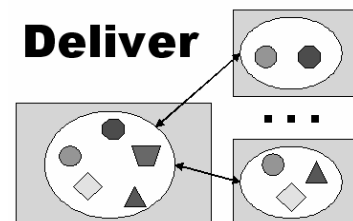


Software Release and Deployment at Exact

A Case Study Report

Drs. Remy Jansen
Dr. Gerco Ballintijn
Prof. Dr. Sjaak Brinkkemper

Centrum voor Wiskunde en Informatica
Kruislaan 413
NL-1098 SJ Amsterdam
<http://www.cwi.nl/projects/deliver/>
deliver-list@cwi.nl



1	INTRODUCTION.....	3
1.1	DELIVER.....	3
1.2	EXACT SOFTWARE	3
1.3	E-SYNERGY	5
1.4	RESEARCH GOAL.....	7
1.5	MAIN CONTRIBUTIONS	7
1.6	REPORT OVERVIEW	7
2	RESEARCH APPROACH.....	9
2.1	DEFINITIONS AND CONCEPTUAL MODEL.....	9
2.2	DELIVERY MODEL.....	9
2.3	INTELLIGENT SOFTWARE KNOWLEDGE BASE.....	10
2.4	RESEARCH QUESTIONS	11
2.5	RESEARCH METHODS	11
3	EXACT SOFTWARE.....	13
3.1	ORGANISATIONAL STRUCTURE	13
3.2	PRODUCT STRUCTURE.....	13
3.3	TOOLS	18
3.4	E-SYNERGY	19
3.5	GLOBAL WORKFLOW	22
3.6	DEVELOPMENT WORKFLOW.....	23
3.7	RELEASE WORKFLOW	24
3.8	DEPLOYMENT WORKFLOW.....	25
4	ANALYSIS	29
4.1	STRONG POINTS	29
4.2	POINTS FOR IMPROVEMENT	31
5	SUGGESTED IMPROVEMENTS.....	34
5.1	TARGET DEPENDENCIES	34
5.2	DEPLOYMENT MANAGER	35
5.3	CHANGE SETS.....	35
5.4	AUTOMATIC PULL MECHANISM	35
5.5	ROLLBACK	36
5.6	UPDATER FEEDBACK.....	36
5.7	GENERAL IMPROVEMENTS	36
6	CONCLUSIONS	37
7	REFERENCES.....	38
8	APPENDICES.....	40
8.1	METRICS	40
8.2	INTERVIEWEES	41
8.3	EXACT SOFTWARE PRODUCT LISTING.....	42

Abstract

For vendors of product software it is becoming more and more difficult to manage and control the software configurations of all their users at the customer's site. It is labour intensive and error-prone to (semi)automatically register detailed lists of the software artefacts in use by each customer. To alleviate this problem the Deliver project proposes an Intelligent Software Knowledge Base that contains all facts about all artefacts together with their relevant attributes, relations and constraints. In this way, high-quality software configurations can be calculated automatically from a small set of key parameters. It also becomes possible to pose what-if questions about necessary or future upgrades of a customer's configuration. This document describes a case study performed at Exact Software into the processes of release and deployment. The results of the case study are presented, existing of process descriptions of the development, release and deployment processes at Exact Software, a comparison to the Intelligent Software Knowledge Base, and an analysis of the results.

1 Introduction

In this first chapter of the case study report a description is given of the Deliver project, followed by a general description of Exact Software. A detailed description is given of the e-Synergy product. This description is followed by the research goal, the main contributions and this chapter finishes with an overview report.

1.1 Deliver

The Deliver project studies the delivery, deployment and maintenance phases of software products. The research project is funded by NWO Jacquard and is based at the Centrum voor Wiskunde en Informatica (CWI). The focus of the project is as follows [1]:

“For vendors of product software it is becoming more and more difficult to manage and control the software configurations of all their users at the customer's site. It is labour-intensive and error-prone to (semi-) automatically register detailed lists of the software artefacts in use by each customer. To alleviate this problem the Deliver group proposes an *Intelligent Software Knowledge Base (ISKB)* that contains all facts about all artefacts together with their relevant attributes, relations, and constraints. In this way, high-quality software configurations can be calculated automatically from a small set of key parameters. It also becomes possible to pose *what-if* questions about necessary or desired upgrades of a customer's configuration.

Managing software configurations is, however, only part of the story. They still have to be delivered to customers. To facilitate this delivery, the Deliver project studies how the computed difference between an existing configuration and a desired configuration can be used for the *Web-based Delivery* of upgrades, furthermore, delivery protocols and implementation are studied. The results of the project are scientific publications, case studies and, in collaboration with industry, prototype tools.”

The main aim of Deliver is to ease software release and deployment effort by managing software knowledge explicitly. The main areas of research for the Deliver project are configuration management, software products, and software deployment. The main focus of the Deliver group lies on large component based product software, such as ERP (Enterprise Resource Planning) systems. To manage this kind of software effectively, it is essential to carefully administrate the specific combination of standard components, tailored components, and customer specific components. The Deliver group offers an external assessment of release and deployment processes within the Software industry. These case studies supply the Deliver research project with knowledge about the problems in the industrial fields of release and deployment of software.

1.2 Exact Software

Exact Software is a manufacturer of software for accounting, human resource management (HRM), customer relationship management (CRM), e-business, and enterprise resource planning (ERP), based in Delft, the Netherlands (see the list of products in Appendix 8.3). Since its establishment in 1984, Exact Software has developed into one of the world's premier companies delivering business software solutions to medium-sized organizations. Software created by Exact Software is now used in 126 countries and available in 26 languages. Exact Software Holding N.V. is listed on the Euronext Amsterdam Stock Exchange since June 1999. Exact Software serves over 160,000 customers in 126 countries and is based in 25 countries.

In 2003 the number of employees at Exact Software was about 2025. Of these 2025 people 20% (see Table 1) are active in the development of software. These 400 developers were spread out over the 31 Exact Software locations. International Development was the largest part with 180 employees in Kuala Lumpur working on Globe, and 20 employees in Delft, working on e-Synergy. The other employees are divided among different departments, such as Exact Software Custom Solutions, Exact Software Client Server, Exact Software Grote Beer, or Exact Software Client Server. 60% Of Exact Software income is from service and support contracts and only 40% from software sale.

Exact Software is relevant to the Deliver group for a number of reasons. To begin with, Exact Software is a successful company in the area of business applications, such as enterprise resource planning software and office automation software. Secondly, Exact Software produces different software products in different locations and is coping with a customer base of over 160,000 customers. Finally, Exact Software claims to have covered all problem areas of the processes of release and deployment with the use of their proprietary tools.

Table 1 Employees per Division (percentage of 2025 average)

	2001	2002	2003
Support	29%	28%	27%
Services	14%	14%	13%
Research and Development	19%	19%	20%
Sales and Marketing	18%	21%	22%
Finance and Administration	7%	7%	7%
Staff	9%	7%	7%
General and Management	4%	4%	4%
Total			100%

This table shows the division of personnel in Exact Software in percentages. The total workforce in the year 2003 was 2,025. Exact is mainly focussed on two products, e-Synergy and Globe. Some statistics about development, quality assurance can be found in Table 2.

Table 2 Quality Assurance, Development, and Release and Deployment personnel in fte's (2004)

	e-Synergy	Exact Software
Quality Assurance	4	96
Development	22	249
Release and Deployment	3	15

Table 2 shows the amount of full time personnel active in different development departments. Release and deployment activities are only the activities related to development and do not include logistics.

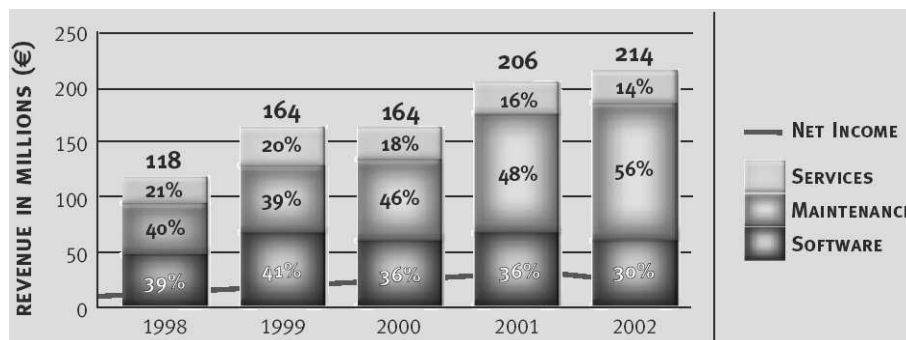


Figure 1 Revenue Generated by Departments from 1998 to 2002

Finally, Figure 1 shows the development of revenue over the years 1998 to 2002. The graph in Figure 1 shows that revenue out of software is quite stabilised around 50 million euros. Maintenance, however, resulting from support contracts, has grown considerably from 40% in 1998 to 56% in 2002 and to approximately 60% in 2003. The increase in maintenance revenue can be explained by the fact that most customers keep using Exact Software products after the first license expires and need to buy new licenses.

1.3 e-Synergy

e-Synergy is a software product developed and marketed by Exact Software. e-Synergy supports the e-business approach, where all business processes are handled electronically. e-Synergy is based on the One-X architecture (see [7] and Section 3.4.2) where all business information is entered only once and globally integrated with all related information. All information is stored and maintained in a central database that can be queried through a web interface. The statistics for the e-Synergy product can be found in Table 3. More information on the e-Synergy product can be found in Section 3.4.

1.3.1 E-Synergy Product Description

The e-Synergy suite is made up of seven (theoretically optional) modules. The seven modules are

- **e-Documents**

e-Documents is the document management system for all e-Synergy users. The e-Documents module enables version control, search functionality, publish functionality, etc.

- **e-CRM**

The e-CRM module enables customer relationship management. The e-CRM module links customers to contracts, service calls, commercial activities, etc.

- **e-Logistics**

e-Logistics is the product data management module of e-Synergy. The e-Logistics module enables an organisation to design a product, generate bills of materials, assemble products, and specify the relationships among product components. More information on e-Logistics can be found in Section 3.2.

- **e-Project**

e-Project is an extension to e-Workflow that supports workflow as parts of projects. It can also handle cost information for e-Financials.

- **e-HRM**

e-HRM is the human resources management module of e-Synergy. The e-HRM module registers all information on employees, their security levels, their financial relations to the company, etc.

- **e-Financials**

The e-Financials module supports all financial reports. It is possible to constantly overview the financial information from e-Financials. This enables real-time financial reporting.

- **e-Workflow**

e-Workflow is the workflow and service call module for e-Synergy. It keeps track of all service requests and other workflow items for employees of the company.

Each module supports a number of specific business processes. It is therefore possible for customers to order only some of the modules that are applicable to a specific customer. Most customers, however, order at least the e-Workflow module, since it is central to the e-Synergy architecture (see Figure 2), and the e-Documents module. Furthermore, each module can be purchased and implemented to work with other Exact Software modules. The e-Synergy application runs on a web server (IIS) and a database server (MS SQL Server). The application is programmed mostly in ASP and Visual Basic with some additional C++ code. The full e-Synergy product consists of about 600.000 lines.

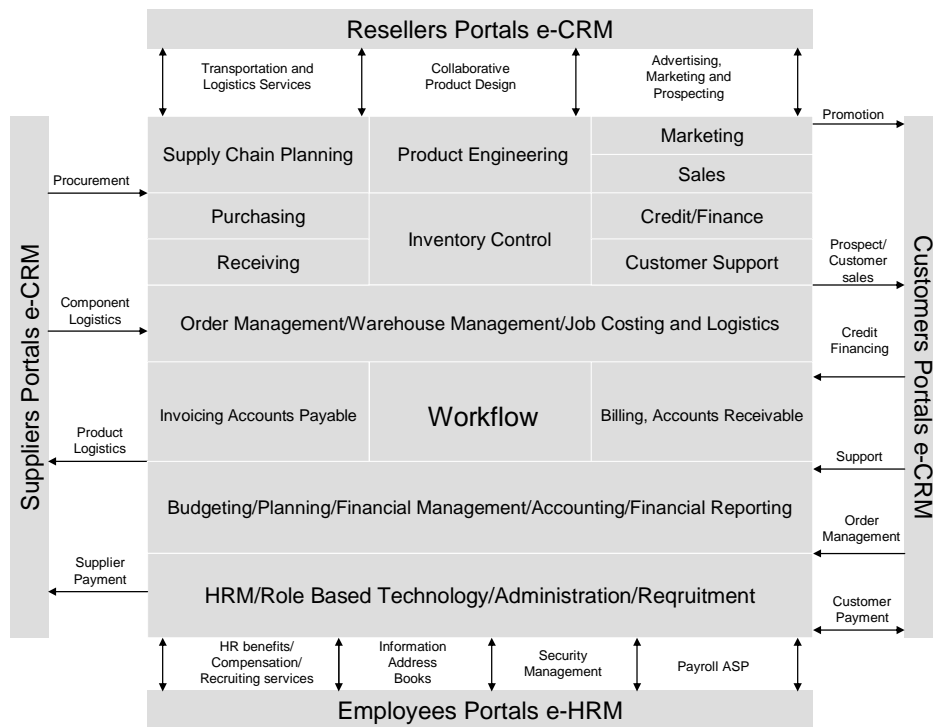


Figure 2 Exact e-Synergy Product Structure

Figure 2 shows the e-Synergy product structure. Four different parties can access the application, the customers, the employees, the resellers, and the suppliers. Each group gets their own view of the application since each group wishes to perform different operations. The combined modules in the centre of the picture comprise the seven modules and the underlying data model.

