the gathering of research data. In section 5.2 I will present the development of the data model.

5.1 DATA GATHERING

5.1.1 Companies participating in the research

We have obtained document management information from four companies while developing the data model. The companies are described here with their main business areas and key figures. Names of the companies are not revealed due to confidentiality reasons.

Company A is a global service and engineering company with annual net sales of 5500 million euros. The company employs 35 000 people. Company B is a global consumer electronics company with net sales of 30 billion euros and a personnel of 52 000 people. Company C is a global manufacturer of plastic components. Their net sales amount to 251,9 million euros, and they employ 2000 people. Company D is a solution provider for creating and managing product information. They employ over 100 people, and their turnover is 16,5 million euros. Two of the companies have conducted networked PD projects together.

The companies provided us with the material described in sections 5.1.2 and 5.1.3. In addition, we discussed the preliminary versions of the model with case company representatives. We also discussed the data model with two PDM experts.

5.1.2 Data models

Companies A, B, and C provided us with their internal data models. The models varied on their richness. The simplest included only five attributes for documents, where as the richest described documents with 106 attributes. The smallest set was only a plan, though, so instead we decided to use a model, which is actually in use (data model B). This model describes only one type of documents, though. The companies consider their data models proprietary information, which they do not wish to be published. Therefore, only the result of the analysis of the models is presented in the thesis.

5.1.3 Document Management Systems

We had a chance to get acquainted with the document management systems in companies A, B, C, and D. In addition, we built a prototype around a document management system called EDMS, which is built at HUT. This system is described in more detail in chapter 7.1.2.

The data models and DMS data models are intertwined to some extent. Only data model B was described without reference to the system it is used in. Also, only DMSs without company specific attributes were DMS A and DMS D. Most of the DMSs are very flexible in terms of defining attributes for different objects. Nevertheless, there are also systems that have a fixed data model with a fixed set of attributes. Such a system was not used by any of the case companies, though.

5.2 DEVELOPMENT OF THE DATA MODEL

The metadata definition project was conducted between June 2002 and July 2003. The data model was developed in an iterative manner. The development included analysing both the internal data models received from the case companies, as well as different DMS's data models. The construct was presented to PDM experts and companies' representatives, and it was further developed according to their comments.

5.2.1 Research data analysis

There were more than 200 different attributes found in the data models analyzed. Table 7 presents the attributes found in three or more of these data models. This level was chosen so that company specific attributes are not revealed. The attributes only present in one or two of the models were also not considered relevant for a general data model.

Table 7 Common attributes in the companies' data models

Attribute	Data model A	DMS A	Data model B	DMS B	Data model C	DMS C	DMS D
Document ID			x	x		x	
Name	x	x	x	x	x	x	x
Type	x	x	(x)[11]	x	x		
Creator	x	x		x	x		x
Creation time	x	x		x			x
Version	x	x	x	x	x	x	x
Parent version	x	x		x		x	
File name	x	x	x	x	x	x	
File format	x			x		x	
Modified by	x	x		x	x	x	
Modification date	x	x		x	x	x	
Current state	x	x		x	x	x	x
Responsible person	x		x	x	x	x	
Responsible team	x			x	x		x
Doc description	x	x		x	x	x	x
Comments			x	x	x		
Language	x					x	
Approved by	x		x	x			
Reason	x		x	x	x		
Related docs	x				x		
Item code	x		x	x	x		
Project	x	x	x	x	x		

Some systems use a unique name for documents, which is then also the document identifier. *Document ID* element is listed for those models that list both a *name* and a *document ID*. Some elements were not explicitly listed in the material we received. It is most likely, that the DMS adds these attributes automatically (e.g. *creator* of a document). Also, there can be system attributes, such as *document ID* that are not shown to the user, but are still relevant in the exchange of documents.

The attribute *reason* relates to engineering change requests (ECRs) in three of the four cases it was present (an example value for the attribute: "customer request"). Here it means the reason

[11] Data model B was for one specific document type, so implicitly it includes this attribute

for creating any document, in which case it applies for more than one type (ECR) of documents.

5.2.2 Mapping with RosettaNet

This attribute set was mapped to the RosettaNet PIPs 2A1 (DistributeNewProductInfo) and 2C5 (EngineeringChangeOrderNotification) to find out if these PIPs could be used as such in the document exchange process.

In case of PIP2C5 Notify of Engineering Change Order the compulsory fields to fill in are *changeOriginationDate* and *EngineeringChangeDocumentIdentification*. The first one corresponds to the *creation time* attribute found in three of the four systems analyzed. It is most likely that this attribute is a system attribute, which can be found in all DMSs. If not found, the value of the attribute can be set to be the creation time of the PIP message. *EngineeringChangeDocumentIdentification* is made of an *engineeringChangeOrderIdentifier* or an *engineeringChangeRequestIdenfifier*, and a *GlobalEngineeringChangeStatusCode*. The identifiers are *ProprietaryReferenceIdentifiers*, equivalent to the *document ID* attribute above. The *GlobalEngineeringChangeStatusCode* corresponds to the *current state* attribute. In addition, PIP2C5 includes elements corresponding with the attributes *type, creator, current state, description, reason,* and *item code.*

PIP2A1 ProductInformationNotification includes only one element: *ProductNotice*, which is made of *theNotice* (FreeFormText) and a *GlobalProductIdentifier*, which is the Global Trade Identification Number (GTIN)[12]. This PIP was not designed for DMS integration; therefore it is not very useful in this context.

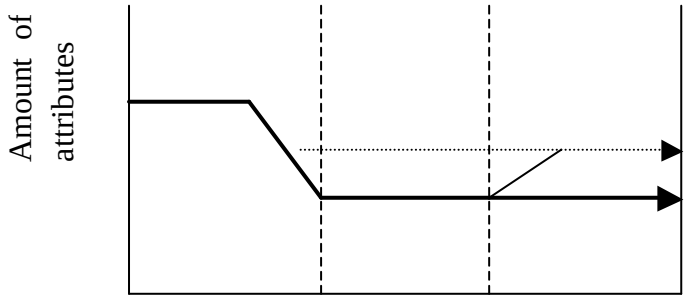
5.2.3 The right attribute amount level

A standard data model brings an additional level to the figure of attribute amount levels in two DMSs, which need to be integrated (Figure 5). If the data models are not converted straight from model one to model two, there can be a situation, in which the standard model includes less attributes than either of the company models and thus results in loosing attributes that exist in both systems. Figure 9 shows this kind of a situation. The bold line shows the resulting amount of attributes in DMS 2 after converting DMS 1 attributes to the

[12] GTIN is a 14-digit number that uniquely and globally identifies a product. It is specified by RosettaNet.

standard model. The dotted line shows the amount that could have been possible to achieve without the standard model (the upper arrow points to the original attribute amount level in DMS 2). This level cannot be achieved in an automated system, since you cannot increase the attribute level without human intervention. Therefore, the line going upwards from the standard level to DMS 2 level is not possible, unless, for example, a person from company two calls someone in company one and asks what the missing attribute values are. This is obviously not an acceptable solution.

Figure 9 Attribute amount level with a standard data model



As a result, we will have to decide on an adequate level of attributes. By defining some of the attributes optional we accomplish minimum and maximum levels for the amount of attributes. The minimum level includes attributes that should be found in all data models and DMSs. Of course, there can always be systems that do not include these attributes, but in such a case some default values or heuristics will have to be used to create the values of those attributes. Another possibility would be to have someone fill in the missing attributes each time a document is sent to a partner, but then the solution would not have removed human intervention from the process. In such a situation the problems of having people in the middle would remain.

The minimum level of attributes includes the attributes found in all three companies[13], either in their data model or the system they use. These are: *name, type, creator, version, file name, modified by, modification date, current state, and project.* In addition, *creation time* is in this minimum set, as it is most likely that it is a system attribute in DMS C, and therefore not

[13] Company D is not considered here, since it is PDM solution provider

explicitly mentioned in the documentation. The attribute *parent version* only exists for child versions of a document, and it is thus not a compulsory attribute. The attributes *responsible person*, *responsible team*, and *doc description* were also found in all three companies, but they were neither mandatory nor system attributes. Therefore, they might not have a value for each document, which would cause problems if they were a part of the minimum set. In addition, there are two other attributes present in all three companies, *reason* and *item code*, but these are type specific attributes. They are only found in "change request" or "drawing" type of documents, and cannot thus be made mandatory for all documents.

