Wageningen University & Research Centre
Department of Social Sciences
Management Studies Group

Valuation of software

Student:	Martijn Klaver
Student number:	881201 438 130
Date:	31 August 2011
Supervisors:	Prof. Dr. P. van Beek and Dr. M. Leloux
Company:	Leloux Science & Business
Contact person:	Dr. M. Leloux

Foreword

This report is the result of a Bachelor thesis executed at the Management Studies Group of Wageningen University & Research Centre. Mainly in the months June until August 2011, I have worked on this project. This report consists of a literature research performed for the consultancy enterprise Leloux Science & Business. I have got an insight in the software branch and the valuation and protection possibilities.

I would like to thank my first supervisor, Prof. Dr. P. van Beek, for his help during the development of this report. Furthermore, I would like to thank my second supervisor Dr. M. Leloux, for commissioning the research assignment and giving advice. Finally, Dr. F. Foruin I want to thank for her advices during the start of this thesis.

Martijn Klaver

Abstract

This research project is focused on valuation methods for software in the case of a transaction. Different business models for software are playing a role in the valuation possibilities.

Based on literature study, it was found that different software types exist. Mass market software, embedded software and internal use software have their own characteristics. The type of software is important to know for a correct valuation. From literature four valuation methods were identified: direct assessment of future income, R&D spill-over, real options valuation and market capitalization. Each of these methods has a focus on a part of the software in an enterprise. For the purpose of valuing software no single method is employed as the best valuation method. Crucial for an optimal valuation is the combination of various valuation methods because no one specific valuation method is the best for valuating software.

All the software types present the same unique problem in valuation, that is, that in order to ensure continued usefulness and applicability, software must be periodically updated so that it remains up to date. These maintenance costs are generally more than 60 percent of the total costs. Protection of software can be done to make software more valuable. The protection measures discussed in this research are: copyright, licensing and encryption, trademarks, software patents and trade secret law.

Table of contents

1. Introduction

This chapter gives an introduction to a research assignment commissioned by Leloux Science & Business. This research assignment, which mainly consists of a literature research, deals with valuation of software and has been carried out as a Bachelor thesis (12 ECTS). The purpose of the introduction is to draw up a framework for our research (Wilkinson, 1991). By making use of this framework, a better understanding can be obtained about the position of our project in the context of the whole valuation market.

In this first chapter, the design of the research will be explained. First of all, some background information about Leloux Science & Business is given. Second, the conceptual design of the research will be explained. In the conceptual design some background information on the design of the research is given, the problem is described and research questions are formulated (Verschuren and Doorewaard, 1999). After that, the research framework is made. Finally, the research strategy and planning are given in de technical design.

1.1 Leloux Science & Business

Leloux Science & Business is a small Dutch consultancy firm established in 2004. It is specialized in the commercialization of knowledge. The commercialization of inventions and the valuation of knowledge is a complex process and require combined expertise in the fields of Research and Development management, valuation and technology transfer business (Leloux Science & Business, 2011). Leloux Science & Business has developed this expertise and used an international network of professionals for their services. These services consist of, for example, developing business models and financial scenarios, supporting the commercialization of projects and negotiating in partnership and licensing contracts. Also, the company assists its clients in developing their patent strategy, in line with their business strategy. Some projects deal with the commercialization of software (components). Until now, the valuation possibilities for software products are mostly unknown. However, to be able to advice clients dealing with the commercialization of software, valuation aspects are important. From that point of view, it is important to know for Leloux Science & Business how software can be valued in different situations and business models.

When a methodology is found to valuate software (components), the software can be commercialized with the existing expertise within Leloux Science & Business. The commercialization of software can be used in transaction settings like takeovers, mergers, joint ventures and selling transactions.

1.2 Conceptual design

In the conceptual research design the focus is on clarifying and limiting the field of research (Verschuren and Doorewaard, 1999). The conceptual design consists of three different parts. First of all, the problem statement and objective of the research are defined. Thereby an answer is given on the question: 'what are we going to study?' Second, a research framework is designed, in which all the steps which have to be taken in the research are presented. This framework provides an answer on the question: 'how are we going to study?' For this research, six research questions are formulated. At the end, some definitions of concepts are given, in order to clarify the meaning of some specific terms used in the research.

1.2.1 Problem statement

Based on the given background information, the problem statement to which the research should contribute can be described as follows.

Leloux Science & Business has the expertise to valuate and commercialize knowledge in different aspects. For software products and software based applications, the expertise of Leloux Science & Business on valuation and commercialization is not completely developed yet. This research focuses on application software, further explained in chapter 2. To strengthen the position of Leloux Science & Business in the field of application software valuation, the company has to get a better view of the different kinds of software products, the different kinds of business models in the field of software, protection possibilities for software and the way in which software can be valued in a transaction process. With a transaction, the transfer of the application software product from one party to another party is meant. This transaction setting is important for the clients of Leloux Science & Business. The clients make use of the consultancy firm for advice and support for the commercialization of (software) inventions and the valuation of knowledge about software. Also protection of software is an important subject because the protection possibilities for software are related to the valuation possibilities. A better protection of the software implies higher value.

When Leloux Science & Business has the expertise of application software valuation, it is possible to serve the customers with commercialization and valuation questions about software. For Leloux Science & Business this implies a greater market potential and probably more consultancy customers with their related software questions.