An obvious strength of the e-Synergy product is the fact that all information is linked in the database through the One-X architecture. The fact that all information can be found by clicking through the system makes the system intuitive to work with. As an example, it is possible to look up a person. From that point on, it is possible to see when she is available, what documents she has seen most recently, what workflow she is responsible for, the status of that workflow, what projects she is responsible for, what customer she is dealing with, and what her e-mail address is. A main feature of e-Synergy is the reporting function. It is possible to generate a report of any data through e-Synergy. This report facility makes it possible to create real time financial reports or any other report on information stored into the database. The linking of data, allowing users to access information arbitrarily is called arbitrary data access. The counterpart of arbitrary data access is central data access which assumes one starting point for each different object of data.

1.3.2 The usage of e-Synergy within Exact Software

Exact Software uses e-Synergy to support their business process. Each employee accesses e-Synergy to see their workflow and uses e-Synergy as a store for their documents. Exact Software experiences the use of products developed internally as a success factor for the quality of their products. As an example, the Exact Software POS (point of sale) software is used in the office cafeteria, all employees use e-Synergy, and Exact Globe is used at their financial departments.

e-Synergy is also used by the development departments. The e-Logistics module functions as a repository for all the software developed within Exact Software. Also, all workflow is handled through e-Synergy, with their own customised workflow templates, such as bug reports and functionality requests. Workflow is also grouped into projects, stored in the e-Project module. This internal use of the e-Logistics module for development results in the fact that the source code for e-Synergy itself is stored in the e-Synergy product. More explanation about the e-Synergy product can be found in Section 3.2.1.

e-Synergy is used for the spread of information through different portals by Exact Software. Exact Software uses an employee portal, a customer portal, a supplier portal, a reseller portal and a publicly accessible website. These portals are all used to keep associates up to date about developments within Exact Software.

All products in Exact Software are managed by a product manager. The product manager is responsible for the product and manages all senior developers. A senior developer stands at the head of a team of three to six developers. Senior developers are usually responsible for one or two modules of a complete product. Each group of developers is supported by a quality assurance engineer, although normally this is not one specific person.

Approximately twenty people, five groups of 3 to 6 developers, are working on e-Synergy in Delft. Each group of developers is responsible for one or two of the modules that belong to the product as described in 1.3.1. Their main tasks are to process bug reports and functionality requests (addition of functionality). These developer groups also provide information to support and customization departments.

1.4 Research Goal

The primary research goal of this case study was to examine a (partial) real-life solution to the problem areas of the Deliver project, to explore and describe these solutions and to propose possible process improvements to Exact Software. The case study took place in the form of interviews, software study, document study, and direct observations. During this case study, a project member from Deliver has worked at the Exact Software site in Delft to conduct interviews with members of the e-Synergy product support and development group.

Exact Software has implemented their release, deployment, and delivery problems through the use of e-Synergy. The Deliver group is therefore interested in the parallels between e-Synergy and the Intelligent Software Knowledge Base as described in Section 2.3.

Exact Software is an appealing subject for a case study since e-Synergy implements features of an ISKB as proposed by the Deliver project. It is interesting to the Deliver group what problem areas, as outlined in the Deliver project proposal [1], are addressed at Exact Software through the use of the Exact Software e-Synergy product. Some of these areas the Deliver group expects to explore at Exact Software are:

- Web delivery of products and updates
- Consistency checking of upgrades
- Workflow support
- Configuration management

This case study report is of value to Exact Software and to Deliver since it will be used for future publications, research, and development. The results of this case study are presentations for industrial groups and academic groups. The interest by the Deliver group was raised by the simplicity with which Exact Software claims to have solved its release and deployment problems.

1.5 Main Contributions

This report describes the observations done by Deliver during a case study at Exact Software. The case study report describes the processes of release and deployment. Some of the development processes of Exact Software are described, when related to the processes of release and deployment. Finally, the tools supporting the processes of release and deployment are described and compared against the features of the Intelligent Software Knowledge Base.

The case study report lists improvements coming from the comparison of the Intelligent Software Knowledge Base features and Deliver processes, to the Exact Software tools and processes. The final conclusion of the case study report is that e-Synergy is not an instantiation of the Intelligent Software Knowledge Base because the knowledge stored by e-Synergy does not resemble the amount or form of information stored in the Intelligent Software Knowledge Base. However, many of the characteristics found in e-Synergy can also be found in the Deliver picture of an ISKB.

1.6 Report Overview

The rest of this case study report is built up as follows. The second chapter deals with the research methods, questions and approach of the case study. The third chapter describes the observations and process descriptions of Exact Software. The fourth chapter describes the analysis of all observations. The fifth chapter describes the suggested improvements for the Exact Software development, release and deployment processes. The final chapter describes the conclusions and contributions of this case study.

2 Research Approach

This chapter describes the research approach for the case study at Exact Software. The goal of this case study was to compare the concepts of the ISKB to the day-to-day use of e-Synergy. This chapter describes the concepts of the ISKB and its structure, the research questions and finally the research methods.

2.1 Definitions and Conceptual Model

The main focus of this case study is comparing the concepts of the Intelligent Software Knowledge Base (ISKB) and the implementation of some of these concepts to the e-Synergy product by Exact Software. To make this comparison, the concepts associated with the ISKB and the research must first be defined. The release and deployment processes have been studied within Exact Software. This chapter gives definitions of these three processes.

- Release**
 Hoek et al. [4] defines software release as "to package and make a software system available to a customer." Looking at this definition, the process of releasing is not only the finishing step of development, but also packaging customer specific installations.
- Deployment**
 According to Hall et al. [5], the deployment process contains the "delivery, assembly, and maintenance of a particular software system at a site." The delivery process description consists of the delivery of software to the customer, and the delivery methods, communication mechanisms to the vendor of the software and the instantiation of the deployment process.

The processes described above have been used to map and direct the research because these processes are the prime focus of Deliver. Since all these processes are part of the software life cycle, the life cycle has been studied as well.

2.2 Delivery Model

The Deliver team has a vision of what Software release and deployment should look like. Figure 3 shows the general architecture the Deliver group proposes for software delivery. The software vendor develops components in all shapes and sizes, which in turn will be deployed at different customer sites in different configurations. Two main characteristics of this architecture improve the delivery process:

- The use of the **Intelligent Software Knowledge Base (ISKB)** that contains exhaustive information about all software artefacts and their constraints.
- A **Web Based Delivery Process** to deploy, upgrade, and replace software components based on the information in the ISKB.

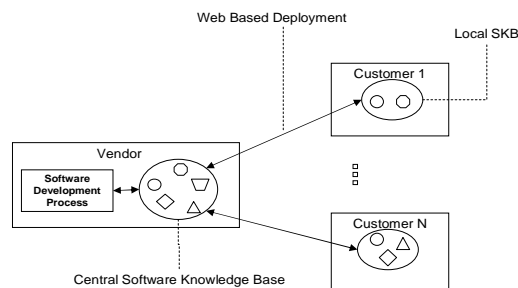


Figure 3 The Deliver view of software delivery

Figure 3 displays the Deliver view of software delivery. In the past, software was delivered to the customer on some media (CD, floppy, etc) and then deployed at the customer site. The software vendor did not have any knowledge about the customers' installed situation. Deliver sees the process of software delivery differently.

Figure 3 shows the customer and vendor sharing knowledge about software artefacts by keeping a vendor and a customer knowledge base. This knowledge base holds information about components, deployment restrictions, configuration details, etc. This knowledge can be used by the customer to deploy software, to ask what-if questions, and to keep up to date with the most recent versions of software.

2.3 Intelligent Software Knowledge Base

The **Intelligent Software Knowledge Base [1]** (ISKB) is a concept introduced in the Deliver proposal [1]. It stores exhaustive information on software artefacts and assists the software vendor in many ways during development and deployment. The focus of the Deliver group, however, lies in the areas of release and deployment. The ISKB implements features from product data management, software products, and software configuration management. In this chapter some of the features implemented in our ISKB that are relevant to this case study are explained.

2.3.1 Structure

The ISKB supports and improves the processes of release and deployment. Each of these processes poses different challenges for the system. Support for the release process with the ISKB is founded on the idea that the artefacts of every software system are stored in some kind of repository for development. The ISKB assists developers performing operations like product composition and other development tasks. These operations require knowledge about components the relationships among them [12]. This knowledge is stored in a versioned database and is accessible to all development personnel.

The knowledge stored in this development section of the ISKB can be published once a release is performed. This knowledge is publicly available for the ISKB to assist the process of delivery and deployment. The delivery process contains processes such as media creation, web delivery, “what-if” query handling, and all other tasks to do with the task of getting the software to the customer. The ISKB runs a service in a specific location for customers to connect to and query.

Finally, to assist and support the deployment of software at a customer site, a customer side knowledge base holds all the information on deployed software. This part of the ISKB assists the customer in finding out what the customer needs, and what components are required to get functionality not yet deployed on the customers’ computer.

2.3.2 Usage

The following list summarizes some features and processes that are improved by implementing an ISKB as proposed by Deliver that are also applicable to this case study.

- **Configuration Management System (CMS)**

The Deliver project is not planning to implement yet another CMS; however, Deliver assumes that products are stored in some kind of repository. This repository typically stores different versions of source code that can be extracted at will. Relevant to the Deliver group is whether dependencies are administered between different versions of sources in a CMS. This makes it possible to deliver a component or software package with compatible sources, even if they are from different released software versions. The same holds for inter-component dependencies. The ISKB uses a Feature Description Language that describes dependencies among components.

- **Build Systems**

The knowledge stored in the ISKB can be used to build products as well. When a product is designed, it is useful to use the dependency relationships between source files to assure consistency and completeness. The ISKB is able to feed dependency information to build systems and implements its own build system.

- **Customer Configuration and “What if” questions**

The ISKB implements mechanisms to generate an installation for a customer depending on the components the customer has installed. This means that some kind of querying mechanism is needed to generate a list (or tree) of what components need to be installed at the customer site to provide functionalities a customer has requested, or a list must be generated stating the conflicting components.

- **Customer Installation**

When the software is delivered to the customer site, a tool that deploys, configures and possibly builds the software is needed. Rollback functionality is a feature that is required for configurability. One of the main features Deliver wishes to see in the ISKB is support for web delivery. The ISKB should support all features mentioned here through some kind of web interface.

- **Software update and Installer Media Creation**

The ISKB should enable some kind of mechanism to do updates, and detect conflicting and inconsistent sets of components. There should also be some kind of auto update facility, so the applications based on the ISKB should support push, automatic pull, and pull mechanisms. The applications built around the ISKB should also support some kind of installer creation. This installation package can then be transported, through some media. It should be possible to create these installers automatically.

These features emphasize the main functionalities that have been studied at Exact Software. These features also illustrate how the Intelligent Software Knowledge Base is used for the processes of release and deployment.

2.4 Research Questions

The main objective for this case study was to compare the e-Synergy product from Exact Software to the Intelligent Software Knowledge Base from the Deliver project. From this main objective the Deliver group has derived the following research questions.

1. What do the release and deployment processes look like at Exact Software?
2. How are these process supported by the tools used in Exact Software?
3. How do these processes and techniques relate to the Deliver concepts and techniques?
4. What processes and techniques of Exact Software can be improved or extended?

The first research question represents the Deliver aim to model the release and deployment processes at Exact Software. The answers to this question, formulated in process descriptions, served as a basis to answer the other research questions. With the processes of release and deployment modelled it was possible to explore the processes and compare them to the Deliver viewpoints, to define the problem areas, and to look for improvements in the Exact Software processes.

The second research question is focussed on the tools supporting the processes for release and deployment. These tools needed to be mapped and classified to compare the features of these tools to the features of the ISKB. This question also enabled Deliver to see a different solution to the problem areas of release and deployment.

The third research question shows that Deliver is focussed on the ISKB and its features. Deliver wanted to explore the features of the ISKB and compare these features to other solutions in the same problem field. With this question answered it was possible to evaluate features of the ISKB by (theoretically) applying them to production processes.

The final research question was asked to see whether the features of the ISKB and the concepts of Deliver could be applied to the Exact Software production environment. The impact and importance of features were evaluated and compared to the features of the Exact Software tools and processes. The result of the comparison between the features of the ISKB and the Exact Software tools are the improvements in Section 5.

2.5 Research Methods

During the case study facts have been collected to answer the research questions. The means through which the Deliver group gathered these facts were:

- **Interviews**

The main research questions have been answered in part during the interviews with the people responsible for the development and usage of the e-Synergy product. The interviewees are listed in Appendix 8.2.

- **Studying the software**

Exact Software has granted an academic license to its software. This license has helped the Deliver group to gather many facts, by examining, using, and experimenting with the software.

- **Document study**

Documentation has been found in e-Synergy. Studied literature can be found in chapter 7.

- **Direct observations**

During the research direct observations have been made and documented to answer the research questions. These direct observations were made mainly throughout the stay of the deliver member within the International Development department of Exact Software.

- **Participant observations**

During the case study all work done by the Deliver team has been recorded into the e-Synergy system. Also, all meeting requests and documentation downloaded was stored in e-Synergy. This gave the Deliver team sufficient opportunity to get some experience with the e-Synergy way of working.

Because of the fact that interviewing is the primary means for research the results are largely of a qualitative nature. This is appropriate since this case study is of an exploratory nature [3].

3 Exact Software

This chapter describes Exact Software. First the organisational structure is given, explained with the focus on development, quality assurance, and product release. In Section 3.2 the structure of Exact Software products is explained, including a detailed description of how products and product families are built up within Exact Software. This is followed by a description of the tools used by Exact Software, including e-Synergy. Finally, this chapter describes the global and developer, release, and deployment workflows.