The maximum set of attributes includes the common attributes from the companies' data models and DMSs, which can be found in Table 7 (page 47). The set also includes other attributes, which are important to RosettaNet messaging, or otherwise found necessary to include in the standard model.

5.3 SUMMARY

This chapter has described the companies participating in this research, and the data they have provided. The development of the data model included finding the common elements in different data models used, and mapping these elements with RosettaNet. In addition, the right attribute amount level has been discussed.

The attributes and the reasons for adding them to the standard data model are described in the next chapter. The model is also described with object-oriented concepts, all the attributes are described in more detail, and an example of document metadata in the standard format is given.

6 THE STANDARD DATA MODEL

The standard data model includes the attributes listed in Table 7 in the previous chapter. In addition, there are attributes that were otherwise found necessary to include in the model – either because they were needed for RosettaNet messaging or as a result from interviews in the case companies or with PDM experts.

The data model is presented as a UML (Unified Modeling Language; Booch et al. 1998) class diagram in Figure 10. The objects of the data model are classes in the diagram, and the attributes of the objects are attributes of the classes. The relations of the objects are association relationships in the diagram. There is also one generalization relationships, which represent the different types of objects: **partner** and **owner** are specializations of **company**.

The elements are also listed and in Table 9, which gives a description of the attributes, their type and cardinality. Whether the attributes are compulsory or optional is described as the multiplicity of the attribute: compulsory attributes' multiplicity is [1] or [1..*], whereas optional attributes multiplicity is [0..1] or [0..*]. The value lists for attribute values are modeled as enumerations. The Date type refers to RosettaNet DateTimeStamps, a fundamental business data entity described in Table 8.

Table 8 RosettaNet DateTimeStamp

Name	Definition	Data Type	Min	Max	Representation
DateTimeStamp	Specifies an instance in time. Based on the ISO 8601 specification where "CC" represents the century, "YY" the year, "MM" the month and "DD" the day. The letter "T" is the date/time separator and "hh", "mm", "ss.sss" represent hour, minute and second respectively. This representation is immediately followed by a "Z" to indicate Coordinated Universal Time. Informal format: CCYYMMDDThhmmss.sssZ	DateTime	13	19	9(8)X9(6)V9(3)X

Source: RosettaNet (2001)

The relationships are also listed in Table 9, because the XML representation of the data model is created based on that table. The **objects**, *attributes* and relationships are described in the following.

Figure 10 UML class diagram of the standard data model

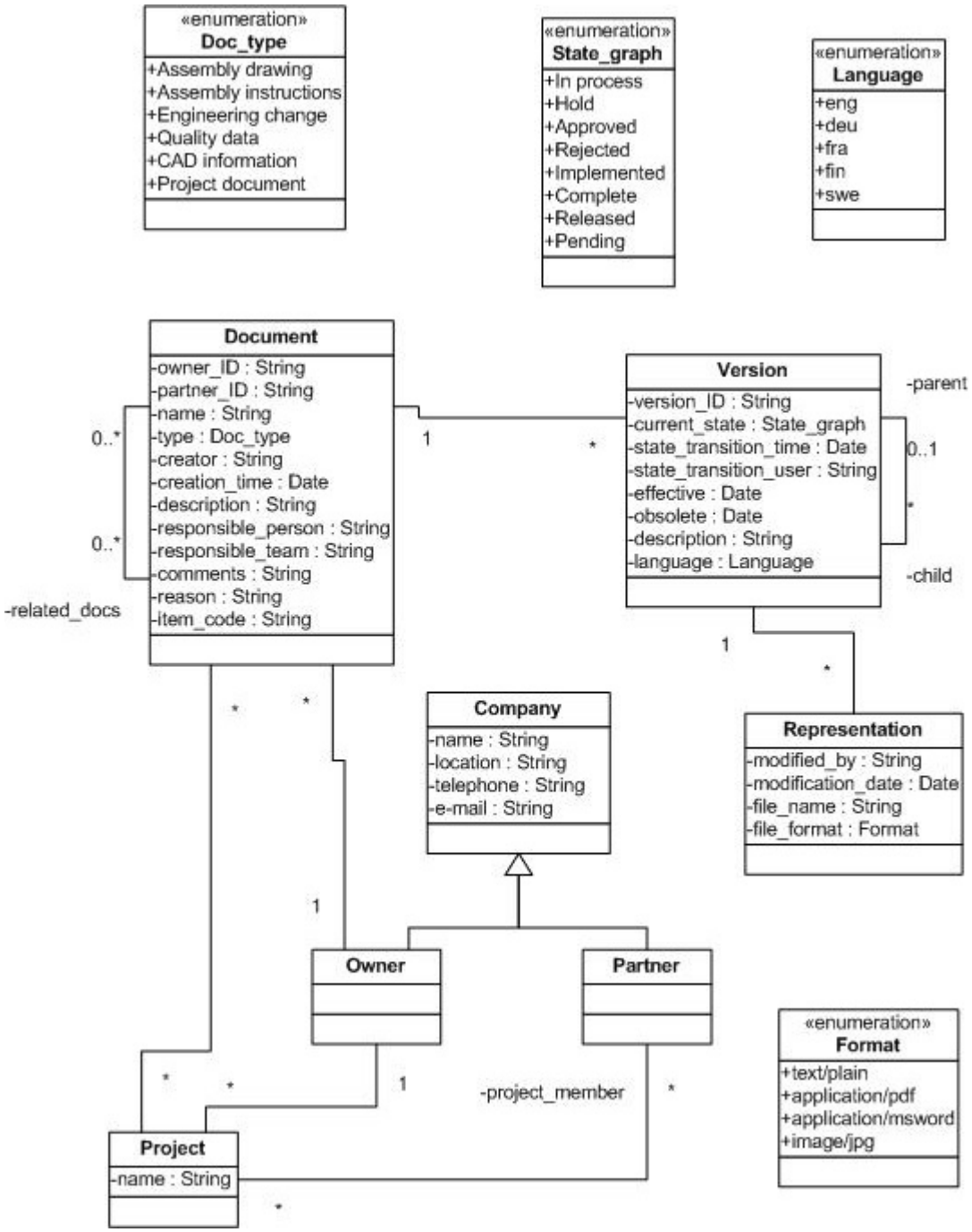


Table 9 The standard data model attributes

Class	Attribute	Type	Cardinality	Description
Document	*owner_ID*	String	[1]	The ID the owner of the document uses
	partner_ID	String	[0…1]	The ID the partner who receives the document uses
	name	String	[1]	Name of the document
	type	Enumeration	[1]	Type of the document
	creator	String	[1]	The creator of the document
	creation_time	Date	[1]	The time the document was created
	related_docs	String	[0…*]	Other documents this document is associated with
	description	String	[0…1]	Description of the document
	responsible_person	String	[0…1]	Person responsible for the document
	responsible_team	String	[0…1]	Team or department responsible for the document
	comments	String	[0…1]	Comments on the document
	reason	String	[0…1]	Reason for creating the document
	item_code	String	[0…*]	Item codes of products the document is associated with
Version	*version_ID*	String	[1]	ID of the document version
	parent	String	[0…1]	Parent version
	current_state	Enumeration	[1]	State of the version
	state_transition_time	Date	[0…1]	The time the state was changed
	state_transition_user	String	[0…1]	The user who changed the state
	language	Enumeration	[0…1]	Language of the document version
	effective	Date	[0…1]	The date the version becomes effective
	obsolete	Date	[0…1]	The date the version becomes obsolete
	description	String	[0…1]	Description of the version
Representation	*modified_by*	String	[1]	Creator of the representation
	modification_date	Date	[1]	The time the representation was created
	file_name	String	[1]	Name of the file
	file_format	Enumeration	[1]	Format of the file
Company	*name*	String	[1]	Name of the company
	location	String	[1]	Location or department of the company
	telephone	String	[1…*]	Telephone number
	e-mail	String	[1]	E-mail address of a contact person
Project	*name*	String	[1]	Name of the project

6.1 THE DATA MODEL ELEMENTS

Document

A **document** has two identifiers: *owner_ID*, which is the ID the owner of the document (e.g. manufacturer) uses, and *partner_ID* used by the partner in question (e.g. supplier). The reason for having two identifiers is that companies do not use common identifiers for products: each company has its own name space for the products or components in question. In addition to the IDs, a document has a descriptive *name*. The *name* of the document is also used as its ID in some DMSs. In such a case, the attributes *name* and *owner_ID* are identical.