1.2.2 Research objective

The research objective gives direction towards the research and can make a significant contribution to the clarification of the subject. The objective in relation to the problem is: *to give Leloux Science & Business insight in how different kinds of software products can be* *valued and protect, by investigating the different combinations (see Table 7.1) of software valuation and protection.*

Kind of research

This research can be considered as a practice oriented research. The purpose of practice oriented research is to contribute to the development, implementation and evaluation of solutions for practical problems which exist within organizations ('t Hart, Boeije and Hox, 2007). This research tries to identify the different valuation possibilities for the different kind of software products. Leloux Science & Business can use this information for questions from their customers in the field of software valuation.

1.3 Research framework

In figure 1.1 a model is developed, which gives a broad overview of the way the research is constructed. Via the steps in this framework, the research objective has to be achieved. The research questions (RQ) are integrated in the framework. Using the framework, RQ1 to RQ6 are drawn to answer the main question: *To what extent can software be valued in the case of a transaction?*

This research starts with a literature study which describes the characteristics of software and which different types of software exist. Thereafter, there is an investigation of which

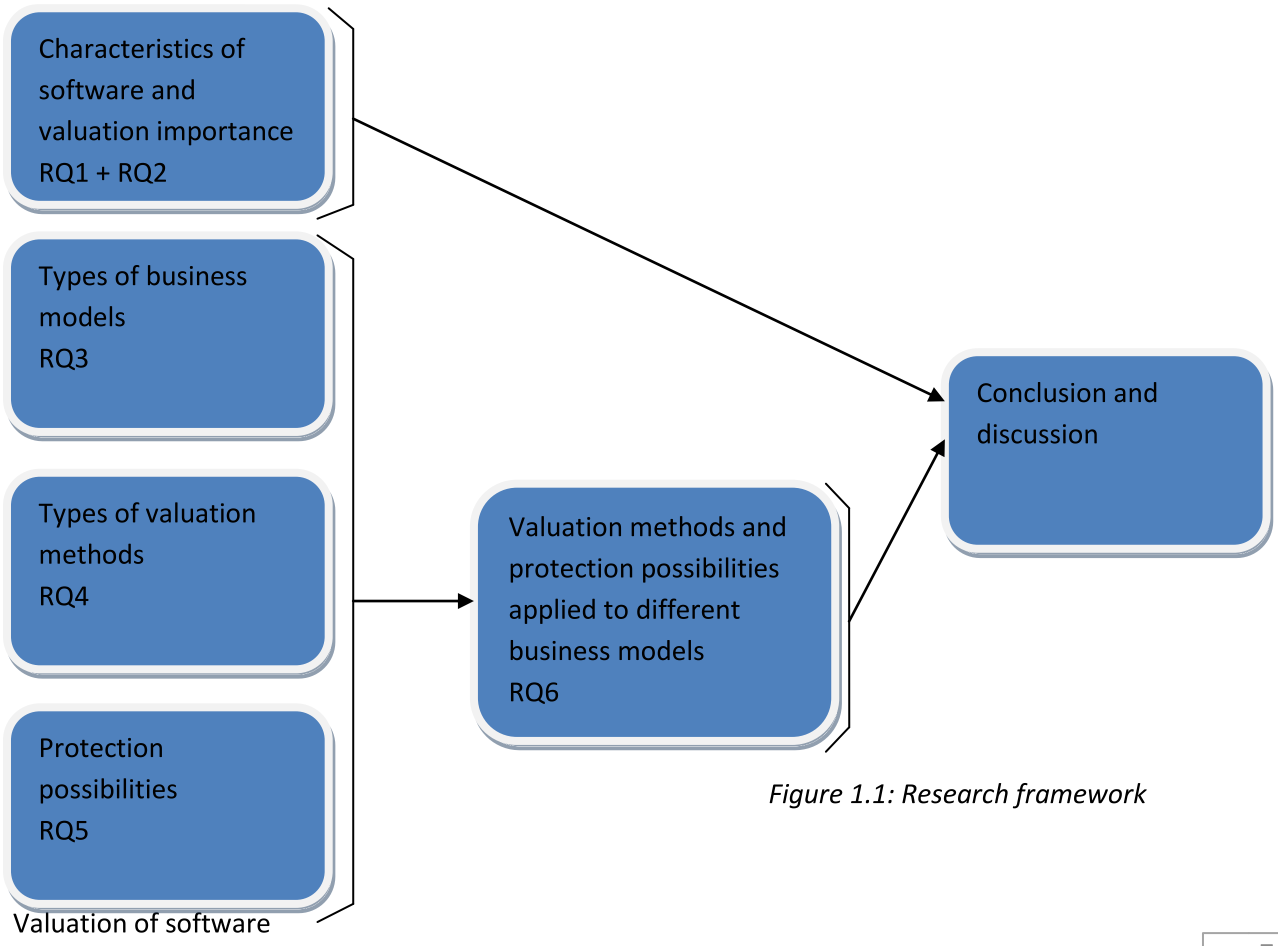
Characteristics of software and valuation importance RQ1 + RQ2
Types of business models RQ3
Types of valuation methods RQ4
Protection possibilities RQ5
Valuation of software
Valuation methods and protection possibilities applied to different business models RQ6
Conclusion and discussion

Figure 1.1: Research framework

types of business models exist. When the different kinds of business models are clear, the focus is on the types of valuation methods and the question*: When is it important to determine a value for software?* It is good to know how protection plays a role in software valuation, so it becomes more understandable which protection measures must be taken into account in relation to valuation. After the first five sub-questions, the focus is on the relation between the business models, valuation methods and protection possibilities. Information from research questions three, four and five come together in RQ6.