3.1 Organisational Structure

Exact Software is organised in a large holding company with smaller divisional companies. Each subsidiary operates independently and is responsible for a specific product or service. Each subsidiary is responsible for its own financial health and existence. This division creates clear boundaries between separate problems and opportunities in the market for ERP software. Examples of these subsidiaries are International Development, Exact Software North America, and Exact Software Retail. Development subsidiaries are generally focussed on one technology or operating system. The research for this case study happened mainly within the International Development department of Exact Software.

3.1.1 Development

Exact Software had an average of 2025 employees over the year 2003. Of these 2025 employees, approximately 400 people were developing software. The largest part of these 400 employees is in International Development with 180 employees in Kuala Lumpur working on the Globe application, and 20 employees in Delft, working on e-Synergy. The remaining employees were divided among different departments such as Exact Software Custom Solutions, Exact Software Grote Beer, or Exact Software Client Server.

Exact Software develops new software products by first building a prototype within research and development. Once this prototype reaches a certain level of stability it is moved on to a new subsidiary that is from then on responsible for the further development of the product. This subsidiary has no contact with other subsidiaries, except for the previous spearhead team from Research and Design. The modules of the prototype, which were formerly developed by a small development team (usually no larger than 8 programmers), are divided up between teams of programmers who are responsible for their module. In this group of three to six programmers there is one senior programmer who is responsible for the total module. The other programmers are responsible only for the target files assigned to them by the senior programmer. This division in responsibility allows people to see who is responsible for a recent change or fix, and it is also possible to see who knows about implementational details of certain targets. However, also many responsibilities are in the hands of the developer.

3.1.2 Quality Assurance

Quality assurance plays a big part in the software development process of Exact Software. Quality assurance guarantees that a product is working and that each new addition to a project has been completed. To reach this goal, the quality assurance department approves changes, bug fixes, and functionality requests from developers, tests new releases before they are promoted to the C repository (see section 3.7 on Product Release), and maintains standards of quality of developed software. Quality assurance personnel check all developers within Exact Software. Someone from the quality assurance team must first approve every change that is done, every feature that is implemented, and each task that a developer deems finished. If they disapprove the developers' work, they must state a reason and report the disapproval back to the developer. If a quality assurance team member approves a change, bug fix, or functionality request the member later on is held responsible for bugs and erroneous functionality. The second task of the quality assurance personnel is to check whether a release can be promoted (See Section 3.7) to a higher level. The quality assurance team does so by testing this release thoroughly.

3.2 Product Structure

The structure of all Exact Software products is stored in a product data management system. A product data management system is able to manage attribute and documentary product data, as well as relationships between them, through a relational database system. Information of similar types can be grouped together in named classes. More detailed classification is possible by using 'attributes' to describe the essential characteristics of each component in a given class. The use of a product data management system is consistent with Exact Softwares' claim that the manufacturing of software is no different from any other product. The product data

management system used within Exact Software is, not surprisingly, part of e-Synergy and is called e-Logistics. Products belong to product families. Developer groups are usually active within one product line, called assortments within Exact Software.

3.2.1 Product Description

Deliver sees a product family as a set of software-intensive systems sharing a common, managed set of features. These features are developed from a common set of domain specific features in a prescribed way. In e-Logistics the equivalent of a product family is called an assortment. Product families, such as Globe, contain all items and products belonging to that product family. As an example, Globe comprises Globe for Dos, Globe for Windows, Globe 2000, and Globe 2003.

e-Logistics makes use of atomic entities called items, which are grouped into assortments (product lines). An item can be any business item, such as a promotional umbrella, a printout of a manual, but also some saved version of that manual. Items are split up into groups, of which the most relevant groups for this case study are sales items, source items, and target items.

- **Sales items**

Exact Software uses sales items to encapsulate sellable goods. A sales item can be a service agreement, a manual, a piece of software (including a paper manual and a CD), or any other good sold by Exact Software. This term is derived from product data management. From each sales item a bill of materials can be generated, stating what items are necessary to complete the product. When a sales item is a software product the bill of materials includes target items, but no source items. Sales items are part of the sales view of Exact Software products. Products are large sales items.

- **Target items (targets)**

Target items are deliverables dependent on source items, the most concrete example being executables depending on their source files. However, target items also include digital manuals, resource files, etc. Target files are depending on source items, even if they are exact copies of the source item. Target items are part of the developer view of Exact Software products.

- **Source items (sources)**

Source items are source files that are required to create a target. Source items are source code files, resource files, etc. Other companies use the source items to store their basic materials and resources with which they create their products. Source items are part of the developer view of Exact Software products.

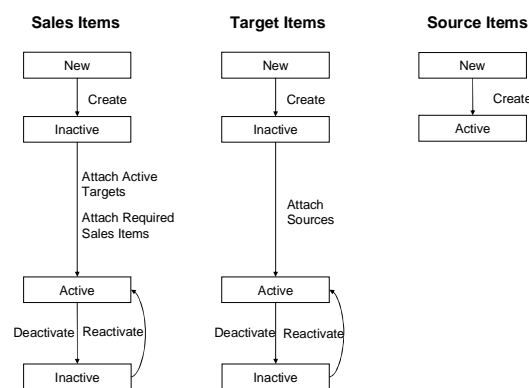


Figure 4 Item Lifecycle for Sales and Target Items

Figure 4 indicates the lifecycle for an item. To create a product, a sales item is first created. That sales item cannot be activated (and therefore released) unless it suffices certain requirements, such as attached sales contracts. Target items cannot be activated unless they have sources attached to them. Once objects have been created they can only be removed by their creator. Source items can only be created and have no status, since their status depends on whether the source item is a source file for an active target.

Items and products are grouped into assortments. All the items that belong to a product are also part of the product family to which this product belongs. These items can be shared though, through a parent-child

relationship with other product families. It does not matter to an item what assortment a product belongs to, even though items too belong to an assortment. An example of this is the BacoSettings.dll that is part of both the Globe 2003 and the e-Synergy product. Product families do not have any relationship with other product families, except for the fact that product families might share items.

3.2.2 Developer View

Development teams operate within one assortment. When a development team is working on a product, they are making changes to the source items and target items of that product. When a product requires a target item from another assortment, the developers simply include that target item as part of their product. That target can still be changed by the development team working within the other assortment, thus allowing the other development team to break the product. An example of a developer view of a product is shown in Figure 5. Product x1 is made up of three targets. Target y3 belongs to a different assortment, yet this external target is still part of product 1. A developer working on Product x1 will consider target y3 immutable, considering that the developer cannot change target y3.

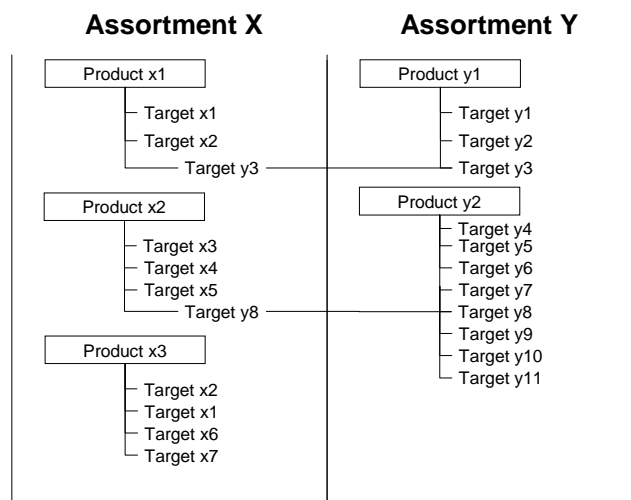


Figure 5 Assortments and Products

An example of a target item “BacoSettings.dll” is displayed with its sources in Figure 6.

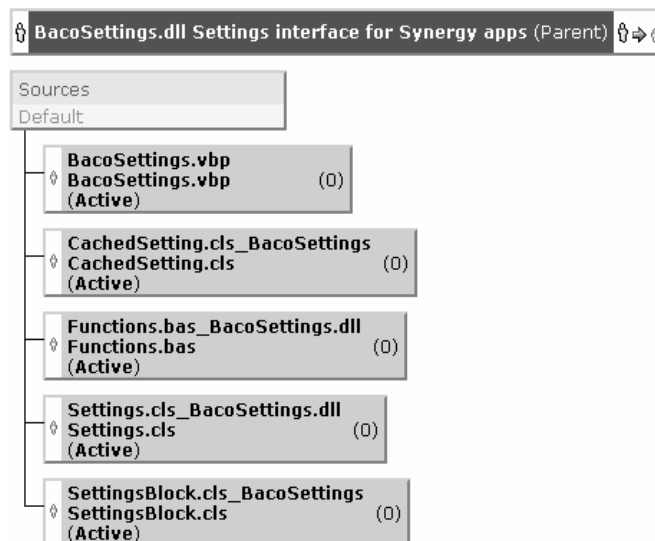


Figure 6 BacoSettings.dll and its sources

This figure displays the target BacoSettings.dll. The target is linked to its sources through a Sources relationship, which means this item depends on these source items for building, without needing them to be deployed (which is a standard rule defined in e-Logistics). The sources relationship is the only relationship that applies to the developer view of a product. Relationships such as “optional” are part of the requirements specification for a product and the sales view of a product. Structural relationships between targets are not defined in e-Logistics.

3.2.3 Source Items and Attachments

Within Exact Software development the term target item is used to describe all files that are necessary to complete a software package. Target files are files that are required to make a complete product set and usually are dll files, asp files, executables, libraries, program resources, and help files. Targets have source files associated with them in the e-Synergy data model. These source files are registered as source items in the system. A source item always has a set number of attachments, one for each release of the product. In the case of the example in Figure 7 there are five 4.0 attachments. These five attachments represent the O, D, C, B, and A repositories.

Release Owner	File name Version	Size Compressed	Uploaded	Checked out	Children
304	Functions.bas	8,632 1,987	Sebas 05-03-2002 20:50		Q
					Show Download Copy
304a	Functions.bas	8,632 1,987	Sebas 05-03-2002 20:50		Q
					Show Download Copy
4.00	Functions.bas	8,632 1,987	Bart 19-11-2003 14:15		Q
					Show Download Copy
4.0A	Functions.bas	8,632 1,987	Bart 19-11-2003 14:15		Q
					Show Download Copy
4.0B	Functions.bas	8,632 1,987	Bart 19-11-2003 14:15		Q
					Show Download Copy
4.0C	Functions.bas	8,632 1,987	Bart 19-11-2003 14:15		Q
					Show Download Copy
4.0D	Functions.bas	8,632 1,987	Bart 19-11-2003 14:15		Q

Figure 7 Sources, attachments and the developer

In Figure 7 the attachments to the item “Functions.bas” are shown. The developer, Bart, is the responsible owner for this particular file at present. A developer ‘Sebas’ previously owned this file, since he has uploaded files in the past. Each of these files represents a file in a different repository, as will be explained in Section 3.7. In Figure 7 the development release of the Function.bas, edited on the 19th of November, has been promoted (see Section 3.7) all the way up to the NULL repository. If the user Bart had made this screenshot, an upload, commit, and lock button would have been visible under the file near the 4.0d repository.

3.2.4 Products and Items

As explained before, items are part of assortments. Items can be tangible parts, item containers that describe a full product, software items, etc. These items can then be linked together through item relations. Some of the available item relationships in the two views are:

- Mandatory;
- More-of;
- One-of;
- Optional;

- Targets (meaning the file is required for the parent item)
- Sources (meaning the file is a source file for the parent target);

The relationships defined here are similar to the relationships defined for feature diagrams [12]. The relationships in the product data management tool of Exact Software are only used to define contracts and license files for customers and not to define the software structure, since all components and files required by a product are defined as “targets” of that product (targets have no relationships among them).

A product is composed of sales item containers and a target item container representing the two different views on a product. The sales item containers contain sales contracts, optional support contracts, license agreements, legal statements, etc. The target item container contains all target files that make up a product. There are no relationships defined between these targets, the container simply consists of a pointer list to each target. In Appendix 8.1 the item counts can be found for the e-Synergy product and the Globe 2003 product.

The sales item container contains the sales items that represent the product in a commercial context. The following subsections explain the sales view of Exact Software products.

3.2.5 The Sales View

From a sales point of view, it is not relevant what targets and sources look like. For the sales view it is required to know what options there are to a product, what kind of sales agreements are possible, and what physical media and material makes up a product. The sales view differs from the developer view with respect to variation points mostly. Developers see the product as a complete set of targets making up a product, in which options can be activated and deactivated at runtime. The sales view is oriented on the fact that these variabilities need to be bound during the signing of a contract. Another difference between the sales view and the developer view is that the developer view is focussed on source files and compiled targets, whereas the sales view is focussed on tangible parts making up the product with their attached prices.

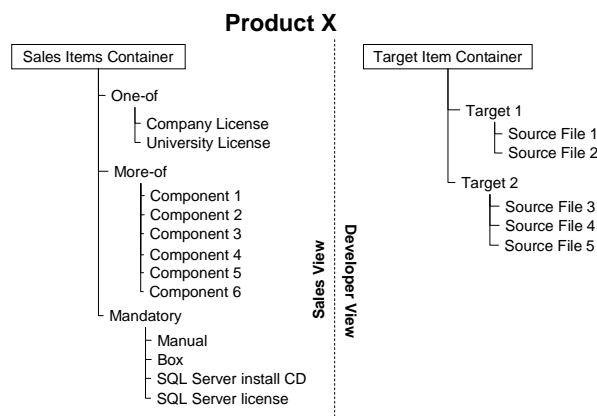


Figure 8 The Sales and Developer View of a Product

Figure 8 shows the two different views of a product within Exact Software. The figure displays that the two views are different in the fact that the developer view is concerned with targets and sources and the sales view is concerned with sales items, variation points, and tangible items that are part of a product.