A document has a *type*, e.g. “Assembly drawing”. The types are chosen from an enumeration list. This means that the document types used within the organization have to be mapped to these common document types. For example, a “memo” would be mapped to “Project document”. The receiving company will then map “Project document” to a proprietary type that corresponds to it, e.g. “office documents”. Examples for the enumeration for doc_type are shown in the UML class diagram.

Some of the other attributes can depend on the document type. For example, if the type of a document is “Engineering change”, the document needs to have a relationship to related_docs, which are the subject of the change. For example, if a drawing for the cover of a mobile phone needs a change, this change is described in a document whose type is “Engineering change” and which is associated with the original drawing in question. Documents of other types can also be associated with other documents. The relationship is [0..*]-[0..*], which means that one document can be related to many (or none) documents. Those documents can then again be related to other documents. However, we only want to list the documents the one we are sending is related to. The relationship is noted by the attribute *related_docs*, whose value is a list of document identifiers, *owner_ID*s.

The *creator* of a document is the person who created it. *creation_time* is also recorded. In addition to the *creator*, a document can also have a *responsible_person* and a *responsible_team*. The latter refers to the department or team responsible for the document.

A document also has a *description*, which is an optional attribute. It can be used to list attributes from a document management system that are not included in the standard model. Special use of the description attribute can be defined in the TPA. *Comments* is an element for commenting the document; it differs from *description* in that it does not describe the contents of the document but comments them.

Reason is the reason for creating the document. It is most often found in EC documents. *Item_code* is the identifier of the product the document is associated with. Of course, there can be documents not related to any products, and also documents related to more than one product.

Documents have an **owner**, a **company** who is responsible for it. There can only be one owner for a document, but the owner can change during the life cycle of the document. For example, the manufacturer owns the document in the design phase, and the supplier in the production phase. The owner of a document is the only one allowed to make changes to it, i.e. create new versions of it.

Version

A document can have many **version**s. A version, on the other hand, can belong to only one document. A version has an identifier, *version_ID*. The value list for the identifier can be defined in the TPA. A common way to identify versions is to number them 0.1, 0.2, … 1.0, 1.1… Still, even in the few data models we analyzed, there were also many other ways to identify versions.

Parent notes the parent version of the document version. The version can have only one parent version, or none at all. A document version can also have [0..*] child versions. The information of child versions was not found from the companies' data models, though, and it is thus left out of Table 9. The relationship is noted by the attribute *parent* for a **version**, and its value is a version identifier, *version_ID*.

There can be different language versions of a document. Therefore, the *language* of the document version has to be noted. The value is chosen from an enumeration list, which includes the GlobalLanguageCode entities from RNBD. Examples are given in the language enumeration in the UML class diagram.

A state graph defines the possible states a document version can have as its *current_state*. The states used within a company's proprietary state graphs will have to be mapped to the states found in RosettaNet. These states are actually GlobalEngineeringChangeStatusCodes, but we use them for all document types as they cover the different states found in the research data. The states are listed in the State_graph enumeration in the UML class diagram. Their descriptions can be found in Table 10.

Table 10 EC states in PIP2C5

GlobalEngineeringChangeStatusCode
Entity Instances In process: The requested change is in process. Hold: The requested change is on hold. Approved: The requested change is approved. Rejected: The requested change is rejected. Implemented: The requested change has been implemented. Complete: The requested change is complete. Released: The requested change has been released. Pending: The requested change is pending.

Source: RosettaNet

State_transition_time and *state_transition_user* are also recorded, the latter being the person responsible for the state transition. This attribute corresponds with the *approved by* attribute found in the companies' data models. It is extended to cover all state changes; if the state is "approved" the *state_transition_user* is the approver. The state can be changed either by the owner company, or a partner. For example, the partner can check a document after the owner has announced it to be ready. The state of the document is common for the partners, i.e. a document cannot be in different states in different document management systems.

A document version can also have two dates describing its validity: *effective* and *obsolete.* Both are dates: the former is the date the version becomes effective, and the latter the date it becomes obsolete.

There is also a *description* for the document version, which can be used in the same manner as the description of a document if there are version attributes not included in the standard model, but which want to be exhanged. This attribute also corresponds with an *additional information* attribute found in two of the data models.

Representation

A **representation** is the actual file. It is always associated with one version of a document, but there can be many representations for one document version. For example, there can be PDF-representations for documents such as CAD drawings that would otherwise require expensive programs to view.

A representation has attributes for the last time it was modified: *modified_by* and *modification_date.* The *modification date* refers to the original creation of the representation, not to the time it was copied into the DMS. Most of the DMSs do not allow the user to modify this attribute, though. If a partner sent the file, copying it to the receiving system automatically sets its creation time to that moment. Therefore, it might be necessary to save the original creation time (that received in the document metadata) to another attribute; for example, to add the text "original creation time " and the value of the attribute to a description attribute's value in the receiving DMS.

Also the *file_name* and *file_format* are recorded. File format is a separate field from *file_name,* because RosettaNet messages need the MIME type of the attachment as a separate field (see Table 3). Examples of the MIME types are also given in the Format enumeration in the UML class diagram.

Company

A **Company** can be either the **owner** of a document or a **partner** company. For a company its *location* has to be defined, as there can be many divisions of an organization participating in the same product development project. Also the information system, for which the received document is meant, should be stated. Companies might want to have only one integration server for RosettaNet messaging, which receives all the RosettaNet messages. The server should somehow be able to forward the message to the system in question, e.g. PDMS, ERP.

A company also has contact information required in the RosettaNet toRole element: *e-mail* and *telephone.* Telephone number is represented as a string to allow for separating area and/or country codes.

Project

Documents are associated with projects: a document can belong to many projects, and projects can have many documents. The *name* of the project has to be the same in all of the integrated systems: the owner's and the partners' systems.

Projects have one owner, an **owner** company, and many project_members, **partner** companies. This information does not have to be sent to the partners with each message, though. It is assumed that the partners already have this information and manage it with their DMS.

Figure 11 shows an example XML presentation of a document's metadata according to the standard data model developed.

6.2 SUMMARY

This chapter has described the elements of the standard data model. They have been illustrated with a UML class diagram, and an example of document metadata in the standard format has been represented in XML. The model has been validated with a prototype system, which is described in the next chapter. Both the data model and the prototype are evaluated in chapter 8.