1.3.1 Research issue

Below, the main question (MQ) and six sub-questions (RQ1 through RQ6) are given.

Main question:

MQ: To what extent can software be valued in the case of a transaction?

Sub-questions:

RQ1: What are the characteristics of software and what types of software exist?
RQ2: When is it important to determine a value of software?
RQ3: What types of business models exist in relation to software?
RQ4: What types of valuation methods are available for software?
RQ5: To what extent can software protection play a role in software valuation?
RQ6: Which valuation methods and protection possibilities can be applied by the different business models?

As stated earlier, this research is a literature research, so all question are answered by using a literature study for each sub-question. RQ6 can be answered after having the answers of RQ3, RQ4 and RQ5. When all sub questions are answered, an answer on the main question is given, stating to what extent software can be valued in the case of a transaction.

1.3.2 Definition of concepts

In order to standardize some important terms during this research, a list with the definition of frequently used concepts is given below.

Intangible assets:

Claims to future benefits that do not have a physical or financial form and cannot be seen, touched or physically measured. Examples are patents, bioengineered drugs, brands, strategic alliances, copyrights, trademarks and trade secrets (Hand and Baruch, 2003).

Intellectual property (IP):

Ideas, inventions, discoveries, symbols, images, expressive works (verbal, visual, musical, theatrical), or in short any potentially valuable human product (broadly, "information") that has an existence separable form a unique physical embodiment, whether or not the product has actually been "propertized", that is, brought under a legal regime of property rights (Landes, 2003).

Software:
Coded instructions in the form of programs that perform certain tasks using a computer's hardware. Software includes a computer's operating system and all its applications (colloquially, apps). These are written in source code (a programming language such as Java or C++) and are then converted by a compiler program into binary (Dictionary of Media and Communication, 2011).

Transaction:
A transaction consists of a trade of values between two parties. In a transaction one party gives X to another party and gets Y in return. This transaction can be a monetary transaction, when the transaction involves money, or a barter transaction when the transaction involves services as well as goods in return (Kotler and Armstrong, 1994).

1.4 Technical research design

The technical design focuses on the practical implementation of the research. It answers the question: 'how should it be executed?' The technical design consists of different parts (Verschuren and Doorewaard, 2009). First of all the research materials are given. Those are the sources from which the research will be executed. Next to that the research strategies are described. In this part the question is answered: 'what needs to be done to effectively arrive at a sound answer to the prime questions?' Furthermore, a research planning is given, this part shows the time planning of the research.

1.4.1 Research material

To following research materials are used to give an answer on the research questions.

1. Scientific literature from professional and academic journals about software, software valuation, software protection, intellectual property and business models (library, digital sources, information available at Leloux Science & Business).
2. Information gathering from parties involved in the software sector. Especially focused on software valuation.
3. Websites about the topics mentioned by point one in this section, with professional background.

1.4.2 Research strategy

The research strategy shows how the literature research will be executed. The information gathering part of this research consists of what is written about the subject in the literature and websites. For the literature research the search engines Scopus, Web of Science, JSTOR, Google Scholar and the Wageningen catalogue are used. Search queries in these databases used are: “Software, intellectual property, valuation, valorization, business models, patent and copyright.” Also information about software and intellectual property which are available within Leloux Science & Business is used.

1.4.3 Research planning

In the table 1.1 the time table of the research is shown. In total the project takes 13 weeks fulltime to work on. Week 1 starts at 6 June 2011.

Table 1.1: Time table of the research project.

Week number	Action	Start	Finish
1	Meeting supervisor / company about topic		Week 1
2	Find literature	Week 2	Week 6
3	Structure literature	Week 3	Week 6
4	Start answering RQ1, RQ2, and RQ3.	Week 4	Week 6
5	Continue literature part		Week 6
6	Start answering RQ4, RQ5	Week 6	Week 8
7	Continue literature part, meeting		Week 8
8	Start answering RQ6	Week 8	Week 9
9	Ending literature part		Week 10
10	Start writing conclusion and discussion	Week 10	Week 11
11	Circulate provisional final report, meeting		Week 12
12	Last points	Week 12	Week 13
13	Final presentation and submission final report	Week 13	Week 13

2. Characteristics and types of software

This chapter answers research question 1: *What are the characteristics of software and what types of software exist?* First the characteristics of software are defined. This is important because first the background of software must be clear, thereafter it is possible to build on this information. Second, different types of software are explained.

2.1 Characteristics of software

All types of software are written in a source code. With a specific program this source code can be converted into a binary language. This binary language can be used by hardware like printers. For using software, hardware is needed.