When designing a product first an assortment must be chosen. A product can then be created in the designated assortment. The product currently is empty and requires target items with their sources and sales items to be completed. Firstly the creator of the product now adds targets to the assortment. The creator then takes the sources and attaches them to newly created targets. Once the targets are defined, sales items can be created. A sales item can be a manual belonging to the product defined with a mandatory relationship, a license package for multiple users defined with a one-of relationship, or a list of components with a more-of relationship. Once the product definition is completed it can be sold. To sell a product it must be instantiated. To instantiate a product a contract must be filled in specifying what product and binding the variations by making the relationships concrete, for instance by choosing the components that must be shipped.

A product is bound at two points in the release phase. When a release is promoted (see Section 3.7), immediately a new set of targets becomes available for that release. However, when a customer signs a contract, a license file needs to be generated to activate the components that the customer has purchased. The creation of a license file implies the binding of all relationships for a product defined as one-of, optional, and more-of. The person entering the contract into the database binds these relationships manually. It is also possible to create another product, in which these variabilities are already (pre)bound. Variation is thus supported by the e-Logistics system at product design time and at product instantiation time.

Once a contract with a customer is filled out it is entered into the database of e-Logistics. Exact Software has a script running on the contracts database looking for new contracts. If a new contract is found, an Exact License File is created for that customer, specifying all options. The Exact License File is then attached to the customers' record. The customer, who can see her own record, can now download the Exact License File.

3.2.6 Product Life Cycle

Exact applies a straight forward process, displayed in Figure 9, to develop a product. First a requirements document is created and approved by upper management. Once approved an inactive product is created in e-Logistics. An inactive product cannot be sold, but can be worked on. A spearhead team of software implementers and designers starts working on the product. Once the spearhead team delivers a prototype the product is handed over to a team of developers. These developers activate the product once it is ready to be sold.

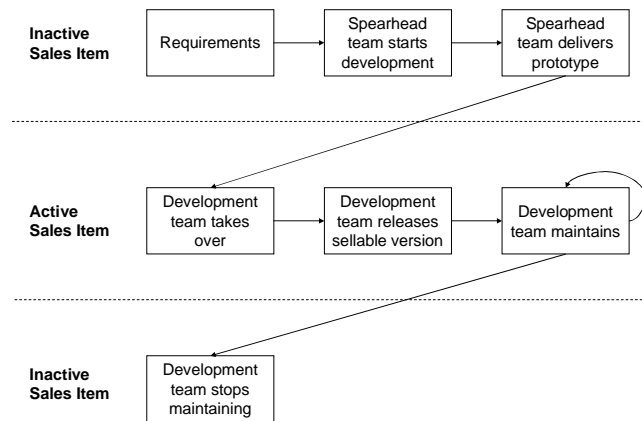


Figure 9 Exact Software Product Lifecycle

The development team will keep on releasing new releases, according to the promotion scheme explained in Section 3.7, until the product is no longer sold and supported. The development team, doing mainly software maintenance work until then, deactivates the product. The targets of that product can still be used in other products.

3.3 Tools

Three tools support development and deployment: **e-Synergy**, the **Exact Software Product Updater** and the **Source Manager**. The Exact Software Product Updater can download a full version of a product or update a current installation. The Source Manager tool is used to download specific targets, specific sources, or all sources and targets. e-Synergy is the tool that is used by all employees of Exact Software to handle all their day to day business. e-Synergy is explained further in

3.3.1 Source Manager

The Source Manager tool is used to download specific targets, specific sources, or all sources and targets. The tool can upload multiple files (if the developer is responsible for each target he/she uploads), download the sources that belong to different repositories (to test a hotfix, for instance), lock files, and upload newly created targets. The source manager was created for developers to enable the upload of larger batches of files. The e-Logistics module offers the same functionality but is not as easy to use for larger numbers of files. Target files have internal version numbers that are increased upon each upload. These internal version numbers are later used to compare an installed set to an available release and upgrade to the latest version of the software.

3.3.2 Product Updater

The Product Updater provides the mechanisms for delivering packages and updates to the customer. The Product Updater also deploys the packages and updates at the customer. When run at the customer site, the Product Updater needs to be provided with an installation location (CD-ROM or the Web), a license file, an optional local installation, and the users' username and password. The Product Updater will then download a list of files and their internal version numbers from the installation location. The internal version numbers represent commits during the lifetime of the files and are unrelated to batch numbers. This list is then compared to the local installation list, which is empty in the case of a new installation. Once decided which files are newer these files are downloaded to a temporary location. Downloaded files are ASP files, binary files, help files, graphics, SQL scripts, etc. The installation files are downloaded through a secure HTTP [10] connection. These updates are not checked for integrity once they are downloaded. It is also possible to get the update with a Verisign [9] signature. Since the Product Updater needs to be started manually, the Product Updater only serves as a pull

mechanism. Once the download is finished, the Product Updater sends the batch number of this upload to the Exact Software e-Synergy CRM module in Delft, so that Exact Software employees can check what version was downloaded last by the customer. This batch number does not include any information about the success of the update. The Product Updater is configurable such that a language can be selected, the name for the updater can be changed, the icons, and of course the software location. This makes it possible to create a Product Updater for each software package that is produced within Exact Software.

The Product Updater provides the deployment mechanisms for all Exact Software packages. The Product Updater performs a number of custom checks before downloading, for instance in the case of the e-Synergy server application, the Product Updater first checks whether MS SQL server is installed, IIS web services are available, and whether there is enough disk space. Once all the files have been downloaded to a temporary directory, the available SQL scripts are run on the database. SQL scripts are named as follows [from batch number]to[to batch number].sql, as an example 302to305.sql. These SQL scripts can alter the tables by performing data conversion, table addition or removal, table alteration, and data deletion and addition. Once all these SQL scripts have been executed successfully, the files copied in the temporary directory are copied to the installation directory, overwriting when necessary. The batch number, which is mainly the version number for the database, is set to the batch number of the release that is installed after the SQL scripts and the copying were successful. The Product Updater does not give any other feedback to Exact Software besides the latest downloaded batch number from the customer. It is impossible to perform updates at runtime with the product updater. Exact Software claims that their products do not need runtime updating because their offered services only need to be available at working hours. Rollbacks are also impossible because the updates are destructive and not incremental.

3.4 e-Synergy

As mentioned before e-Synergy is used as a tool by developers, but also to support release and deployment. Since e-Synergy is web based and supplies different portals it is also used to spread information to the associates of Exact Software. e-Synergy is used for development for three main functions, being; a repository for the releases, a workflow tool, and a knowledge sharing tool. The uses of e-Synergy as a support tool for the processes of release and deployment are summed up here:

- **Development**
 - e-Synergy can, through e-Documents, supply developers with informational pages to **knowledge about development**. Extensive examples are the SDK development pages and the development team pages. The same system is used between developers to share documents, work on concepts together, and to review designs.
 - e-Synergy is also used by development to manage their **workflow**. Workflow usually comes in the shape of tasks, bug reports, and functionality requests. Each processed task, bug report, or functionality request is reviewed by quality assurance personnel and this process is also managed through the workflow module. Larger projects are done through e-Project, generating more workflow.
 - e-Synergy's most influential function to development is the e-Logistics module, serving as a **repository** for all development personnel. The e-Logistics module is used to support the promotional scheme for releases and to commit, download, upload, and lock targets.
 - e-Synergy is used to generate customer and dealer **feedback**. Exact Software employees can then produce bug reports and functionality requests using this feedback.
- **Release**
 - Besides development, other processes, such as **product data management** are also supported by e-Synergy. The e-Logistics module, as described in Section 3.2, supports product composition and product family and product design.
 - e-Synergy is used to **inform** customers about new releases.
- **Delivery**
 - e-Synergy is used to deliver the **Exact Software License File** to the customer.
 - e-Synergy is used to store the latest releases so that the **Product Updater** can connect to the update server and download the most recent version of software.
- **Deployment**
 - The Exact Software License File, generated and delivered through e-Synergy, specifies what modules are **activated** for a product.

Table 3 e-Synergy Lines Of Code count

	LOC	Comments
ASP	325,917	16,209
VB	263,193	42,170
C++	73,021	6,263
Total	662,131	64,642

This table shows the lines of code count for the e-Synergy product. The product mainly consists of ASP and visual basic code. There are also some C++ modules that handle the access routines to the database.

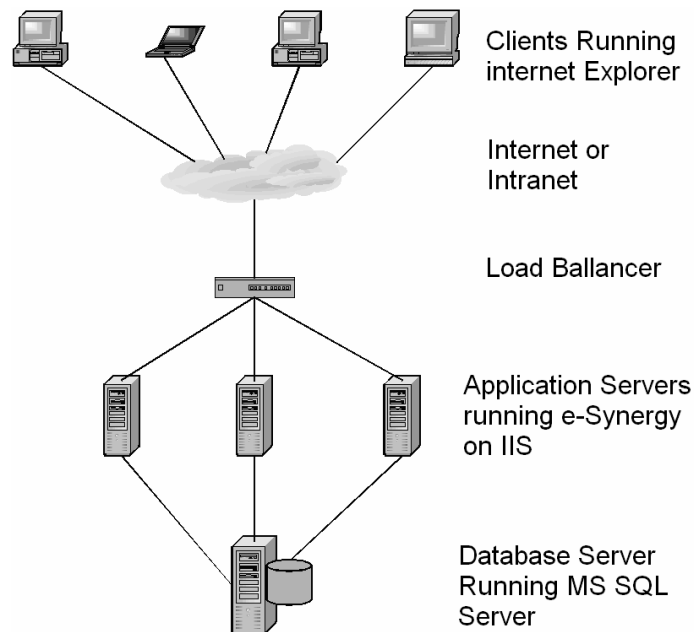


Figure 10 The e-Synergy infrastructure

e-Synergy is a three-tier web application running on MS SQL server together with IIS and some Exact Software specific services (See Figure 10). It provides a front office application with all the modules mentioned above.

3.4.1 Software Architecture

This section describes the software implementation and architecture for the e-Synergy product. The e-Synergy product works on a centralised database, with different clients accessing it through a web browser (through a wide area network). The e-Synergy server requires a running web server and a MS SQL server database. Upon the database a database layer has been built, implementing the database accessing routines. These routines are called from ASP files (views), generating HTML files on the server side. The ASP files need to be opened first in Internet Explorer. In Figure 11 a simplified version of the software implementation of e-Synergy can be found. Each layer in the architecture is supported by a technology used by Exact Software.

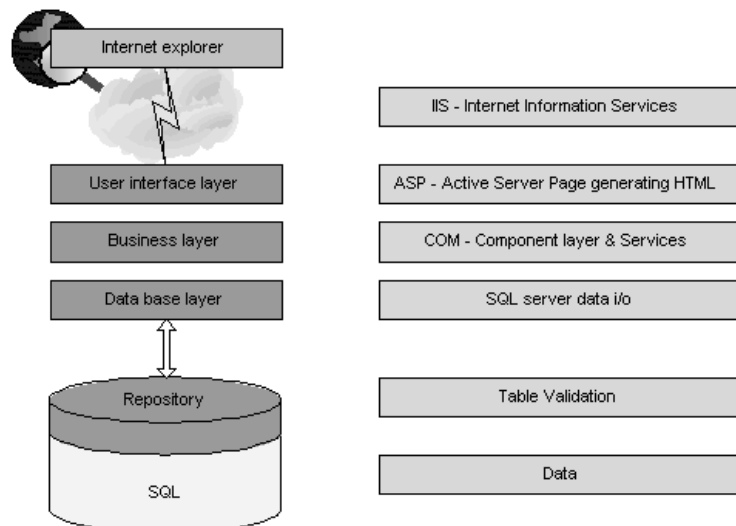


Figure 11 The e-Synergy architecture

The e-Synergy architecture, as seen in Figure 11, shows what layers in the software architecture are handled by what technology. The tables are stored in the MS SQL database, the COM component model [18] is used to achieve business integration, and a user interface is provided through ASP [17] web pages.

3.4.2 The One-X Data Model

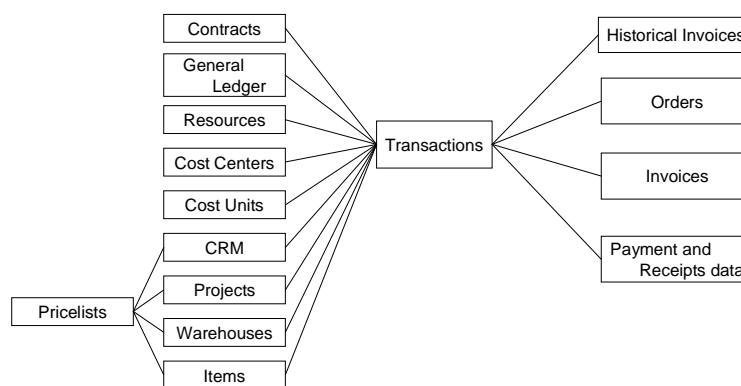


Figure 12 Simplified One-X Data Model

The One-X data model [7] is displayed in Figure 12. The One-X means that every piece of information needs to be entered into the database once and from then on is available to every other part of the system. The database architecture for all Exact Software products, related to business processes, is largely based on Figure 12. The intuitive notion that all information about different business processes can be found through one interface makes it a powerful model for information retrieval. Since all the information is stored into one database all information can be accessed by one application. As an example, when looking at a document, it is possible to go to the creators' human resource information to see whether the creator is reachable today. It is also possible to see what that person is working on at present and when he/she went into e-Synergy last. With the right permissions it would even be possible to see the salary of this person.