Figure 11 A standard document metadata example represented in XML

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
  <Document_metadata>
    <Document>
      <owner_ID>Cover345</owner_ID>
      <partner_ID />
      <name>Cover345</name>
      <type>drawing</type>
      <creator>Jukka Borgman</creator>
      <creation_time>20021122T105047.047Z</creation_time>
      <related_docs />
      <description />
      <responsible_person>Jukka Borgman</responsible_person>
      <responsible_team>NetData</responsible_team>
      <comments />
      <reason />
      <item_code />
    </Document>
    <Version>
      <version_ID>fi.-.1</version_ID>
      <parent />
      <current_state>draft</current_state>
      <state_transition_time>20021122T105231.031Z
      </state_transition_time>
      <state_transition_user>Jukka Borgman</state_transition_user>
      <effective />
      <obsolete />
      <language />
      <description />
    </Version>
    <Representation>
      <modified_by>jborgman</modified_by>
      <modification_date>20021122T105446.046Z</modification_date>
      <file_name>cover345.vtx</file_name>
      <file_format>VTX</file_format>
    </Representation>
    <Owner>
      <name>Helsinki University of Technology</name>
      <location>SoberIT</location>
      <telephone>+35894515799</telephone>
      <e-mail>Jukka.Borgman@hut.fi</e-mail>
    </Owner>
    <Partner>
      <name>Viritys Oy</name>
      <location>Joensuu</location>
      <telephone>+35894515799</telephone>
      <e-mail>Vesa.Makela@viritys.fi</e-mail>
    </Partner>
   </Document_metadata>
```

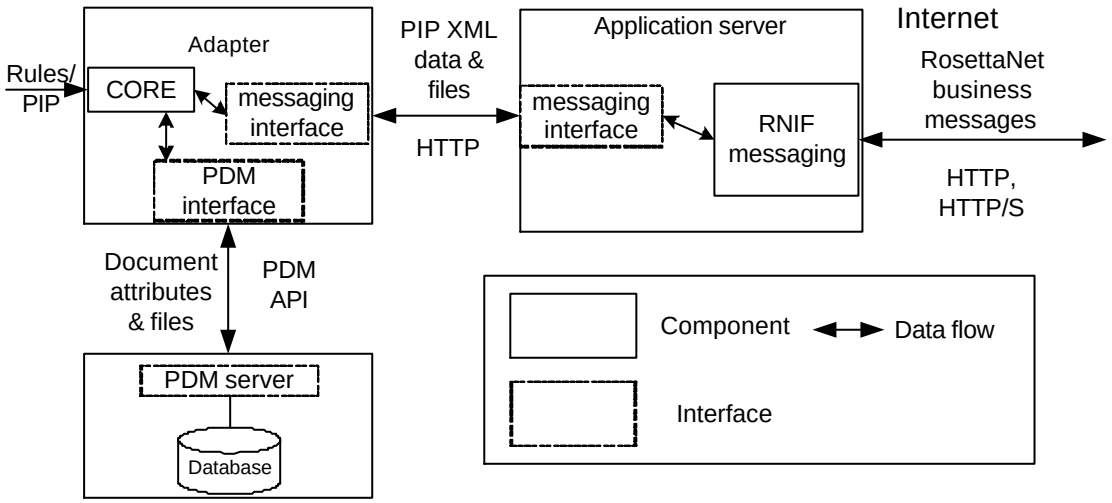
7 PROTOTYPE

We have designed and implemented a prototype for design document management in networked PD processes. The prototype was built for two reasons: first, to test if it is possible to exchange documents between companies with the RosettaNet standard, and second, to see if the standard data model developed in this thesis is adequate for this context. This chapter first describes the architecture of the prototype and then discusses its implementation.

7.1 PROTOTYPE ARCHITECTURE

The prototype is made of three main components: a document management system, an integration server, which takes care of the RosettaNet messaging over the Internet, and an adapter between these two. The architecture is illustrated in Figure 12 and the three components are introduced in sections 7.1.1-7.1.3.

Figure 12 Architecture of the prototype



Source: Kotinurmi et al. 2003a, 7

Parts, if not all, of the functionality of the adapter could have been implemented in the other two components. The reason for making it a separate component is to standardize the integration of legacy document management systems between companies. Business processes change, and the infrastructure should be able to change as well. This means, that a company should be able to change parts of the infrastructure without the change affecting the whole

system. With a common adapter the logic is all in one place, and one should be able to change, for example, the integration server without having the change affect the document management system.

7.1.1 Integration server

The integration server[14] takes care of the inter-organizational messaging over the Internet. The messaging is done according to the RosettaNet Implementation Framework (RNIF). The server also communicates with the adapter: it receives and sends the document metadata and the associated file(s). The server works like a post service: it wraps the service content (i.e. the metadata in XML format and the binary file(s)) in a RosettaNet envelope (*business message*) with proper headers, encrypts it, and sends it to the partner. When the server receives a business message from a partner, it extracts the service content and attachment files from it and sends them to the adapter.

7.1.2 Document management system

The DMS manages the design documents within an organization. The system notifies the adapter when there are changes to the objects in it (e.g. a document version's state is changed). The DMS offers an interface for the adapter to access and modify the objects within the system.

7.1.3 Adapter

The adapter is the "glue" between the other two components. It includes a user interface for defining rules for the delivery process, a rule engine for evaluating these rules, functionality for forming messages to the server of the documents to be delivered, as well as functionality for receiving messages from the server and saving them to the document management system. The interfaces to the other components are designed to be independent of the DMS or integration server used.

The message to the integration server includes the metadata of the document and the respective representation file(s). The metadata sent is the data defined in the standard data model presented in the previous chapter. This data and the file are queried from the DMS, after which the file is posted to the integration server. The adapter constructs an XML

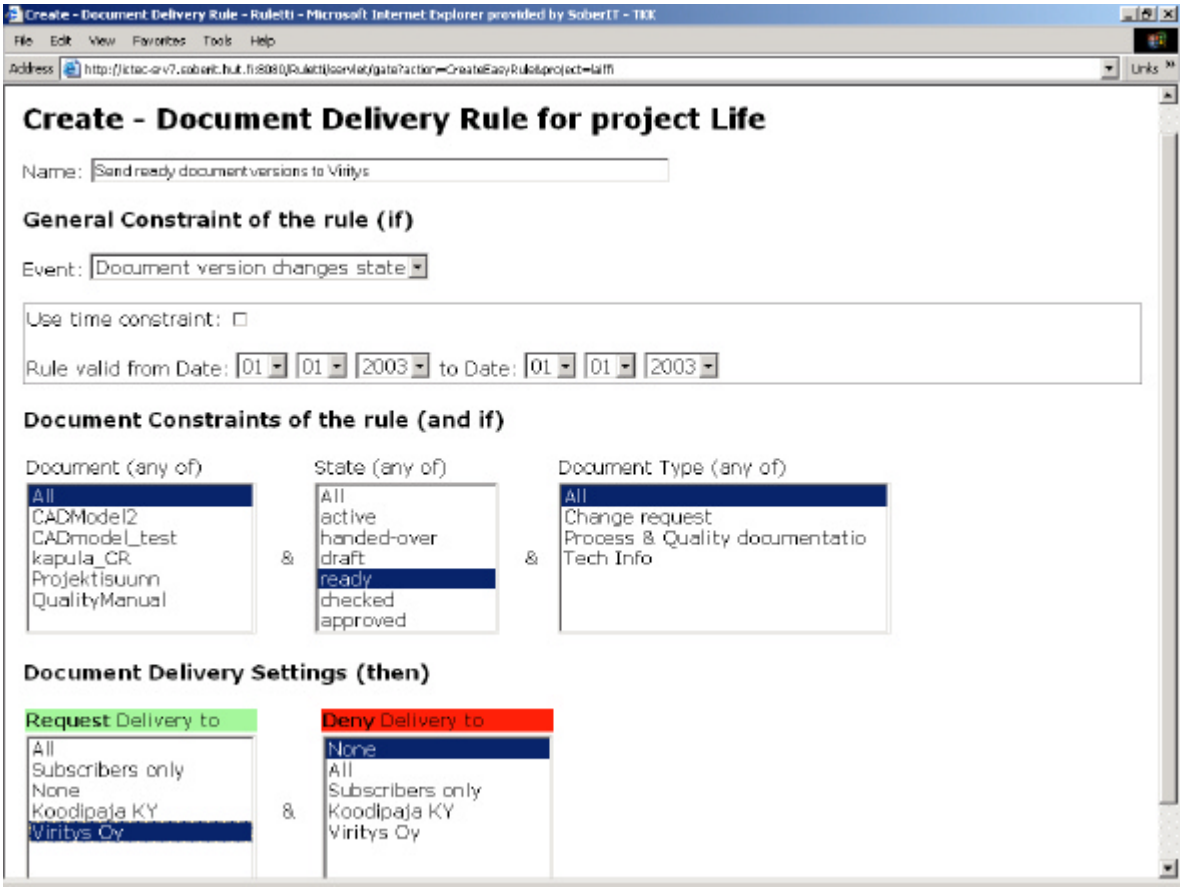
[14] This is the same thing as the application server in Kotinurmi et al. 2003a

message of the metadata, which it then converts to RosettaNet PIP service content form with an eXtensible Stylesheet Language Transformation (XSLT) and sends it to the integration server.

The rules for document delivery are defined with a web-based tool. The adapter evaluates the rules after receiving event notifications from the DMS. The rules are project-specific, therefore only the relevant rules need to be evaluated at a time. If documents should be delivered to partners the adapter decides which PIP to use and constructs the respective PIP message with the document metadata. It then sends the message and the representation file to the messaging server.

The rules have inputs, conditions and consequences. They take DMS objects (e.g. documents, users, projects) as inputs. The rule conditions can be related to, for example, document version state or document type. The consequence defines the action to be taken if the rule applies: normally the consequence would be to send the document to one or more partners. In addition, there can be a date constraint for the rule, which means that the rule applies only on a predefined time period. An example of the rule definition in the adapter's user interface is presented in Figure 13.

Figure 13 Rule definition for the document delivery process



The user interface also includes functionality for manually sending the documents and simulating events from the DMS to see which documents would be sent to partner according to which rules. There is also a delivery manager, which shows delivery information of all the documents sent.

7.2 IMPLEMENTATION

The prototype has been implemented for the most parts. It now enables sending and receiving RosettaNet business messages and quering and storing the information in the DMS. The web-based tool for defining the document delivery rules is ready and tested, as well as the rule evaluation. So far our implementation supports two PIPs: 2A1 Product Information Notification and 2C5 Notify of Engineering Change Order.

The implementation was partly conducted as student assignments on programming and software project courses at HUT. The student groups concentrated on the adapter part. I was a member of both of these groups, and I was also responsible for configuring the document

management system. Research assistant Hannu Laesvuori did all the implementation at the integration server end.

7.2.1 Integration server

The integration server component is built on Microsoft BizTalk server and Microsoft BizTalk Accelerator for RosettaNet. The additional logic is implemented with .NET[15] and COM[16]. The communication with the adapter is organized as a Web Service on both ends. The attachment file is first posted with HTTP (HyperText Transfer Protocol) to the integration server (or from the integration server to the adapter), and then the metadata with reference to the attachment is sent as a SOAP (Simple Object Access Protocol) message to the Web Service at the integration server (or the adapter, respectively).

7.2.2 Document management system

The component used for document management is called EDMS (Engineering Data Management System). It was built by the Product Data Management Group (PDMG) at HUT in a joint project with KONE Elevators. It is built around a rich data model with document versions, subdocuments and multiple representations of document files. The system is in use at KONE, and it is being further developed by the PDMG. (Peltonen 2000, 92-95)
EDMS includes commit procedures that are executed each time an object (e.g. document, user) is modified. These procedures have been configured to send an event notification to the adapter each time there are changes in the database objects, e.g. new documents are created or their states are changed. The event notification includes an identifier of the object concerned.

EDMS offers a Java API called PINJA (Product Data Management API in Java) for accessing, modifying and storing documents. PINJA is also developed at the PDMG. It uses EDMS's server protocol to access the server. All requests refer to the EDMS data model (e.g. documents, subdocuments, document versions). (Peltonen 2000, 140)

In the prototype there are two instances of EDMS servers and two separate databases with different data models to represent two partner companies and the messaging between them.

---

[15] .NET is a set of Microsoft software technologies for connecting information, people, systems, and devices. (Microsoft 2003a)