The total value of assets of an enterprise consists of current assets, like inventory, and fixed assets. Fixed assets are those that will last a long time, such as buildings (Ross et al., 2008). Some fixed assets are tangible, such as machinery and equipment. Other fixed assets are intangible, such as patents, software and trademarks (Ross et al., 2008). Software is an intangible asset because it cannot be seen, touched or physically measured. Basically software is only source code which provides tasks and gives a specific interface. Software, as an intangible asset, can be protected as an intellectual property (IP).

Software can be divided into system software and application software (Schaefer, 2002). Both software types are written in source code. The function of source code is to provide instructions (algorithms) that the computer system can understand in a way that it can perform a processing activity. Examples of source code are C++, Java, SQL, and Visual Basic.

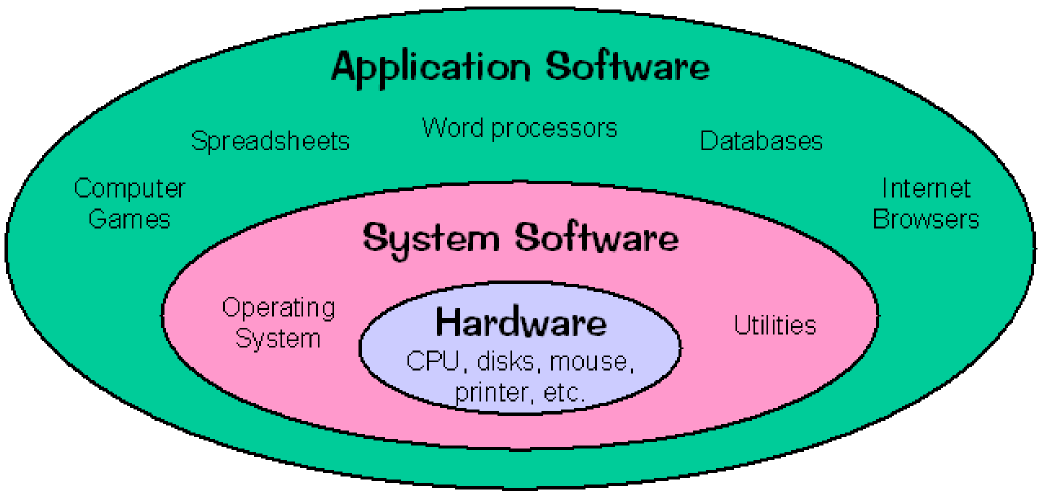

Figure 2.1: Overview of system and application software (Balci et al., 2007).

System software are programs that coordinate the activities and functions of the hardware and various other programs (Schaefer, 2002). This software "provides a general programming environment in which programmers can create specific applications to suit their needs. This environment provides new functions that are not available at the hardware level and performs tasks related to executing the application program" (Nutt, 1997). As shown in figure 2.1 on the previous page, system software acts as an interface between the hardware of the computer and the application software that users needs to run on the computer. System software perform basic tasks, such as recognizing input from the keyboard and sending output to, for example, the screen. Also it ensures that different programs and users, running at the same time, do not interfere with each other. System software provides a software platform on top of which other programs, like application software, can run (Balci et al., 2007). Examples of system software are BIOS, Microsoft Windows, Mac OS and Linux.

Application software are programs that help users solve particular computing problems (Schaefer, 2002). Application software makes use of system software for the communication with the hardware of the computer. Application software is a broad concept. Word processor and spreadsheet like Microsoft Office are application software but also Media players, web browsers, games and the Graphical User Interface (GUI) are examples of application software. The applications at the computer make it possible to use it for specific tasks.

In this research the focus lays on application software because this software type is the most common type that producers make. All the programs for games, new business administration programs, websites for selling items and media players are examples of applications software which makes use of one of the well-known system software programs like Microsoft Windows or MAC OS. For valuation of software the application software is taken into account as the reference software.

2.2 Different application software types

Application software is used and sold in different kind of categories. For valuation issues the different categories have their own valuation possibilities. The next topics gives an overview of the different kind of categories for application software and the maintenance costs related to software products.

2.2.1 Mass marketed software

Mass market software is for example the software packages of Microsoft Office. Mass marketed software is the stand alone software which is sold via the internet, in shops or as an optional package available with the purchase of hardware (Kemp, 1987). It is not embedded in, for example, computers. Mass marketed software products are easy to reproduce at a cost that is low. Its distribution is relatively simple and can occur via the internet too. Even at a competitive price, each incremental sale generates much more profit than the incremental cost of production (Wiederhold et al., 2009). Mass marketed software generally contains an externally visible notice which states that opening the plastic cover, or clicking on the acceptance button, constitutes acceptance of the terms stated thereon (Kemp, 1987).