Since the database model is similar for all Exact Software products, the database can be accessed by different products as well. An example of this is the Exact Software database, which can be accessed by the Globe Application, the e-Synergy application, and the Exact Software Web server. An example scenario: a document

can be published by someone in e-Synergy, which can be viewed on the website of Exact Software, while someone in Globe is registering this document as a sales item for a product.

All the software Exact Software builds can be access this uniform data model and perform the tasks the application was built for, making the software flexible. Since all the information is stored into one database, different applications can access the same shared information, giving customers the opportunities to improve integration of front-, head-, and back office activities opening the ‘medium to large sized company’ market. Since each application built around the One-X architecture can make use of the currency and language information in the database, applications are easily made internationally available thereby also opening more markets for Exact Software. The One-X architecture is also built for expandability, so that Exact Software and its customers can create custom solutions to extend the functionalities of products.

3.5 Global Workflow

Exact Software generates income through the sales of their software and through the services and support Exact Software delivers to customers. The Research and Design department develops this software first, forming a spearhead team that is specialised in the new product, its development techniques, and the prototype they are developing. Once a product has reached the state of a mature prototype, the spearhead team writes documentation for the prototype and the prototype moves on to a team of the development department. The development department together with the quality assurance personnel is responsible for turning the prototype into a sellable product.

Once a product reaches the state of a sellable item, it is released by entering the product as a legal sales item. This sales item is then part of a larger product, or is an independent product. Once released, it is available for pilot customers who can put the product to the test in production environments. If Exact Software can use the product internally, they typically first test it within their own production environment.

After release, the product has to be transported to the customers. The product will from then on be delivered through Exact Softwares usual means of delivery, either by CD or by the Product Updater.

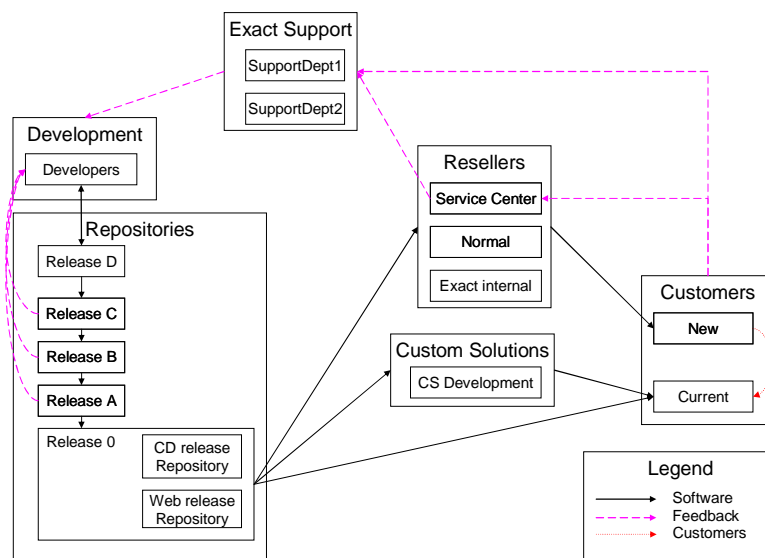


Figure 13 Global Workflow

The delivery process within Exact Software is also depicted in Figure 13. This figure also displays the stream of feedback to the development process and part of the release process. The rest of this subsection is split up as follows; first the development workflow is described, then the release workflow and the repositories. Finally, the deployment workflow describes the delivery and deployment of Exact Software products.

3.6 Development Workflow

Development of software within Exact Software is largely based on two main facts, being that Exact Software operates in a well-defined problem area, and the fact that the first design by the spearhead Research and Design team is a stable prototype. Once the Research and Design team delivers a prototype, the responsibility for the

prototype is taken over by members of the development team. Each member of the development team becomes responsible for a part of the system. Since it is assumed that the design of the prototype is good and no large architectural changes will be made, the responsibilities can easily be divided. The product is defined as a number of target files, files that will be delivered to the customer, for which designated development team members are responsible.

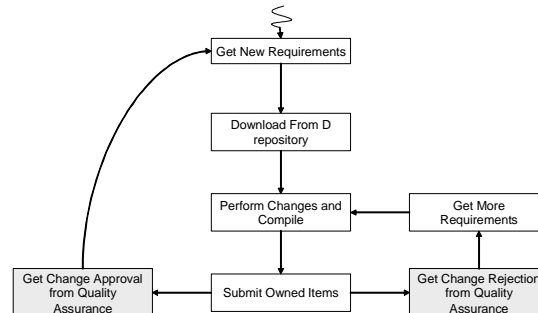


Figure 14 The maintenance and development process

Figure 14 displays the workflow of developers as they develop and maintain an Exact Software product. The developers are mostly driven by their own workflow. These are incoming tasks and requests that can be bug reports, functionality requests, or any other form specified within the e-Synergy system. These tasks usually require the developer to change some of the target files he/she is responsible for. When a request or bug report has been processed, the workflow is passed on to a quality assurance representative who can then approve or reject the change or bug report. This workflow provides clear traceability because it is possible to see what kind of workflow has recently been performed on a target. This does not ensure that the last upload is specifically linked to a task, so a developer can still upload his/her files without having to specify why this upload is taking place.

A product item manager is responsible for the complete software product. The product item manager is usually an experienced manager with extensive experience with the product since this person is held personally responsible for the quality of the complete product. The product item manager within Exact Software is usually also the manager for the development team of that product. Product items are usually subdivided into smaller items, called modules. The responsibility for these modules, for instance one of the seven modules in e-Synergy, lies in the hands of senior developers who manage 3 to 6 developers. These developers are then each responsible for a set of targets. All these sets make up all the targets for that module.

Every target has a target owner: the software engineer responsible for the target. The software engineer realizes the targets for the product. When finished, the targets, including the sources, are uploaded in e-Synergy. This is all part of the source management system. Formally, the release/patch manager is responsible for releasing the targets. However, since this is usually the manager of a development team, this responsibility is propagated to the developer and the quality assurance team member. [8]

The targets are stored in the D repository (see Section 3.7). Because Exact Software wishes to constantly have an up-to-date compiled version in the e-Synergy repository, the developer locally compiles the sources. The developer then uploads both the source code and the targets. The consequences of this developer side compilation are that there is no need for a build server, and hypothetically, the repository always contains the latest compiled version. The downside is that developers are themselves responsible for correct compilation, with the right target dependencies and compiler flags. Another downside is when a large number of targets are depending on each other; these targets have to be uploaded at the same time to keep a correct set. Uploading around the same time solves this inconsistency.

3.7 Release Workflow

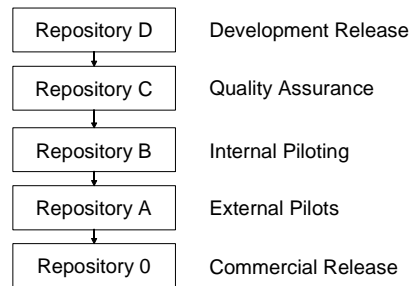


Figure 15 Exact Software Promotion Scheme

Exact Software uses a promotion scheme [20] to maintain the quality of their developed software. There are five stages, realised by repositories. The repositories and promotion scheme are modelled in Figure 15. The first repository is the development release on which developers do all the changes. From then on, periodically, repositories are copied manually to new release repositories, to finally reach the repository available to customers. Each repository contains all the source files, help files, binary files, executables, resources, and SQL scripts, for one product (such as e-Synergy or Globe). Each time a copy is made from the D repository to the C repository, the batch number attached to the release contained in the D repository, currently at 327 for e-Synergy, is increased. At all times, depending on the security level of the person downloading, there are five repositories from which releases can be downloaded. Periodically, depending on certain criteria for each repository, the full repository is manually promoted (copied) upward from one repository to another.

Table 4 Promotion Criteria

Repository	Available to:	Promotion Criteria	Approval Required From:	Promotion Period
D	Development, QA	None		1 week
C		All changes Approved by QA	QA Management	2 weeks
B	Exact Internal Use	Product Approved by QA	QA Mgmt, Product Manager	8 weeks
A	Pilot customers	Product Approved by all NULL release does not promote	QA Mgmt, Product Manager, Upper Management	3+ weeks after B NULL release does not promote
NULL	Customers		NULL release does not promote	

The D repository is the development repository. On the development repository all changes caused by bug fixes and functionality requests are performed. A developer can upload his/her files to the development repository at all times. A developer can use the updater to update to the latest version stored in the D repository to download all the most recent changes and additions from his/her developing colleagues. Periodically a repository (this used to be weekly) is copied from the D to the C repository. This happens when all uploaded bug fixes and programmers have tested new functionalities. It is the responsibility of the programmer to upload newer versions some time before the next promotion from repository D to repository C.

Once every two weeks, the current C repository, after being approved by quality assurance personnel, is copied to the B repository. The release stored in the B repository is used internally by all Exact Software personnel and is therefore tested as a whole. This generates bug reports and functionality requests again. Exact Software claims that by the time a release is promoted to the A repository 80% of bugs have been filtered out and the customers report only 20% of the bugs. Furthermore, of this 20%, customers report 16% using the release from the A repository and 4% is reported by customers using the release stored in the NULL repository.

When the release stored in the B repository is deemed stable enough (decided by primary business users such as the director of Exact Software Finance and Administration) it is copied to the A repository. The release stored in the A repository is then supplied to pilot customers who report their experiences back to their pilot manager within Exact Software. A pilot manager is usually a member of a support department and knows to reroute the support questions to development instead of standard support personnel. This again supplies the development team with feedback. The release stored in the A repository is then given to piloting customers who are willing to

test the software before that release is promoted to the NULL repository. These customers profit from this because they get to use the software before others do, they can have some limited influence on functionality, and they do not have to pay the full service costs. The new NULL release becomes available after the release stored in the A repository has been used for three weeks or more. From then onward the new NULL release is burned onto CD's and the new release stored in the NULL repository is made available on-line, by overwriting the old release stored in the NULL repository.

On the occasions when a high impact bug is discovered that has been moved through to the versions stored in the B, A, or NULL repositories, a hotfix procedure is started. This hotfix procedure implies that a fix is created for each repository containing the bug that is not due to be overwritten. An example: a bug is discovered in the release stored in the B repository that will be copied to the A repository tomorrow. A hotfix procedure is started and fixes are made for the releases in the D, C, and B repositories. The release stored in the NULL repository is unaffected, even if the bug is also present in there, since it was not reported before it was found in the release from the B repository.

A fundamental characteristic is that releases are intentionally ‘lost’ because some releases propagate quicker than others. As an example, if a release stored in the C repository is planned to promote in two weeks, and the newer release stored in the D repository promotes a week earlier it is copied over the release stored in the C repository. In practice the promotion scheme is fully applied to Exact Softwares’ two largest products, Globe and e-Synergy. Other products and product builder departments, custom solutions for instance, use a promotion scheme with only two repositories since their applications cannot be piloted internally or externally, and only management can test and approve the custom solution.

Customers are notified of new NULL releases through the Exact Software e-Synergy customer portal. Exact Software customers can go to the customer portal to be informed about the status of their contracts, their service calls, the product updates, and new products available from Exact Software. If a new release of a large product is available to customers they might also hear about this through commercial channels, such as television commercials. If a customer runs the Product Updater regularly, the customer will always have the newest version of a product and also its added functionality. This additional functionality in some cases must first be activated through the license file for the customer to actually use this new functionality.

3.8 Deployment Workflow

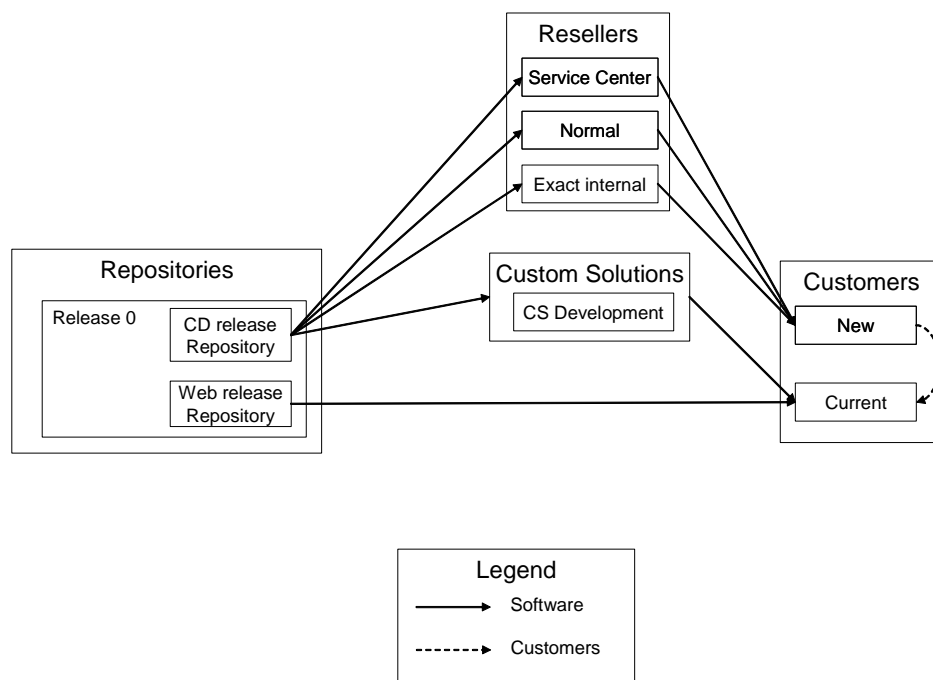


Figure 16 Deployment Workflow

Once the software is available to the customers, the sales items that are available to a customer can be obtained in two ways. The first one is through the Exact Software Product Updater, which can download an update or even a full set of installation files if there is no installation present yet. The second way to obtain the software is by ordering a CD from which the deliverable files can be installed. When a release has been promoted to the NULL-Repository, a copy of the software can be obtained from the CD or the Web repository.