[16] Component Object Model (COM) is Microsoft's framework for developing and supporting program component objects. (Microsoft 2003b)

There were a few additions needed for the data model used in the EDMS to enable RosettaNet messaging. First, we had to add a new user type, "RosettaNet partner", to differentiate between the person-users within the company and the partner companies. This allows defining common access rights to documents for all partner companies. Also, the partner-users need an attribute *subscribes* to list the PIPs in their TPA. If a PIP is not agreed on between the partners, it cannot be used for messaging.

Secondly, there are attributes defined in the standard data model, but not found in EDMS. We had to add those attributes to the EDMS data model.

7.2.3 Adapter

The adapter is implemented using Java programming language and runs under a Java application server (Jakarta Tomcat) on a Linux server. It has general interfaces to the other two components. The DMS interface has been implemented for EDMS, and the integration server interface for BizTalk. In addition, there is a user interface for rule definition. The rule evaluation in the adapter is based on a rule engine, Drools[17], which provides a Java API and an algorithm called Rete-OO for implementing custom object-related business rules.

7.2.4 Experiences from the implementation

The most difficult part in implementing the adapter was deciding what the rules should use as inputs, and what should be sent to the partners after the rules were evaluated. We decided to ease the task by handling all subdocuments as separate documents, but this of course creates problems at the receiving end, as the documents have the same name, but possibly no other relation to each other. If the parent tag is not empty, i.e. the separately sent document versions have the same parent version, the receiver can infer the subdocument relationship. However, if the receiver's DMS does not have a subdocument concept this relationship has to be handled in some other manner. For example, by adding a comment that states the subdocument relationship.

Different document versions create another problem. Different versions can have different states, for example, the latest version can be waiting for approval (e.g. "pending"), the second latest "rejected", and the third latest "approved". The rules for document delivery can be

[17] http://drools.org

bound to document versions changing state, in which case it is natural to have the rule relate to the version whose state is changed. It has been planned, though, that the rules can be bound to other events or dates as well[18]. If the evaluation of such rules results in the action to send a document it needs to be decided which version of the document to send. Our decision was to always send the latest approved version of a document. In the above example this would mean sending the third latest version. Another possibility would have been to always send the latest version, but we thought this might lead to sending the new versions to partners too early.

Another issue is which representation to send. Primary representations are often in the format of the program they were created with, and might not be viewable with other software. Therefore, they are useless to the receiving partner, unless the partner uses the same software. Secondary representations, in formats like JPG or PDF, would solve this problem, but the receiver is not able to modify the document further. In our implementation we only use primary representations, so the issue remains to be solved in future implementations.

There are many possibilities for triggering the evaluation of rules. These are queries from partner companies, date/time, or changes in the DMS. We decided to implement the last possibility, but there might be a need to implement the first two as well. Also, our user interface now only allows rules to be bound to document or document version state changes, but other possibilities can be needed later (e.g. creating new documents).

We made the assumption, that the owner organization of a document is the only one modifying it. Therefore, the same document (version) will not be sent back to the owner company. This way there will not be problems with having two different versions of the document with the same version identifier, which would be the case if the owner would make a new version of a document after sending it to the partner and the partner would want to make a new version of the document it received.

At the moment there are no PIPs that actually fit our purposes. We have decided to use PIP2C5 Notify of Engineering Change Order for documents whose type is change request and PIP2A1 Product Information Notification for all other documents. After new 2D

[18] This functionality will be implemented later.

Collaborative Design & Engineering PIPs have been defined we will have to add those to the prototype.

The student groups used approximately 1800 hours in the implementation projects of the adapter. The EDMS configuration took approximately 100 hours. Integration of the different components and other additional work on the adapter took another 100 hours, so all together the adapter took about 2000 hours to build.

7.3 SUMMARY

Designing and implementing the prototype system has proved that it is possible to integrate DMSs with RosettaNet. It has also been very beneficial in the development of the standard data model. Information that is necessary for RosettaNet messaging might not have been thought of had it not come up during the implementation. Also the document exchange process became clearer while implementing it. For example, the issue of different document versions would not have been addressed if it had not come up during the implementation.

Both the prototype and the data model are evaluated in the next chapter. The chapter also presents discussion on both constructs.

8 EVALUATION AND DISCUSSION

The main objective of this thesis has been to develop a data model of product development documents for B2B integration. The first research question of the thesis was "What metadata of PD documents needs to be shared in collaborative design networks". Table 7 in chapter 5 presents the common attributes in different companies internal data models and in different DMSs. Table 9 in chapter 6 also shows which of these common concepts need to be transferred with every document and which are optional.

The second research question was "Can DMSs be integrated with the RosettaNet standard". The RosettaNet standard does not include PIPs explicitly designed for this purpose. The DMS concepts do not map well to RosettaNet concepts found in PIPs 2C5 and 2A1. The prototype presented in chapter 7 shows that they can be used for this purpose, though. To make the solution more elegant, the RosettaNet standard has to be further developed to include PIPs designed for this purpose. These PIPs should include the metadata defined in the standard data model (Table 9).

This chapter contains a brief evaluation of the outcome of this research: the standard data model and the prototype system. The evaluation is based on a comparison with previous research, discussions with both case company representatives, as well as PDM experts, and experiences with the prototype. The chapter also includes suggestions for further development of the prototype.

8.1 EVALUATION OF THE DATA MODEL

The data model was evaluated in interviews with document management experts and real users of document management systems in partner companies. Their comments have been used in the development of the model. A preliminary version of the data model was also presented to NetData partner companies' representatives and other interested people from the industry in a public seminar on RosettaNet (Kotinurmi et al. 2003b).

The core set Päivärinta et al. (2002) found in 19 metadata standards was also used as a starting point for this data model. The element set defined seems to follow the one by Päivärinta et al. pretty well. A comparison of the standard model with Päivärinta et al.'s core elements can be seen in Table 11. Only attributes omitted from their set were *keywords,*

publisher, *location*, and *subject*. The first two are related to the information collection or library context of the metadata standards analyzed by Päivärinta et al.. The *location* of the document (file) is not important if the file is attached to the message, as the user of the metadata (the receiving system) does not need to locate the file. Even if the file was not transferred with the metadata, the document ID and the sender information should be enough to infer where the file is located. The *subject* attribute was only found in one of the company data models (for only one type of documents), so it was left out of the standard model.

Table 11 Comparison of Päivärinta et al. (2002) core elements with the standard data model

Päivärinta et al. core elements	**Standard data model equivalent**
Availability	Effective & obsolete
Creator	Creator
Date	Creation_time
Description	Description
Format	File_format
Identifier	Owner_ID
Keywords	-
Language	Language
Location	-
Notes	Comments
Organization	Responsible_team
Publisher	-
Relation	Related_docs
Subject	-
Title	Name
Type of resource	Type

It was expected that the internal data models in different companies vary on the amount of attributes, as well as on the attributes themselves. A bit surprising was how easy it was to find the common set of attributes from the data models. Those attributes that were not included in the standard model were actually almost all only present in one of the data models. From those attributes present in two of the models two (*language* and *related_docs*) were later added to the model, because PDM experts and company representatives suggested their addition.