2.2.2 Embedded software

The principal role of embedded software is the interaction with the physical world (Lee, 2002). It is executed on machines that are not necessarily computers (Lee, 2002). Software is embedded in everything today, enabling the functionality of products that touch every industry. Low replication costs apply to the software that is embedded in so many of seemingly tangible products, from mobile phones to aircrafts. For example in a car there is software embedded but also in navigation systems, audio equipment, pacemakers and toys. A half-dozen computers is installed in the car control components, from the engine to airbag systems. The allocation of income stemming from software versus the remaining product is a difficult problem (Wiederhold et al., 2009). This is further explained in section 5.3.2

2.2.3 Internal use software

Many businesses depend on internally generated software that is created in-house or made to order by a vendor. This definition of internal use software encompasses the following:

- *Commercial off-the-shelf (COTS) software:* COTS software refers to software that is purchased from a vendor and is ready for use with little or no changes (FASAB, 1998). It is an exciting software program that can be used without considerable changes (Schaefer, 2002).
- *Developed software:* The software is designed to solve a unique and specific problem (Also called proprietary software) or is a blend of off-the-shelf software and further developed (Schaefer, 2002). It can be internally developed or contractor developed:
    - *Internally developed:* Software refers to software that employees of the entity are actively developing, including new software and existing or purchased software that are being modified with or without a contractor's assistance (FASAB, 1998).
    - *Contractor-developed:* Software refers to software that an enterprise or governmental institute is paying a contractor to design, program, install, and implement, including new software and the modification of existing or purchased software (FASAB, 1998).

2.2.4 Maintenance costs

All types of software present the same unique problem in valuation, that is, that in order to ensure continued usefulness and applicability, software must be periodically updated so that it remains up to date (Wiederhold et al., 2009). Maintenance costs comprise between 60 and

80 percent of software research and development expenses in mature companies, amounting to around 15% of the prior development and maintenance costs. The effect is that software is always evolving via maintenance efforts.

The purchaser of software will only be indirectly aware of the changes that software undergoes. However, most purchasers understand that, unless they have a maintenance contract, they will have to buy a new version of the software every three to five years, since the previous version will become obsolete. Such obsolescence comes about not because the installed software has changed, but because related technology and performance expectations change. An example is the software of the enterprise Adobe. Adobe will invest $ 100 million in the next three to five years, particularly in applications based on Adobe Integrated Runtime software. With this investment Adobe wants to create a versatile foundation for capturing and holding audience attention through more active and effective applications and media (Adobe, 2011). Much of the software must be updated when business rules, accounting standards, and taxation methods change. With the changes in code come requirements for documentation updates. Note that new editions of technical books exhibit similar renewal cost software (Wiederhold et al., 2009). Maintenance costs for software is a topic to consider at the R&D spill-over valuation method (section 5.2.2). In this research the maintenance costs for software do not play a direct role for valuation methods and protection possibilities. For that reason maintenance costs are not further investigate in this research.

2.2.5 Overview application software

There are three mainstream application software categories distinguishable. Mass market software is software which is directly sold in shops and via the internet. It is easy to reproduce at low costs. The next category is embedded software, this software can be found within products like airplanes but also pacemakers. Embedded software interacts with the physical world. The last category is the internal use software which can be especially developed for a certain enterprise. This development can be internally done or by a third party.

For valuation reasons it is important to make a difference between this three categories of software because they are used in different situations. Every software product in every category have the same unique problem in valuation: Software must be periodically updated so that it remains up to date. For valuation of software, application software is taken into account as the reference software.

3. The importance of software valuation

This chapter answers research question 2: *When is it important to determine a value of software?* When it is clear in which situations valuation of software is important it is possible to get this situation as a starting point in valuation issues.

Nowadays the value of an enterprise is determined by real, tangible and intangible assets. While the importance of intangible assets in knowledge-oriented businesses is well established, legal and accounting definitions are still evolving. Traditionally the book value of an enterprise, as presented in formal terms, has mostly ignored intangible assets. Acquired intellectual property and goodwill are shown, as well as capitalized software development costs. International Financial Reporting Standards allows intangibles to be shown, but they are often omitted or poorly valued.

Intangible assets of an enterprise are all assets that are neither physical nor financial objects (Baruch, 2001). Such assets include marketing intangibles such as trademarks and trade names, as well as intellectual property such as know-how, software and trade secrets. In modern knowledge–based enterprises, these intangibles are the primary business drivers. The role of these assets is to generate income at a level that exceeds reimbursement from the labor expended, the use of commodity products, and the margins expected in routine business operations. Owners and stockholders acknowledge this fact by recognizing a market value of an enterprise as being distinct from its book value, which focuses on tangibles. In 1982, intangible assets contributed about 40 percent of enterprises' value, but at 2002, 75 percent of the market value of all the enterprises in the United States was attributable directly to intangibles, while tangible assets accounted for only the remaining 25 percent (Kamiyama et al., 2006).

3.1 Familiar with valuation

The major contribution of intangible assets to the market value of an enterprise makes it important to assign a value to, for example, software and in a broader view to intellectual property (IP). But IP valuation of technological assets is not considered as a routine within many organizations. A 2007 study performed by Micro Focus and INSEAD highlights the current state of affairs: of the 250 chief information officers (CIOs) and chief finance officers (CFOs) surveyed from enterprises in the United States, United Kingdom, France, Germany and Italy, less than 50 percent had attempted to value their intellectual property assets, and over 60 percent did not assess the real value of their software (Kwan and Stafford, 2007). Software has been termed as the "last remaining hidden corporate asset".