The delivery process within Exact Software is also depicted in Figure 13. This figure also displays the stream of feedback to the development process and a small part of the release process.

Exact Software customers buy products through Exact Software directly but more often through a reseller. When a customer has purchased a product she will get support through Exact Software. When a customer encounters a bug, the support department reports it to the development team. Some of the resellers have reached the status of “Service Centre” which means that they themselves give support. These resellers then handle most of the questions coming in, and report bugs and functionality requests to the Exact Software support departments themselves. When selling a product to a new customer, the software is delivered through a reseller, a service centre, or the internal Exact Software Sales department. Exact Software releases new versions of e-Synergy and Globe every three months. A bug therefore has a lifetime of approximately 4½ months. The software is usually sent on a CD to the customer. It is possible, however, to use the Product Updater instead for online software delivery.

e-Synergy uses item relationships to define variants of a product. These inter-product dependencies (item relationships) are specified or updated during the development phases. Item relationships are used to generate a valid ‘sales release’ of the product and these relationships ensure, that a delivered product includes all the necessary modules. A new release is advertised only through the e-Synergy customer portal since customers use this Exact Software customer portal regularly.

When a product is released to customers, it can be downloaded through the Product Updater or ordered on a CD. For both ways of deployment, the Product Updater is used to complement the installed set, with the newly released set. The product can only be used once the Exact Software license file (ELC) has been downloaded, which states that the product is accessible to the customer and to what extend. To complete the process of deployment, an ELC must be composed and delivered to the customer by Exact Software. This ELC is generated once an Exact Software Invoice is approved.

Once the product is deployed at the customer site, it is possible to customize the product. The customer can set the language and the currency. The customer can also (re)configure the views and add or remove data which is available from the database, such as the creation date of records. The customer can also, through the supplied SDK, build personal extensions to all Exact Software products. Exact Software also strives to enable communication to other applications through BizTalk. [16]

3.8.1 Custom Solutions

When the e-Synergy product does not suffice for a business process, it is possible to get extensions and custom solutions from the Custom Solutions department from Exact Software. Custom Solutions has existed almost as long as Exact Software itself, because there are many different business processes which are impossible to unify in some general process description. Custom Solutions creates new custom solutions for companies with specific needs.

An Exact Software consultant usually installs these custom solutions. It is possible to install or update a custom solution through the Product Updater. Since the user logs in with their personal account for an update, the Product Updater can check whether this customer has custom solutions. In that case, some files are not overwritten and the custom solution repository is queried for updates. Usually custom solutions are not updated regularly because only one customer uses this custom solution.

Custom Solutions over the years also has created a series of standard building blocks, for instance for rental companies. Extensions have been implemented into the e-Synergy product through a simple event messaging system. This event system, which is an integral part of the e-Synergy product, ignores certain messages if there are no custom solutions available responding to these messages.

The implementation of a custom solution has some effect on release and deployment. To start off, custom solutions are (planned to be) unaffected by updates because of the messaging mechanism. Custom solutions are usually delivered on CD and deployed by an Exact Software consultant. If the customers’ deployment needs to

be reinstalled, the customer first needs to install the custom solution CD, and then run the Product Updater to get the most recent version. Updates on custom solutions are only done on request, in which case a new installation CD is created by the Custom Solutions department and the new version of the custom solution is stored in the custom solution database.

3.8.2 Exact Software Invoice

When a sales department from Exact Software receives a signed contract, they upload the contract into their CRM module. When this contract is uploaded, the right options for the products must be selected. These selected products must first comprise a complete set before an ELC can be generated. When the contract creator is trying to save a contract with an incomplete product, an error is generated, by applying the item relationships, stating what options still need to be included or chosen.

A contract is deemed valid once all options and variations must have been chosen (one-of, more-of, etc). When a signed contract is valid and saved, an ELC (Exact Software License File) is generated automatically, through a script that periodically runs. This ELC is from then on available to the customer on the Exact Software Customer portal. In theory, it is even possible for the customer to download the product and install it through the Exact Software Product Updater immediately. In practice however, the customer generally waits for the CD to come in or for an Exact Software consultant to come by before they install.

3.8.3 Exact Software License File and License Checking

Currently, all targets from a product are installed at the customer side, even if the customer has bought only a subset of that product. For example, if a customer buys only two of the seven e-Synergy modules, all the modules are installed. When the e-Synergy product is started, the Exact Software License File (ELC) is checked to see what modules must be activated. This coded license file can be changed only by Exact Software and is overwritten, but not checked, each time an update is performed.

Exact Software does not see software piracy as a threat. Their mechanisms to prevent software piracy are through software aging [11] and through the coded license file. Also, Exact Software claims that it is not worth it to pirate any software from Exact Software, because the products they deliver require strong ties with Exact Software, e.g. consultancy.

3.8.4 Internationalisation

e-Synergy has been specifically built for an international customer base. The result of this requirement is an internationalisation and language mechanism. The data models for Globe and e-Synergy both include terms. A term is a syntactic unit that is used to write user feedback in ASP pages and Visual Basic applications. Once the application is run, each term is replaced by the word that term represents in the language the user has chosen. This mechanism enables for the developer to develop the code in a different language than the language shown to the customer. This terms structure is quite simple and sometimes causes trouble because of the length of translations, which can be longer than the prescribed length for the layout. Also, the context in which a term is used in language A, can be completely different from the context of language B, creating inconsistencies.

Another powerful feature of the system is the support for different currencies. Different currencies can be saved into the system, making it possible for a user to switch all the monetary values to a different custom currency. This enables the creation of financial reports in different currencies with much ease, creating a feature that enables Exact Software to operate on a global market.

3.8.5 Customisations & Extensions

After installation of the e-Synergy product it must be customised to fit the customers need. The e-Synergy product can be used as an *out-of-the-box* application, but it is designed not to. When the e-Synergy product is started the first time, a default language and currency must be chosen.

Usually companies like to create new types of workflow, types of documents, types of assets, and types of resources. The e-Synergy product can be configured to show different layouts for department news pages and front pages. Finally, all users of the system must be entered with their personal security levels, human resource management links, etc.

It is possible through the SDK (studio developer kit) developed specifically for Globe to build customer specific extensions and functionality. This SDK, as seen in Figure 17, is also available to customers so they themselves

can develop e-Synergy customisations and extensions. Integration with other business tools is also possible through the BizTalk XML standard, developed by Microsoft [16]. The SDK has been built in such a way that the SDK dispatcher can direct signals to the Globe application layer. This application layer has been built up in such a way that it can handle custom signals.

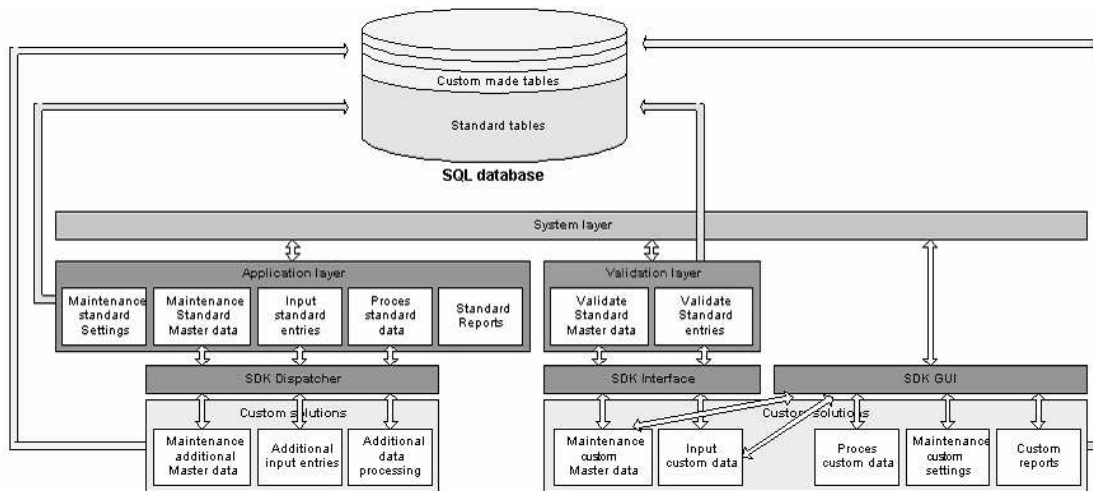


Figure 17 The e-Synergy and Globe SDK Architecture Implementation

Figure 17 shows the SDK interface to the complete software architecture of e-Synergy and Globe. In this picture the channels for communication between the SDK Gui, Interface, and Dispatcher and the applications are displayed.

4 Analysis

This chapter describes the resulting analysis from the case study at Exact Software. The analysis of the case study is a result of the research question whether and how e-Synergy and the ISKB are related. First, the processes within Exact Software are analysed, then the tools supporting these processes. Finally, the advantages and disadvantages of the Exact Software techniques and processes are described.

The four research questions were answered by first modelling the release and deployment processes within Exact Software. Once there was a clear understanding of the processes of development, release and deployment, the focus went to the other three questions. The second research question, “How are the processes of release and deployment supported by e-Synergy”, was answered using the modelled processes. The relationship of Exact Softwares’ e-Synergy product to the ISKB was evaluated once there was a clear picture of the features of the ISKB and the features of the e-Synergy software product. The final question was answered using the features of the ISKB and checking where they could improve the e-Synergy product and the software development process within Exact Software.

To model the release and deployment processes within Exact Software, interviews contributed mainly to the correctness and shape of our model. Documents from the e-Synergy repository gave the finer details of the design of these processes. After this, interviews were taken to check for consistency errors and gaps in the model. By working near the members of International Development some direct observations were made while watching the developers work on the software. Since the workflow for the Deliver project member had to be stored in e-Synergy as well, some participant observations were done and used to model the release and deployment processes. This section describes the differences and similarities for the development, release and deployment processes between Exact Software and Deliver. The development and release processes within Exact Software form a basis for the delivery and deployment processes. Many decisions made in these early processes have a significant impact to the processes of delivery and deployment.

To study how the tools e-Synergy, the Product Updater, and the Source Manager support the processes of release and deployment, there was mainly software study and document study. Some interviews were also done with members of the logistics department, the custom solution department, and some other developers to confirm our hypotheses. The studying of documents ([7], [8]) contributed greatly to this question because of the fact that there is much material available in e-Synergy about the use of e-Synergy for the processes of release and deployment.

The fourth research question, whether there were any improvements possible to the Exact Software way, was answered mainly through document study and interviews, giving some of the interviewees some feedback about functionalities in the tools supporting development and asking them whether an improvement could be made by replacing functionality with a similar functionality of the ISKB with all its options. From these comparisons, improvements and disadvantages were derived. These disadvantages, as stated here, and improvements, as stated in Chapter 5, were presented to Exact Software.

The case study at Exact Software has had some influence on the concepts defined by Deliver. The way in which Exact Software builds their software is through continuous software maintenance. This strategy is quite effective for the problem domain, yet not so relevant to deliver because the products are not volatile and not subject to significant changes. One striking characteristic of the Exact Software approach is the One-X architecture. The linking of all information systems to the information system storing the software knowledge is new to Deliver and will be taken into account in the future.

4.1 Strong Points

Exact Software has been a profitable company over the last 15 years and therefore their approach to software development and sales can be considered successful. The software produced by Exact is easy to use and effectively supports business processes for over 160,000 customers. When looking at Exact Software it can be seen that **simplification** is used as a typical strategy for dealing with potential problems. Exact Software is focussed on the sale of complex business software. As a result of these complexities in the problem domain, Exact Software has attempted to simplify the actual software structure and its development, giving developers more time to focus on customer needs. Simplification can also be found in the architecture of the software and the Exact Software product families. Since all the software is based on **one data model**, all the software can access this data model enabling multiple layers of users and therefore more customers. A striking shift has taken

place in Exact Software over the last five years as well, since the profit has shifted mainly to the support of software (60% in 2003, against 40% profit from software sales).

This Section lists all the strong points that were found during the analysis of our observations. The strong points are divided into a processes subsection, a tools subsection and a development approach subsection.

4.1.1 Processes

- Full deployment eases delivery
- Customisation and extensibility improve product quality
- Integration of knowledge eases release and deployment
- Subsidiaries enable Exact Software to quickly adjust
- Simplifying processes enables serving a large customer base
- One data model for all products

Exact Software delivers the full content of a software installation, including components and options a customer might never pay for or use. Exact Software has many reasons to deploy a complete application instead of just the requested functionalities. To begin with, much of the data model and the source code are shared among modules within one product. The main advantage of minimal deployment sets, being the fact that less data needs to be transferred to the customer, is cancelled out by the fact that bandwidth and storage space are cheap nowadays. Another clear advantage of delivering all the software to a customer is that only the Exact Software license file needs to be updated if a customer has bought an extra module, which **simplifies the task of delivery**. Exact Software does force itself to enable only runtime variabilities.

Customisation and extensibility are two main characteristics of the software produced by Exact Software, through their SDK and customisation departments, which gives Exact an advantage on the market. Deliver proposes the same features for customisation and extensibility for all software products and resembles the processes of Exact Software in this area.