The vast amount of attributes excluded from the model and present in only one company's internal data model show that the standard data model is only useful in the integration of DMSs. It is not adequate to be used as such as an internal data model. The internal model has to reflect the company specific processes and document management practices. However, the

standard data model should be considered when designing an internal data model. It is advisable to include the attributes found in other companies' data models to the internal model in order to make the integration easier.

The benefit of having a standard data model is that companies do not have to analyze their data models separately with every new partner joining a network. This makes the integration of their DMSs easier and faster. In addition, the model includes a possibility to include other information in the description attributes, if found necessary in some collaboration networks.

The choice of using the same data model for all documents limits its usability for special situations. Attributes, which are obligatory for some document types might not make sense for other types. Therefore, they have to be made optional. This is problematic if, as a consequence, a compulsory attribute is left out. For example, an engineering change request does not make sense without a reference to the document that needs to be changed. This is a drawback of using the same standard model for all documents. It was still considered better than having a number of models for different types of documents.

8.2 DISCUSSION ON THE DATA MODEL ELEMENTS

Naturally, there are other ways in representing a document, its structure, and life cycle. Other possible elements for the data model have been found from literature and interviews with PDM experts. These, and the reasons for not using them in the model are discussed in the following.

The data model includes two identifiers for a document: *owner_ID* and *partner_ID*. It would be simpler to have all partners use the same ID for documents. Changing these name spaces is not considered an option, though, as there can be many partners who all have different name spaces for their documents.

The different document types enumerated in the UML class diagram (Figure 10 on page 52) include the GlobalAttachmentDescriptionCode entity instances shown in Table 3 (page 42). In addition to those, at least "Project document" and "Engineering Change" are needed. These should also be added to the RNBD's entity instances for GlobalAttachmentDescriptionCode. There is also an entity called GlobalDocumentTypeCode in RosettaNet, but these types refer to invoicing documents and are thus not applicable in this context.

Documents could be further divided into subdocuments. These would then have different subdocument versions. To keep the standard model simple, these classes were left out. For PDMSs that have richer data models that include these classes (like EDMS), there has to be another way to describe the structure of the document. For example, the document and its subdocuments could all be represented as different documents. The *related_docs* attribute would then list the subdocuments for the original document, and the original document for its subdocuments.

Document versions can also have editions. For example, different language versions of the same document can be modified in a way that keeps the content the same, but rephrases something. In a situation like this it would not make sense to make a new version, as the content remains the same as in an older version. Therefore the new translation is called a new edition[19]. Again, in order to keep the standard model simple, the possibility of multiple editions is left out.

The attribute *language* was only present in two of the companies' internal data models, but it was added to the standard model after interviews with case company representatives and PDM experts. They pointed out the possibility of having different language versions of the same document, in which case it is important to add the *language* attribute to the standard data model.

The attribute *parent* of a version was not made mandatory, although it was found in all three companies' data models. The reason for this was that the first version of a document does not have a parent version. Another possibility would have been to make the attribute mandatory, but allow it to have an empty value. This might have caused misunderstandings, though, and it was therefore thought better to leave the attribute optional.

Version history can also be relevant to a company, but it is left for the document management system to be taken care of. The standard data model does not forward this information to the partners. Only history information included in the model is the *parent* version of the document version in question.

[19] Example provided by Asko Martio.

Different collaborative companies might want to exchange different attributes, relevant to their PDM systems. It would mean a lot of extra work to define a different set of attributes with each partner, though. Therefore the same set of attributes is sent to each partner. If the partner does not need one or more of the attributes, they can be ignored when mapping to the attributes of the partner's PDM system's attributes. If there are attributes not included in the standard model, but necessary for the collaborative partners, they can mutually agree to include them in a free form text field, such as the *description* of a document. This can be done in the TPA.

The *effective* and *obsolete* attributes were only found in one DMS analyzed, but PIP2C5 includes an obsoleteDate element. The attributes were also considered important in the exchange of documents when working with preliminary information. For example, an engineering change can be sent to partners in advance with a date that tells when the change is effective.

Within a company there can be many people interested in the document in question, but the delivery to those people has to be solved within the private process of the company. For example, an additional document subscription service has been built to EDMS in one company (Peltonen 2000, 164). Their web UI allows a user to subscribe to documents and then checks the database at regular intervals to see if there are new approved versions of those documents. The user is informed of new versions by e-mail, and can then download the documents through the Web interface. Our prototype includes a separate tool for defining rules for the document delivery to partner companies. Therefore, information about document subscribers is omitted from the standard data model.

Projects are usually associated with products. This attribute could be added to the project object, but it does not seem necessary to do so with every document sent. It is assumed that the partners already know with which product each of their projects is associated. The association could also be added to the document object, but there can be documents that are associated with many projects and products. Of course, it is also possible to have projects that have to do with more than one product - or none at all.

8.3 EVALUATION OF THE PROTOTYPE

Tests with the prototype have proven that document management systems integration is possible with the standard data model developed, and using RosettaNet. The tests were conducted in a prototype environment, but the systems used (EDMS and Microsoft BizTalk Server) are also used in industrial environments. Therefore, it is presumable that the concept would also work in an industrial environment.

The prototype has been built for proof-of-concept purposes only. Therefore, there are issues that have not been taken into consideration during its development. These include, but are not limited to handling exceptional situations, control of two-action PIPs, and access control of incoming documents. In addition, the prototype has been implemented for only one DMS and one integration server. As a result, we cannot say whether the interfaces developed would reduce the development time of the adapter for other systems. However, the implementation of the prototype has revealed many issues that would not have been considered otherwise, and it is designed extendable and not information system specific. Therefore, it is presumable that another implementation of the adapter for other servers would take less time.

The prototype has been demonstrated to NetData partner companies' representatives and other interested people from the industry in a public seminar on RosettaNet (Kotinurmi et al. 2003b).

Domazet et al. (2000, 2) have identified requirements for computer systems needed to support virtual teams working on a collaborative PD. These requirements can be summarized as follows:

A. Shared Product Data Repository

A1. Support of fine grain data sharing

A2. Use of a standard product data model

A3. Use of active objects that generate specific events when certain operations are invoked

A4. Users may be anywhere on the Internet and should access data any time

B. Collaborative Workflow Management

B1. Users should be able to specify new activities or modify executing workflows

B2. There is a need to support transactional workflows

B3. Workflow exceptions must be catered for