According to Wiederhold et al. (2009), Chandra (2005) and Chang et al. (2005) it is important to specify a value for IP in the case of a transaction. Most major transactions, such as acquisitions, long term supplier or distributor contracts, and outsourcing, involve IP but also in the case of bankruptcy. Managers have to tell their bosses the value of the project for new technology and negotiators must show the fair market value of the IP if they are to deal with a technology transfer project in an exchange market (Chang et al., 2005). Establishing a purchase price, royalty rate, or transfer price is best done on a consistent basis, rather than on a case by case assessment, often provided by outside advisors who have unknown experiences and prejudices. Similarly, when seeking financing, a solvency opinion may be required, to assure that debts will not be excessive (Wiederhold et al., 2009).

For Leloux Science & Business software valuation plays an important role in the very early state of the software product (Leloux, 2011). Researchers, enterprises and private persons are customers of Leloux Science & Business and ask valuation and protection information about their developed software product which they want to put in the market. In many situations the person or enterprise with the software idea does not want to put the related product(s) by their own in the market. In this situation it is important to determine a value for the software product because a third party will be involved.

To conclude: intangible assets of an enterprise have often a huge contribution of the total market value of that enterprise. A part of the intangible assets are software products. By transactions, and to establish the real market value of an enterprise, it is also important to have a clear view of the value of intangible assets. Also the valuation of a single software product is important for transaction between parties and cooperation agreements. For this research the focus is on software as an intangible asset. Other intangible assets are not investigated in this research.

4. Different business models

This chapter answers research question 3: *What types of business models exist in relation to software?* First a definition for a business model will be given. An overview of a business model is given by looking at the nine building blocks for a business model explained in the book 'Business model generation' written by Osterwalder and Pigneur in 2010. After that the focus in section 4.2 is on five business model patterns. These patterns are the unbundling business models, the long tail, multi-sided platforms, "free" as a business model and the open business models.

4.1 The nine building blocks of a business model

A business model can be defined as follows: 'A business model describes the rationale of how an organization creates, delivers and captures value' (Osterwalder and Pigneur, 2010). The business model is like a blueprint for a strategy to be implemented through organizational structures, processes, and systems (Osterwalder and Pigneur, 2010). The book of Osterwalder and Pigneur describes nine basic building blocks that show the logic of how an enterprise intends to make money. In this section the nine building blocks are short described white the most important characteristics per building block. The nine building blocks in total covers the four main areas of a business: customers, offer, infrastructure and financial viability.

Customer segments
Customers are very important for an enterprise, because without profitable customers an enterprise cannot survive for long. Customers can be different groups of people or organizations. Business models can be focused on mass markets, a large group of customers with broadly similar needs and problems. The opposite of mass markets are niche markets, with tailored made and specific requirements from the customers.

The customer segment of an enterprise can be segmented, diversified or focused at multi-sided markets. An example of an enterprise with the focus on multi-sided markets is a credit card enterprise. On the one side a credit card enterprise needs a large base of credit card holders and on the other side they need a large base of merchants who accept those credit cards.

Value propositions
A value proposition creates value for a customer segment. The value proposition is the reason why customers turn to one enterprise over another. Value may be quantitative like low prices and high speed of services or qualitative like design and a good customer experience. The selling enterprise must have something which has value for the buyer. This

value can be different like a low price (Ryanair in the flying business) or focused on brand/status (Rolex in the watch business).

Channels
The channels building block describes how an enterprise communicates with and reaches its customer segments to deliver a value proposition. Channels can be direct from the enterprise itself, like their own website, or indirect via partner companies. Partner channels normally lead to lower margins but they allow an organization to expand its reach and benefit from partner strengths. Communication between an enterprise and a (potential) customer has different phases. First there must be awareness of the products and services of the enterprise by the customers. Evaluation, the real purchase and the delivery process are the next channel phases. The last phase, after sales, is important for supporting customers with their purchase.

Customer relationships
The customer relationships building block describes the types of relationships an enterprise establishes with specific customer segments. This relationship can be based on human interaction or totally based on self-service. By completely self-service an enterprise maintains no direct relationship with customers. Also other variants are possible like the automated service. By looking for a product in a web shop the enterprise gives the customer also suggestions of other related products which may also be interested for the customer.

Revenue streams
Revenue streams represent the cash an enterprise generates from each customer segment. The most widely understood revenue stream derives from selling ownership rights of a physical product. Other revenue streams are fees paid for gained services, subscription fees for memberships of online access of newspaper archives or a sport institute. Also renting, licensing, asking a brokerage fee and selling advertising for example in newspapers and websites are examples of revenue streams.

Key resources
Every business model requires key resources. These resources allow an enterprise to create and offer a value proposition, reach markets, maintain relationships with customer segments and earn revenues. Key resources can be categorized in four pillars: Physical (assets such as manufacturing facilities, buildings, machines and systems), intellectual (brands, patents, copyright, partnerships and customer databases), human (people; employees) and financial (cash, credit and option).

Key activities
Key activities are the most important actions an enterprise must take to operate successfully. They are required to create an offer. Key activities can be related to production, problem solving or to a platform/network. Enterprises with problem solving as a key activities are for example consultancies and hospitals.

Key partnerships
The key partnerships building block describes the network of business partners that make the business model work. Enterprises create partnerships to optimize their business model, reduce risk or acquire resources. Four different types of partnerships can be distinguished: Strategic alliance between non-competitors, strategic partnership between competitors, joint ventures to develop new businesses and the buyer-supplier relationship to assure reliable supplies.