The knowledge about workflow was never an integral part of the approach for Deliver even though it is knowledge about the software. Another advantage of “linking” all information about all processes is the fact that documents can be attached to targets, sources, and workflow. These documents, design documents for instance, can be used to explain and share knowledge about the products. The developers **use this knowledge extensively to ensure continuity**. Should a developer become unavailable due to job change, department change, etc, then the knowledge about the software is only partly lost. Another advantage of this “linking” of information is the fact that e-Synergy stores information about the people working on e-Synergy. It is therefore possible to determine with only two ‘clicks’, whether a developer is present and whether the developer can be contacted at present. Finally, the e-CRM (customer relationship management) and e-HRM (human relationship module) modules allow developers to see the source of bug reports and functionality requests. Another reason why the integration of knowledge is of advantage to Exact Software is that the processes of release and deployment are alleviated. Knowledge about customers is integrated and specific license files are always available. For more examples see Section 3.4.

Another effect of the simplification mindset is the creation of **subsidiaries with their own responsibilities**. Each subsidiary, which is part of Exact Holding, is responsible for its products and financial health. Exact Software can quickly adjust the organisational structure, through the creation and abolishment of subsidiaries, to economic and technological changes.

The simplifications of the release and deployment process have given Exact Software to opportunity to quickly **expand their customer base**. The full content delivery, Exact License Files, customer knowledge bases, customer portals and many process simplifications enable Exact Software to continuously communicate with and serve large amounts of customers.

The use of one data model for different products allows Exact to sell more products to support different business processes with the same information shared between these products. The fact that these products can therefore be integrated, such as back- and front-office applications, and sold separately, gives Exact Software an economic advantage.

4.1.2 Tools

- Product Data Management tool to create products

- Integration of knowledge eases processes
- Exact Software uses web delivery and update feedback

The Source Manager application which provides an interface to do batch uploads, batch downloads, and facilitates the locking of files, provides all the functionalities a developer needs from the e-Synergy repository.

One reason why this case study has been initiated is the fact that the software within Exact Software is stored in the e-Logistics **product data management tool**. The e-Logistics module supports product variation, product release, product instantiation, etc. These features are not all used by Exact Software though. Product variations are never implemented within Exact Software since all targets are delivered to a customer. Support for the feature of minimal delivery sets is available in e-Logistics, however, Exact Software does not use this feature because they do not divide the targets among different components.

The fact that **all data in e-Synergy is “linked”** makes it possible to connect workflow information to the targets and source files. This knowledge is used frequently in development and quality assurance. This property proves to be fruitful to the development departments but is also used extensively by other departments.

An example of this “linking” is that **when a customer downloads an update** of the software, a **feedback** entry is made into the customer relationship database stating the batch number of the downloaded version. The batch number is later used during support questions and other customer related issues.

Exact Software profits greatly from their **web delivery** mechanisms. The web delivery system that uses the Product Updater enables Exact Software to send updates to their customers without any cost except the web server.

4.1.3 The Software Development Approach

- Software development is seen as maintenance
- Exact Software reduces complexity and therefore man-hours
- Integration of knowledge about software saves time
- Integration of knowledge about software improves product quality
- Building software is propagated to developers, no build servers required

When looking at the development departments of Exact Software we see that their basic tasks are mostly **software maintenance tasks**. This can be explained by the way in which Exact Software works with these “spearhead” research and development teams who first develop a product to a fairly mature state and then hand it over to the development departments. This simplification has the advantage that the software structure delivered by the spearhead team is relatively stable.

Exact Software attempts to keep their software and their architecture as simple as possible. Exact Software claims that this simplification is required to build software in such a complex problem area. A simple software architecture does have many advantages, such as a short learning curve and easily changeable software. According to [14], in a complex problem area, software development gets even more complex. Exact Software is thus **reducing man-hours**. An example reduction is the **removal of build servers** from the development process. The developers now locally compile the software in a uniform build environment and upload the compiled software into the repository.

Exact Software uses the e-Synergy system to store the software system artefacts and all knowledge about the software. **Time is saved by e-Synergy** because employees can always look up developers who have done development work on some part of a software system that is of concern to that employee. The fact that information about the software is accessible to all developers results in more continuity in the development process if, for example, an employee needs to take over activities from another employee. More continuity in the development process results into **better product quality** and quicker development.

4.2 Points for Improvement

The research into the Exact Software release and delivery processes and comparisons to state of the art technologies have resulted into points for improvement for Exact Software. The points for improvement are a direct result from the comparison between the ISKB and the processes, tools, and development approach used by Exact Software. In Chapter 5 propositions are made to improve the points described here.

This section is divided into a processes subsection, a tools subsection and a development approach subsection.

4.2.1 Processes

- No sharing of problem area knowledge between departments
- Software piracy not addressed as a problem, even with infrequent releases (four times per year)
- Software structure not explicitly defined in product data management tool

The processes of development, delivery, and deployment as implemented in Exact Software have downsides as well. These downsides are mainly the result of simplification and the Exact Software business approach.

The main point for improvement, as seen by Deliver, is that the **software structure is not explicitly stored** in the product data management system, even though e-Logistics is well capable of storing and using such knowledge. Simplification has led to all targets being stored in one container item. This container item contains a list of all targets. These targets do not have any relationships among them.

Exact Software feels strongly that **software piracy is not a problem** since they use software aging, which is regular release of updates so that old versions have less functionalities. This point is cancelled out, however, by the fact that Exact Software does not force clients to update and the fact that much of the software developed by Exact is quite stable in its development. As an example, e-Synergy is only released four times a year.

Finally, the company structure of subsidiaries might ensure a financial insightful company, however, there is **less sharing of knowledge between departments** even though they are often working in a similar problem domain but with different technologies. An example is that Exact Software CS (a Unix based accounting package) solves the same problems as are solved in e-Synergy without sharing this information. Exact Software does not promote any knowledge sharing between subsidiaries.

4.2.2 Tools

- No support for versioning or branching
- Exact Software only enables pull mechanism
- Deploys destructive updates
- No rollback functionality

As stated in the Deliver proposal [1], each software development project should be stored in some kind of repository. Exact Software is no different in this, but the nature of the repository is influential on the other processes. The Exact Software repository is quite different from what Deliver builds their ISKB extension on. A clear difference is the fact that the e-Synergy repository **does not support versioning or development branches**. The Deliver intention is to enable maximal flexibility and maximal variability at all stages of software development. Deliver therefore feels that the repositories in e-Synergy are quite limited in their application domain. Simplification in this case leads to a downside because it limits the flexibility of the software development process.

Another difference is the fashion in which updates are delivered to the customer. Deliver would like to compare a deployed installation to a newly available installation through the use of feature descriptions [12]. Exact Software compares file and database version numbers to calculate the update sets. Finally, the mechanisms for delivery of updates and software at Exact Software differ from the techniques Deliver proposes. Deliver wishes to enable all types of update scenarios, such as push, automatic pull, and pull. Currently Exact Software only enables the **pull mechanism**. The process of deployment looks different within Exact Software from the way in which Deliver sees it too. Deliver wishes to see that updates are gradual, non destructive transitions enabling rollback and atomic state transitions. Exact Software, however, **deploys destructive upgrades**.

The Product Updater supports the delivery and deployment process. The Product Updater supports, with simplified mechanisms, the delivery of software and updates. The Product Updater also supports the deployment of the software and updates at the customer site. When looking at Deliver the Product Updater provides a limited set of functionalities. Two commonalities between the Product Updater and Deliver are the fact that the Product Updater checks some dependencies and disk space, and that to the Product Updater the download location for updates and software is abstract (can be a repository, a hard disk location, or a CD). Some striking differences are the fact that the product updater **does destructive updates, does not enable rollbacks**, and uses only simple reporting functions. The Deliver approach for updates is different because features like rollback should be

implemented. Also, the Deliver solution supports reports back to the software vendor whether the update was successful.

4.2.3 Software Development Approach

- Developers are responsible for correctly building the product
- Hotfixes could introduce more errors, branches would be safer
- Simplification leads to software architecture not being stored explicitly
- No integration testing
- No versioning system for development department

There are some downsides to the software development approach of Exact Software. Reducing complexity can have negative effects when it results in less flexibility. An example of how reducing complexity has led to less flexibility is the fact that without build servers, the **developers themselves are responsible for the correct compilation** of products which introduces risks to software quality. A third disadvantage is when a hotfix for all releases (see Section 3.7) is made with an error the hotfix **requires an extra hotfix**. This procedure therefore involves frequent risks.

Simplification in the development process has led to a process in which **the software architecture is not stored into the product data management system** explicitly. Not making the structure of software explicit reduces software quality and ease of adaptation.

Another difference in the viewpoints of Deliver to the development and release processes of Exact Software is the fact that there is **no integration testing of components** that are reused in different packages. The Deliver team wishes to encourage integrity checking to ensure product quality.

Finally, simplification has resulted in a one dimensional version system, where development can only edit one version. Having one version of software has the advantage that merging is not required and complexity is reduced. Downsides are that developers cannot **sidestep the development process** for prototyping and for holding back functionality when near to a release, thus forcing the developer to work in small incremental steps of development.

5 Suggested Improvements

After studying the ISKB and the e-Synergy product, many functionalities that are part of the Deliver picture of the ISKB, could improve e-Synergy if implemented. This chapter describes the proposed improvements, their expected impact and development time, and a description of the reasons why e-Synergy does not implement these at present. These improvements are considered only if they increase the product quality, reduce the total workforce, or create possibilities to access other markets.

Table 5 All proposed improvements and their effects

	Estimated Effort in man months	Product Quality	Time Reduction	Market potential	Main Reason
Target Dependencies	12	++	++	+	Simplifies product composition, branching, and deployment
Product Composition (with TD)	2	++	++	+	Guarantee consistency of created products
Deployment Manager	2	--	-	-	Deploy less (MB) with same options
Change Sets	8	+	--		Minimize risks through development branching
Automatic Pull	2	++	++	+	Optimize update process
Rollback	8	++	+	+	Guarantee more stability of products
Updater Feedback	2	++	+	+	Guarantee higher level of support

Table 5 summarizes all proposed improvements and their expected effort and impact. The plusses and minuses indicate the impact on Exact Softwares processes on a scale from large positive impact “++”, to no impact “”, to large negative impact “--”.

5.1 Target Dependencies

One improvement of the system would be the automated tracking of dependencies among source files and project these relationships to their targets. These dependencies can be used to calculate closed sets of binaries and to determine impact on targets of changes to a source file. In an ISKB this knowledge is useful for impact analysis, deployment management, and product composition.

Currently the e-Logistics product data management system already supports the feature of dependencies for items. These dependencies can easily be used to specify the dependencies among sources and targets. However, the main reason why this feature is not being used is because the real problem is to keep these dependencies up-to-date. Automating this functionality would require some kind of source file interpretation. To implement source file interpretation, a product that is being sold to customers who do not need the functionality of source code interpretation must be changed. Considering this feature will not be used by any of the other 160,000 customers, adding the feature of code interpretation would not be useful.

This improvement could help Exact Software with development, product composition, and completeness checking of delivered products. The development process would improve because this improvement enables risk analysis for developers. Currently, when a developer wants to change a library she first checks with a text search tool where the library is used. This would no longer be necessary if the dependencies were available in the system. This would save the developers time. Product composition would benefit from this improvement as well, because when the dependencies have been made explicit, it is easier to see what components and libraries are required for functionality. Currently, to put together a product, targets are collected and grouped under a sales item. No guarantees can be made, however, about the closures of dependency relations. This means that, even though it is quite easy to put together a product, no guarantees can be made about whether this is a complete

product or whether it is missing some targets. The calculation of complete sets using the dependency relationships is already supported manually by e-Logistics. The dependency relationships would greatly improve product quality. Also, it would become possible improve simultaneous development of different components depending on the same components. Currently when a component belonging to one product family is changed the component is only tested within that product family. With explicit knowledge about dependencies, the developers would know whether the component is being used within other product families as well, improving overall product quality.

5.2 Deployment Manager

Currently e-Synergy deploys complete sets of binaries, and through runtime variabilities the right modules are activated. This enables hackers to edit the ELC file to obtain more modules than they have paid for. In the ISKB it is a definite requirement that partial sets be deployed at the customer side because the Deliver groups' model of a software product is, that one software product can have many shapes and sizes, because of platform dependability, product subset instantiations, and many other factors. The implementation of a deployment manager for Exact Software products is (again) dependent on the availability of dependencies of binaries, so that complete sets can be guaranteed. Implementing such a change within Exact Software would also mean fundamentally changing the structure of the Product Updater and the product itself. The e-Logistics unit of e-Synergy does support the delivery of minimal sets. Exact Software never implemented their product structure into e-Logistics to enable the delivery of minimal sets.

The reasons why a deployment manager should be implemented are software piracy, redundancy, and product modifiability. Exact Software claims to have solved the software piracy issue with software aging and the nature of their software. They claim that piracy is not an issue because of the fact that companies are so dependent on the software and the updates (4 times a year). Next to this, Exact Software does not see the redundancy issue since the full Globe Client installation and the e-Synergy installation comprise only around 100 megabytes. Product modifiability, however, is not so easily ignored. A deployment manager would allow for many options, for instance optional components, interchangeability between components, and light versions. All these options would make the product less static and therefore more adaptable to market changes. The possibility to provide separate conflicting options would also become available. Exact Software does not find that the gains of implementing a deployment manager weigh up to the risks and disadvantages, mainly because Exact Software deliberately avoids variability other than customisation in their software.