B4. Activities can be activated or deactivated according to product data states (database events), operations of linked applications (application events) or other events from the environment (external or time events)

B5. Workflow management should be distributed

The computer system we have developed for the same purpose can be evaluated against these requirements. Only difference is that we have concentrated on documents instead of product data in a wider sense (i.e. including items and product structures). Each of the requirements by Domazet et al. in relation to our prototype system is examined in the following.

A1. All data can be shared with our solution, as long as it is represented in documents. In addition, rules whose evaluation results in exchanging those documents need to be defined.

A2. Our prototype uses a standard data model for product development documents. It satisfies this requirement to the extent product data is represented in documents. For items and product structures this requirement is not met, though.

A3. The DMS generates events each time its contents are changed somehow. These events are sent to the adapter, which decides on further actions (whether or not to send the document that was changed, and to which partners) based on rules defined in the UI of the adapter.

A4. Documents are exhanged over the Internet using RosettaNet. In that sense the users can be anywhere on the Internet. They can also access the data anytime, as the relevant data is always available within their own DMS. However, our prototype solution requires implementing the adapter's server interfaces for each company's DMS and integration server. Therefore, the set up takes time, and the data cannot be accessed prior to this set up. In addition, there has to be a TPA between the partners. In that sense, the users cannot access the data any time, but there has to be some setup work done before the collaboration can actually

begin. Of course, some set up work is required for all document exchange solutions, even if it means only creating a user name and password for a web UI to a DMS.

B1. The adapter includes a UI for defining rules for the exchange of documents. This allows ad-hoc changes to be made to the workflows for the parts concerning the exchange of documents.

B2. RosettaNet includes mechanisms to guarantee a reliable process control in cases of network, equipment or software failures. An acknowledgment of receiving a message has to be sent within a certain time period of receiving it. The inter-organizational process is thus reliable. The reliability of the intra-organizational process has received only a little attention in the prototype system, since it is only a proof-of-concept. This issue will need more attention when designing an industrial implementation of the system.

B3. Workflow exceptions need human intervention. The adapter keeps a log of the files sent to partner companies. An administrator should check this log regularly to make sure everything works as planned.

B4. The prototype has two event listeners: DMS event listener and integration server event listener. It is thus capable of reacting to database and application events (the DMS includes the database). The rules can also have time constraints, which means they are only valid on a specified time period. Time events have also been planned for the system, but they have not yet been implemented.

B5. Each company is responsible for defining document delivery rules for their own documents; even if the documents are shared there is still always one owner for each document. In that sense the workflow management is not distributed. RosettaNet processes, on the other hand, are distributed. They flow in two directions, and both partners control their public/private process interface, as well as the public transactions (actions and acknowledgments).

Our prototype covers most of the requirements defined by Domazet et al. (2002, 2). Only issue that comes up with many of these requirements is that our solution only handles documents. Therefore, also items and product structures should be later added to the system.

8.4 FUTURE DEVELOPMENT OF THE PROTOTYPE

The prototype still needs development. One major issue is combining the sending part and the receiving part of the adapter. These two modules have some common functionality, which is now implemented twice. The reason for this is that two separate student groups implemented the modules.

The approach we have taken to DMS integration exchanges document metadata in XML format. The actual file is still sent in its original formats. This means that the recipient will need to have the right tools to view the document contents. However, there are viewer tools, which are often free or very cheap, so the recipient does not necessarily have to acquire expensive CAD software to view the documents. Another way to overcome this problem is by making e.g. a PDF representation of the document, but this can require extra work from the document authors. It is also possible to automatically create a PDF file of the document before sending it to the partner.

8.5 RESEARCH METHOD EVALUATION

Research data was not easy to acquire or to interpret. Different companies and different systems had dissimilar ways to represent the data models. In addition, the DMS models and company models were intertwined in the documentation we received. This should not affect the result too severly, though. If a concept is present either in the DMS model or in the company model, it is presumably accessible when creating the RosettaNet message.

Another issue with the research data is the amount of it. Only considering the data models of three companies and four DMSs is a small sample size. It should be noted, though, that it was rather straightforward to find the common set of attributes from this data. As previously mentioned, the attributes left out of the standard model were mostly found in only one of the internal models. In addition, discussions with PDM experts helped to identify those attributes that need to be added to the model, even though not found in three or more of the models analyzed. Presumably these attributes would be found in other data models, if more companies' internal data models or DMSs' data models were examined.

Using the same DMS to represent two systems is a bit problematic. Even as we used two different data models to represent two different companies, the DMSs are still the same. This

leaves some issues not catered to. For example, if the receiving system has stricter restrictions to attribute values (e.g. stringlengths) than the sender of the message this issue will have to be dealt with in the implementation. To further validate the construct the DMS and integration server interfaces should be implemented for other systems as well. The best level of validation would naturally come from implementing and testing the system in a real environment. This will hopefully become possible in the future.

Evaluating the data model with company representatives may have been less beneficial than at first hoped for. It is not very easy to evaluate the model if one is only shown a UML representation of it. Therefore, I believe the development of the model through experiences with the implementation of the prototype has been more beneficial to the usefulness of the model. However, it has been encouraging to get positive feedback from the company representatives, and their comments have been taken into consideration in the development of the model (e.g. by adding the attribute *language* to the model).

8.6 SUMMARY

Based on a comparison with previous research, and discussions with company representatives and PDM experts the standard data model developed seems to cover the needs for product development document metadata. The prototype developed has proved that DMS integration can be done with RosettaNet. Nevertheless, the prototype still needs further development. The data model could also be further developed with a bigger sample of research data.

The next chapter includes conclusions of this research, and ideas for future work.

9 CONCLUSIONS AND FUTURE WORK

Conducting product development in company networks brings new problems to the companies participating in such processes. This thesis has provided one solution to facilitate the collaboration. The standard data model provides a common way to describe product development documents exchanged in the network. The prototype developed shows that the integration can be done with RosettaNet. Both the data model and the prototype need further development, though, and this is described in the following.

The development of both the standard data model as well as that of the prototype will continue at least for the duration of the NetData research project. This will include analyzing more companies' internal data models, and possibly also other document management systems (DMS). The prototype will also be further developed according to the description in the previous chapter.

Further validation of the standard data model requires testing it in an industrial environment. The prototype's DMS and integration server interfaces should be implemented and tested with other servers to find out if there are issues that need to be taken into consideration, but not revealed in our limited testing.

Future research should also include other product data management functions. The exchange of item, product structure, and configuration information should be taken into consideration. This is also the way the development of RosettaNet and PDX (Product Data eXchange) standards goes.