Cost structure
The cost structure describes all costs incurred to operate a business model. The business model can be cost-driven or value-driven, but most enterprises fall in between these two extremes. A real cost-driven enterprise is Ryanair and the opposite, a value-driven enterprise, can be a very luxury hotel like the President Wilson Hotel in Genève.

The cost structure of an enterprise consists of fixed costs and variable costs. Fixed costs are costs that do not vary with capacity planning or sales level, for example the costs for the building and salaries (Kotler and Armstrong, 1994). Naturally, fixed costs are not fixed forever but only over a predetermined time period (Ross et al., 2008). With variable costs are meant the costs that vary proportionally with the volume of goods or services to be produced. Large enterprises have economies of scale for example when they buy large quantities of raw materials. The sum of the fixed and variable costs gives the total costs.

4.1.1 A business model
The nine building blocks for a business model are describes in the above section. The building blocks are shown schematically in figure 4.1 on the next page. For every enterprise it is possible to fill the scheme in figure 4.1 with the information of this enterprise. It is normal that different enterprises give different or partly different answers within the scheme because their focus is different. Two clothing producing enterprises can have a different scheme because enterprise A is focused on quality and service while enterprise B is focused on price. The basic of every business model can be explained with the different building blocks. In general there are five different business model patterns distinguishable which will be explained in section 4.2. The combination of importance of different building blocks is the bases of the business model patterns.

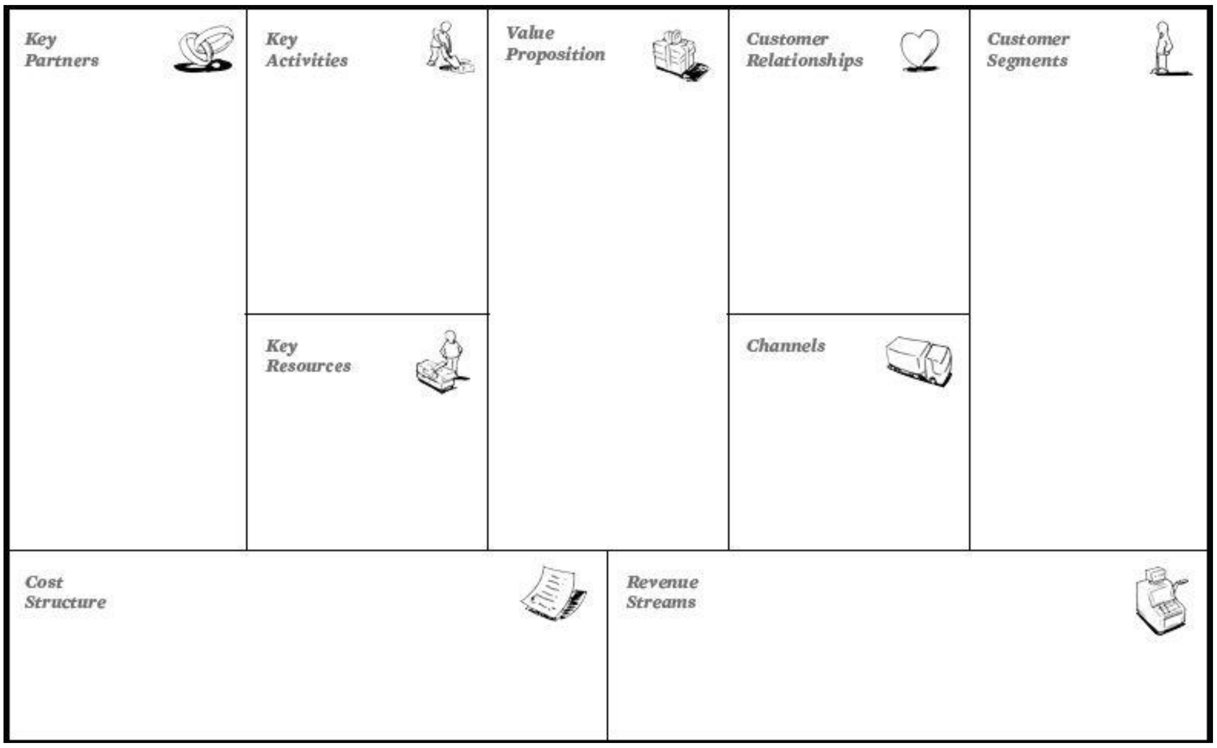

Figure 4.1: Nine building blocks of a business model (Osterwalder and Pigneur, 2010).

4.2 Business model patterns

In this section the focus is on business model patterns. According to Osterwalder and Pigneur (2010) it is possible to distinguish five different patterns. These patterns are:

- The unbundling business models and value disciplines
- The long tail
- Multi-sided platforms
- “Free” as a business model
- Open business models

The most business models fall within this five patterns but it is possible that new patterns based on other business concepts will emerge over time.

4.2.1 Unbundling business models and value disciplines

According to Osterwalder and Pigneur (2010) there are three fundamentally different types of businesses: customer relationship businesses, product innovation businesses and infrastructure businesses. Each type has different economic, competitive and cultural imperatives. For that reason it is better when an enterprise unbundles the different types of businesses into separate entities. Conflicts about the business type to follow within the enterprise can be avoided with separate entities.