5.3 Change Sets

Currently, since there is no change tracking system or version history being recorded, there is one version being worked on at the same time within Exact Software development. However, developers do attempt to sidestep the development process occasionally when they wish to prototype functionality or hold back large changes created right before a release. Developers then choose to keep a local copy that they will then submit after the release promotion. Some developers choose to upload this local copy as an attachment to a document into e-Synergy.

The reliability of this system is at question. A solution to the problem of holding back changes to the next release would be if branching were introduced to the e-Synergy system. In the implementation of an ISKB, Deliver requires the system to support branching. Branching in e-Synergy would enable Exact Software to introduce more variability and therefore flexibility. This flexibility would then enable Exact Software to reach different markets, by releasing their products for new platforms, for instance. Finally, a branching feature would provide a mechanism to hold back large changes for a promotion, thus increasing reliability of the product and decreasing the need for quality assurance personnel.

5.4 Automatic Pull Mechanism

A proposition was made in Exact Software to enable some kind of automatic pull mechanism for the Product Updater. In this proposition a reference was made to the automatic pull mechanism for Windows XP [13]. In an ISKB there should be support for all kinds of delivery mechanisms, among which (automatic) pull and a (automatic) push update mechanism so that each scenario is supported.

At present, the Product Updater can only download the software when the customer starts the Product Updater. A situation that would improve product availability would be if there was a memory resident application that checks for updates periodically. The update would then be downloaded without the user noticing, preferably at a quiet time at which no other downloads are taking place. A different scheme would be if the Product Updater checked for updates every time the computer is restarted. The computers that run e-Synergy and other Exact Software products are generally servers which should provide service 24 hours a day, though, so this scheme

would only work for computers using Exact Software client applications like Globe. This facility will soon be implemented in the Product Updater, to move along with current state-of-the-practice (like Symantec Live-Update, and Windows XP auto-update.) It will improve the product quality and usability, because users are notified every time a new update is available.

5.5 Rollback

At present it is not possible to perform a rollback (undo) when an installation or update is not successful, or the customer is not ready for features and functionalities, because updates are not atomic and no other rollback functionalities are provided.

Currently this is not built into the Product Updater for a number of reasons. In the first place the software vendor gets less repeated bug reports and functionality requests because all customers are running the same, most recent version. Some software developers even require their users to work on the most recent version of a product before they grant support to their customers. Exact Software only recommends upgrading to the most recent version when blatant errors are encountered or functionality requests have been placed. Another reason for not implementing rollbacks is the fact that updates are usually accompanied by database changes and it is a complex process to undo database changes, if possible at all.

The impact of this improvement is of a large scale, compared to its profit. Implementing this improvement will require a complete rethinking of the structure of Exact Software products. Updates need to be designed as incremental development steps for a product (the database scripts already are designed as such) instead of a mere “latest set”. Also, The Product Updater must be adapted to implement version numbers. This change would benefit product reliability, however, because if an error occurs during an update, it can be undone by initiating a rollback.

5.6 Updater Feedback

A final improvement would be if the Product Updater provided other feedback than just the “download complete” message to the Exact Software server. At present it is possible to see what version a customer recently downloaded, but not what version the customer actually is working on. Updater feedback could also be used to assemble information on how often updates fail, how long the average update takes, and what customers have problems updating. Such mechanisms could greatly improve quality of support. An ISKB requires such information if no other knowledge was available about the customer. Once again, this is a scenario a general deployment environment should support.

Implementing updater feedback can happen at different levels. Currently, already, when installing or updating a product the Product Updater registers what actions are taken during the installation or update. The simplest way of implementing Updater feedback is by sending this file back to the server after completion. The communication channels used here could improve services if the product performed some check after installation or update and sent the results of the check to the server. In this way, even simple warnings generated by new installations can be analysed without the customer ever having to find out. Updater feedback would improve product quality and service quality, because Exact Software can use the feedback information for immediate support and future upgrades.

5.7 General Improvements

The organisational structure of Exact Software is not always positively influencing the development departments. There is not much sharing of information between developer subsidiaries even though the infrastructure for sharing knowledge, e-Synergy, stores all produced documents open to all developers. The main result of this organisational structure is that domain knowledge is not being shared, even though two developers might be solving the same problems. Also, since most development teams use a different programming environment, code cannot be shared. The domain specific problems remain the same though.

This organisational structure is also holding back integration testing. When a target is changed on which multiple products are depending, it is usually only tested by the developer on one product. The fact that developers hardly share knowledge results into limited integration testing. Exact Software could counter the effects of this by organising the knowledge present in each subsidiary. The organisation of knowledge will lead to an improved sharing of knowledge, through meetings and internal workshops, of knowledge and solutions. The current e-Synergy infrastructure, with for instance SDK portals, already suggests such sharing should be encouraged. These improvements would improve product quality and reduce the workforce because problems are solved just once.

6 Conclusions

This chapter states the conclusions of the case study, formed by analysis of the case study results.

Exact Software has been a profitable company over the last 15 years and therefore their approach to software development, support, and sales can be considered successful. The software produced by Exact is easy to use and effectively supports business processes for over 160,000 customers. When looking at Exact Software it can be seen that simplification is used as a typical strategy for dealing with potential problems. A clear example of this simplification is the fact that Exact Software delivers all components to customers for a product, and later activates the actual components purchased by that customer. This simplification reduces complexity for the delivery of a product and thus simplifies delivery to a large customer base.

In the areas of software release and delivery, Exact Software uses many state of the art techniques. An example is the use of a product data management tool to manage Exact Software products. Another characteristic of Exact Software products is that they are extendible through a supplied SDK, which enlarges the market for these products. The fact that all data is stored into one data model and accessible to all software products created by Exact Software makes it possible for different layers in an organisation to use different and more appropriate Exact Software products. Finally, Exact Software delivers updates over the Internet and uses automatic feedback from the customers installed product to increase product support.

The Exact Software approach is different from the Deliver approach to software release and delivery. Exact Software does not store the structure of their software products explicitly and does not keep track of dependencies among components. Component dependencies would improve integration testing and product composition. Exact Software also uses destructive updates and does not enable rollback functionalities. Deliver sees incremental updates as a necessity to retain product stability. Finally, Exact Software does not allow branching and does not use versioning besides the version schemes explained in Section 3.7.

This report describes the observations done by Deliver during a case study at Exact Software. The final conclusions are that e-Synergy is not an instantiation of an Intelligent Software Knowledge Base. However, many of the characteristics found in e-Synergy can also be found in the Deliver Intelligent Software Knowledge Base.

Acknowledgements

We would like to thank Exact Software for enabling this case study. We want to thank Raymond ter Riet, Arco van Nieuwland, Sebastien Tout, and Nenad Borota, who all have tried their very best to communicate the processes and working culture of Exact Software. We would also like to thank Paul Klint for critically reviewing this document and Tijs van der Storm for his endless stamina in discussing the case study.

7 References

- [1] Intelligent Software Knowledge Management and Delivery, NWO Jacquard Grant Proposal – P. Klint, S. Brinkkemper – 2002
- [2] Case Study Research, design and methods – Robert K. Yin – Sage publishing 2003, 3rd edition
- [3] Qualitative Methods in Empirical Studies of Software Engineering – C. B. Seaman – IEEE Transactions of Software Engineering, 25(4), pp. 557-572 - 1999
- [4] Software Release Management - Hoek, Hall, Heimberger, Wolf – 1997
- [5] A cooperative approach to Support Software deployment using the Software Dock - R. Hall, D. Heimberger, A Wolf – 1998
- [6] Product Data Management and Software Configuration Management – Dahlqvist et al. - 2000
- [7] One-X architecture – <http://www.Exact Software.nl> – Hugo Scheepens - 2004
- [8] SDK - Deployment introduction 06.703.958 – Toine Hurkmans - 2004
- [9] Verisign. Verisign certification practice statement. - Technical Report Version 1.1, Verisign, Inc., Mountain View, CA. - August 1996
- [10] E. Rescorla and A. Schiffman. - The secure hypertext transfer protocol. - Internet Draft - May 1996
- [11] Discouraging Software Piracy Using Software Aging - Markus Jakobsson and Michael K. Reiter - Carnegie Mellon University - DRM 2001
- [12] Variability and Component Composition – Tijs van der Storm – Centrum voor Wiskunde en Informatica, Technical Report – 2004
- [13] Software Update Service Overview – White paper, <http://www.microsoft.com> - 2002
- [14] Philip L. Glass – Complexity 25 % problem adds 100 % complexity to the software –
- [15] Software Architecture and Software Configuration Management – Bernhard Westfechtel, Reidar Conradi – 2003
- [16] BizTalk.org – a briefing – Neil Hutson – Microsoft, Reading, United Kingdom – 2001
- [17] Active Server Pages - Homer, A., et al - Wrox, Birmingham, UK - 1997
- [18] Inside COM. - Rogerson, D. - Microsoft Press, Redmond, WA – 1997
- [19] Looking For Win/Win Solutions between Lab-work and Hands-on Experience in IT Research – Hoorn, van Dam, Fambach, Nieuwland, van der Veer - Proceedings of SIGCHI.NL 2004 (pp.). New York: ACM Press. – 2004
- [20] *Streamed Lines*: Branching Patterns for Parallel Software Development - Brad Appleton, Stephen P. Berczuk, Ralph Cabrera, Robert Orenstein - PLoP – 1998

8 Appendices

8.1 Metrics

The following report was generated through e-Synergy and lists all the parts of the e-Synergy product.

Table 6 item counts for the e-Synergy and Globe 2003 products

	e-Synergy	Globe
DLL – registered	21	511
Crystal reports (rpt)	1	977
<None>	2645	28
C/C++ header file (h, tlh, tli)	307	1947
VB form resource (frx)	67	1580
C++ source file (cpp)	260	2186
HTML page (htm)	10	56
Executable (exe)	55	605
Active Server Pages (asp)	1507	2
Developer Studio Project (dsp)	41	436
Development tool	4	17
DLL	146	787
Shared C/C++ header file (h, tlh, tli)	2	168
Option	15	269
Term	294	30450
SQL Script (sql)	461	361
VB project file (vbp)	156	1396
Other	270	3235
ActiveX (ocx)	5	114
VB class file (cls)	915	6395
VB module (bas)	268	2546
VB form file (frm)	90	2225
C/C++ library (lib)	16	58
Resource script (rc)	50	431
Database table (definition)	75	282
Images (gif, jpg, bmp, ico)	653	3689
Help files (hlp, chm)	0	1
XML Document (xml)	6	1
Help documents (HTML files + pictures)	0	14073
<None>	0	9136
Selection screens (slc)	0	1195
Layout	0	244
Menu definition (mnu)	0	97
Setting	0	39
Database view (definition)	0	8
Shared C/C++ header file (h, tlh, tli)	0	690
Shared VB module (bas)	0	165

This table shows the item counts generated by the e-Synergy product. All the items shown in this list are predefined items stored in the e-Logistics component. The Globe product is being developed by a team of 180 programmers in Malaysia and e-Synergy is being developed by 20 programmers in Delft. The table shows that e-Synergy is mainly developed in ASP whereas Globe is mainly developed in visual basic. <none> are parts in the physical world like e-Synergy service contracts or legal documents that come with the product. Options are optional product parts and do not include service contracts and the like.

8.2 Interviewees

Table 7 Interviewees

Interviewee	Mark Hakkenbroek
Function	Unit Manager Maatwerk Zuid
Reason for Interview	Custom Solutions & Variability
Main question	Do Custom Solutions introduce variability?
Interviewee	Ed van Heeckeren van Brandsenburg
Function	Delivery & Services
Reason for Interview	Study the final phase of delivery
Main question	How is software delivered to the logistics department and how does it get to the customer?
Interviewee	Paul Kaesler
Function	Developer
Reason for Interview	Previous SCM solutions
Main question	Did e-Synergy inherit anything from previous SCM systems?
Interviewee	Carolien Verrips
Function	Sales Manager
Reason for Interview	Software delivery, the ELC, Knowledge about customer
Main question	What do the processes look like from a sales point of view?
Interviewee	Emile van Bergen
Function	Assistant Customer Support Manager
Reason for Interview	Deployment
Main question	What does the process of customer deployment look like?
Interviewee	Nenad Borota
Function	Developer
Reason for Interview	Product Updater, requirements for deployment
Main question	What does the deployment process look like using the Product Updater?
Interviewee	Leo van Houwelingen
Function	Senior Developer
Reason for Interview	Development
Main question	What does the development process look like within Exact Software?
Interviewee	Sebastien Toet
Function	Senior Research Staff
Reason for Interview	Research and Development
Main question	What are Exact Software Solutions going to look like in the future? (How) will the software development process change?
Interviewee	Arco van Nieuwland
Function	Commercial Director e-Business + one of the founders of Exact Software
Reason for Interview	Exact Software, the big picture
Main question	What does product composition look like on all levels?
Interviewee	Raymond ter Riet
Function	Deputy Director Development
Reason for Interview	Exact Software development
Main question	What does software development look like on all levels?

8.3 Exact Software Product Listing

Assortment Alliance/MFG

Bavaria

CUBIC pro

Custom Solutions

DB-Soft für Windows

Diamante

Dimoni 6

e-Synergy

Exact Software Compact 2003

Exact Software Enterprise SQL

Exact Software Financials C/S Version 2

Exact Software Generation XL

Exact Software Globe 2003

Exact Software Group Press & Brand

Exact Software Lohn XL/XXL

Exact Software Pro

Exact Software Siigo

Excellent

Grote Beer voor Windows

Internal Products

JobBOSS

Macola Progression 7.x

MAX

Paymate New Generation

Shipping Automation System

Siigo for Windows

Soft-2000-Windows

Soft-Research

Szymaniak ProSoftware Windows