In our solution document delivery to partners is conducted according to rules defined and evaluatied with a separate tool. We have assumed that a project manager uses the tool and creates rules concerning the different documents, their types or states, and different partners. It might be useful to have some basic rules already defined. These could be used as defaults for new projects, and modified if necessary. What these default rules should be would require research on current document delivery rules in companies participating in collaborative design networks.

The RosettaNet Partner Interface Processes (PIPs) are constantly developed. This research has shown that the existing PIPs are not adequate for document management systems integration, because their data models do not cover the metadata of product development documents. Therefore, new processes for the integration should be defined as new PIPs, and the standard data model should be included in the message guidelines for those PIPs. These PIPs will presumably be added to segment 2D Collaborative design and Engineering.

For example, we noticed that PIP2A1 is not adequate for DMS integration. However, we have chosen to use it in our prototype. It is a very simple PIP, and we can exploit it through the use of a free form text field by writing all the attributes in one string within the same tag. This is not an elegant solution, though.

We also use PIP2C5 in the integration. This PIP works better, as it includes more of the standard data model's attributes, but it is not sufficient for this purpose either. Therefore, we propose adding the standard data model for PD documents to the upcoming 2D PIPs.

REFERENCES

Amami, M. & Beghini, G. 2000. Project Management and Communication of Product Development Through Electronic Document Management. *Project Management Journal*. Vol. 31, No. 2, 6-19.

Anttila, J. 2001. *Dokumenttien hallinta.* IT Press, Helsinki.

Banerjee, S. & Kumar, R.L. 2002. Managing Electronic Interchange of Business Documents. *Communications of the ACM*, Vol. 45, No 7. pp 96-103

Benefon. 2002. *Benefon and Elcoteq into a deal to jump-start Elcoteq Design Center*. Press release 1.7.2002.
http://www.benefon.com/press_investors/releases/2002/01_07_2002_b.htm

Booch, G., Rumbaugh, J. & Jacobson, I. 1998. *The Unified Modeling Language User Guide.* Addison-Wesley, Boston.

Borgman, J. & Sulonen, R. 2003. A Case Study of the Impacts of Preliminary Design Data Exchange on Networked Product Development Project Controllability. To appear in *Proceedings of the International Conference on Engineering Design*, Stockholm, August 19-21, 2003.

Burnett, K., Ng, K.B. & Park, S. 1999. A Comparison of the Two Traditions of Metadata Development. *Journal of the American Society for Information Science*, Vol. 50, No. 13. pp 1209-1217. In Päivärinta et al. (2002).

Cartwright, J., Cluster 2 & 7 Product Manager, RosettaNet. 2002. *Product Life Cycle Collaboration.* Presentation. 16.3.2002

Chen, S., Tyrväinen, P. & Salminen A. 1998. Integration of Digital Structured Documents in Virtual Manufacturing Organizations. In *Proceedings of the 31st Hawaii International Conference of System Sciences*. Vol 2. pp 195-203

Cooper, R. G. 1993. *Winning at new products: Accelerating the process from idea to launch.* Perseus Books, Cambrigde, USA.

Domazet, D.S., Yan, M.C., Calvin, C.F.Y., Kong, H.P.H. & Goh, A. 2000. An Infrastructure for Inter-Organizational Collaborative Product Development. *In Proceedings of the 33rd Hawaii International Conference on System Sciences.*

Fandert, H., Fischer, K. & Kämper, J. 1992. The Open Document Architecture: From Standardization to the Market. *IBM Systems Journal.* Vol. 31, No. 4. pp 728- 754

Glass, R. 1995. A Structure-Based Critique of Contemporary Computing Research. *J Systems Software* 28. pp 3-7

Hasselbring, W. 2000. Information System Integration. *Communications of the ACM.* Vol. 43, No. 6. pp 32-38

Helms, R.W. 2002. *Product Data Management as enabler for Concurrent Engineering*. Dissertation. Universiteitsdrukkerij, Eindhoven University of Technology.

IPC-2571. 2001. *Generic Requirements for Electronics Manufacturing Supply Chain Communication – Product Data eXchange (PDX).* IPC, Northbrook, Illinois, USA.

Jokela, S. 2001. *Metadata Enhanced Content Management in Media Companies.* Dissertation. Acta Polytechnica Scandinavica, Mathematics and Computing Series No. 114. The Finnish Academies of Technology, Espoo.

Karjalainen, A., Päivärinta, T., Tyrväinen, P. & Rajala, J. 2000. Genre-Based Metadata for Enterprise Document Management. In *Proceedings of the 33rd Hawaii International Conference of System Sciences*. January 4-7, Maui, Hawaii.

Kontio, J. 2001. *Software Engineering Risk Management: A Method, Improvement Framework, and Empirical Evaluation*. Dissertation. Suomen Laatukeskus, Helsinki.

Kotinurmi, P., Borgman, J. & Soininen, T. 2003a. Design Document Management in Networked Product Development Using Standard Frameworks. To appear in *Proceedings of the International Conference on Engineering Design*, Stockholm, August 19-21, 2003.

Kotinurmi, P., Laesvuori, H., Jokinen, K., Borgman, J. & Penttinen, J. 2003b. *RosettaNet: tekniikat, implementointi ja haasteet.* Seminar at SoberIT/HUT on 19.5.2003.

Loch, C.H. & Terwiesch, C. 1998. Communication and Uncertainty in Concurrent Engineering. *Management Science* Vol. 44. No 8. pp 1032-1048.

Lublinsky, B. 2002. Approaches to B2B Integration. *EAI Journal.* February 2002. pp 38-47. http://bijonline.com/PDF/B2BApproachesLublinksy.pdf

McDermott, C. & Handfield, R. 2000. Concurrent Development and Strategic Outsourcing: Do the Rules Change in Breakthrough Innovation? *Journal of High Technology Management Research*, Vol 11. No 1. pp 35-57

Meier, J. & Sprague, R. 1996. Towards a Better Understanding of Electronic Document Management. In *Proceedings of the 29th Hawaii International Conference on System Sciences*

Microsoft 2003a. *Defining the basic elements of .NET* http://www.microsoft.com/net/basics/whatis.asp, 31.7.2003

Microsoft. 2003b. *Microsoft COM Component Object Model Technologies* http://www.microsoft.com/com/default.asp, 31.7.2003

Murphy, L.D. 1998. Digital document metadata in organizations: roles, analytical approaches, and future directions. In *Proceedings of the Thirty-First Hawaii International Conference on System Sciences*, Vol. 2. pp. 267-276.

Nokia. 2002. *Nokia continues to drive the implementation of RosettaNet standards.* Press release 26.2.2002. http://press.nokia.com/PR/200202/850008_5.html

NHS Information Authority. 1999. *The NHS IM&T Standards Handbook.* http://www.nhsia.nhs.uk/napps/step/pages/ithandbook/front.htm

Paasivaara, M. & Lassenius, C. 2001. Communication in New Product Development Networks - A Case Study. In *Proceedings of the 8th International Product Development Management Conference*, Enschede, the Netherlands, June 11-12, 2001. pp. 711-725.

Peltonen, H. 2000. *Concepts and an Implementation for Product Data Management.* Dissertation. Acta Polytechnica Scandinavica, Mathematics and Computing Series No. 105. The Finnish Academies of Technology, Espoo.

Peltonen, H., Martio, A. & Sulonen, R. 2002. *PDM – Tuotetiedonhallinta.* IT Press, Helsinki.

Päivärinta T., Tyrväinen, P. & Ylimäki, T. 2002. Defining Organizational Document Metadata: A case beyond standards. In *Proceedings of the Xth European Conference on Information Systems (ECIS)*, Gdansk, Poland, June 6-8 2002.

Päivärinta T. & Tyrväinen, P. 1998. Documents in Information Management: Diverging Connotations of ”a Document” in Digital Era. In *Proceedings of the 9th Information Resources Management Association International Conference*, Boston MA, IdeaGroup, May 17-20, pp 163-173. In Tyrväinen & Päivärinta 1999.

RosettaNet. 2001. *PIP™ Specification. Cluster 2: Product Introduction. Segment C: Engineering Change Management. PIP2C5: Notify of Engineering Change Order.*

RosettaNet. 2003. http://www.rosettanet.org, Referenced on 5.8.2003

Saarela, J. 1999. *The Role of Metadata in Electronic Publishing*. Dissertation. Acta Polytechnica Scandinavica, Mathematics and Computing Series No. 102. The Finnish Academies of Technology, Espoo.

Salminen, A., Lehtovaara, M. & Kauppinen, K. 1996. Standardization of Digital Legislative Documents. A Case Study. In *Proceedings of the 29th Annual Hawaii International Conference of System Sciences*. pp 72-81

Sprague, R.H. 1995. Electronic Document Management: Challenges and Opportunities for Information Systems Managers. *MIS Quarterly* Vol. 19, No. 1. pp 29-49

Tekniikka & Talous. Tutkimus ja tuotekehitys Plussa. 2003. *Metso Paper ulkoistaa osakokonaisuuksia*. 10.6.2003 p 17

Terwiesch, C. & Loch, C.H. 1999a. Measuring the Effectiveness of Overlapping Development Activities. *Management Science* Vol. 45 No. 4. pp 455-465

Terwiesch, C. & Loch, C.H. 1999b. Managing the Process of Engineering Change Orders: The Case of the Climate Control System in Automobile Development. *Journal of Product Innovation Management* 1999:16. pp 160-172.

Tyrväinen, P. & Päivärinta, T. 1999. On Rethinking Organizational Document Genres for Electronic Document Management. In *Proceedings of the 32nd Hawaii International Conference on System Sciences.*

Ulrich, K.T. & Eppinger, S.D. 2000. *Product design and development.* McGraw Hill, London.

Yang J. & Papazoglou, M.P. 2000. Interoperation Support for Electronic Business. *Communications of the ACM* Vol. 43 No. 6. pp 39-47

Zirger, B.J. & Hartley, J.L. 1996. The Effect of Acceleration Techniques on Product Development Time. *IEEE Transactions on Engineering Management*, Vol. 43, No 2. pp 143-152