The three types are comparable with the value disciplines for enterprises defined by Treacy and Wiersema (1997). The first value discipline is operational excellence. This one focuses on reliable, standard products and services at competitive, low cost, prices. The quality of the products is acceptable, but the possibilities are very limited. Second, organizations can focus on product leadership. The goal is to providing products that continually redefine the state of art with a high quality. The last value discipline is customer intimacy, according to this strategy the enterprise delivers value by creating and combining successful consumer propositions. Customer intimacy fits best with organizations offering customized products and services (Kasper, Helsdingen and Gabbott, 2006).

Operational excellence is comparable with infrastructure businesses, product leadership with product innovation businesses and customer intimacy with customer relationship businesses. The goal of unbundling business models is to choose for one type of business (or the same: one type of value disciplines) per enterprise when the focus is on more types. An example is the telecom industry, like the Dutch enterprise KPN. For years the focus of this industry was on the network and tries to become a product leader. Nowadays a lot of maintenance work of their networks is outsourced and they focus on their brand and customer relationship because that is the key asset of a current telecom enterprise (Osterwalder and Pigneur, 2010).

Unbundled business models within the software industry can exist. For example: an enterprise which makes designed software on request of a third party but also basic software packages sold for a low price in shops and via the internet. This pattern is not common seen in the software industry. For that reason this pattern is disregarded in this research.

Many enterprises, especially starting owns, focusing on one of the three value disciplines. For these enterprises unbundling the business model is not necessary because they focus on only one value discipline. The enterprise can sell their products, for example software, to another party (selling ownership rights) or selling the right to use the software for a specific time period, for example one year (selling subscriptions).

Examples

Subscriptions are asked in the software industry for example for antivirus programs like McAfee (McAfee, 2011). Selling ownership rights are not very common for software products in business to consumer trading. Between enterprises selling ownership rights take place at takeovers. In the most cases customers can buy a license for a software product, like Microsoft Office. This license can be used on a maximized number of computers. The license can be valid for years but most users wants to update their license because the software program will be outdated within approximately three to five years. The software products can directly be sold by the manufacturer or by third parties.

4.2.2 The Long Tail

Long Tail business models have a focus on offering a large number of niche products, each of which sells relatively infrequently, in low volumes (Osterwalder and Pigneur, 2010). Long Tail businesses have an opposite focus of Top businesses. In Top businesses the focus is on a small number of products (approximately the top 20 percent of products in a branch), each selling in high volumes. Aggregate sales of niche items can be as lucrative as the traditional Top model. Long Tail business models require low inventory costs, because the product is not sold frequently, and strong platforms to make niche content readily available to interested buyers. Software programs can be stored on computers, only hard disk space is needed in that case.

With the emergence of internet and falling technology costs the costs for tools of production, distribution costs and search costs are greatly reduced. For that reason a Long Tail business model can be very profitable. Software can be send by using mail or via download possibilities on websites. This makes the costs of distribution very low. Also inventory costs negligible compared with tangible products. For this reasons a Long Tail business model can be applied to software enterprises which sells a large number of software products in low volumes.

Examples

An example of a Long Tail business model can be found by toys enterprise LEGO. LEGO produce many kinds of kits around a variety of themes which can be sold in real and online shops. In 2005 LEGO started experimenting with user-generated content by introducing LEGO factory. At the enterprise site of LEGO customers can create their own kit by using the LEGO Digital Designer software and eventually sell this kit. Also other visitors can sell the kits create by other customers. For LEGO it is important that the user-designed kits expand a product line previously focused on a limited number of best-selling kits. An example of an enterprise focusing on Top businesses is IBM. The company's strategy is to focus on the high-growth, high-value segments of the IT industry (IBM, 2011).

4.2.3 Multi-sided platforms

Multi-sided platforms are platforms that bring together two or more distinct but interdependent groups of customers. Value creation is possible by connecting these groups and playing a role as intermediary. The platform creates value by facilitating interactions between the different groups. A multi-sided platform grows in value to the extent that it attracts more users. This is known as the network effect (Osterwalder and Pigneur, 2010).

Examples

In the software industry an example of a business model based on multi-sided platforms is Google`s search engine. The value proposition of Google is to provide extremely targeted text advertising globally over the web, for example by Google`s search engine. The first side of the platform consists of people who search something on the web. Probably, these people will use Google`s search engine because they know the name, they use it because of the quality or relevance of the search result or because the search engine is free to use. On the other side advertisers can publish advertisements and sponsored links on Google`s search pages. Google ensures that only relevant advertisements, with a link to the search term, are displayed. The service is attractive to advertisers because it allows them to tailor online campaigns to specific searches and particular demographic targets.

The multi-sided platform of Google will only work when enough people use the search engine and advertisers make use of Google`s platform. Without one of these parties Google`s search engine has no right to exist. For starting enterprises using multi-sided platforms as business model it can be difficult to have enough users of both sides because of the 'chicken and egg' problem. The software used for Google`s search engine plays a significant role in the business model.

Other examples of multi-sided platforms are Sony`s Playstation 3 and credit card companies. The two platforms Playstation 3 are buyers of the Playstation 3 game console and buyers of the games especially made for Playstation 3. Credit card companies must have on the one side merchants who accept the credit card and at the other hand the cardholders.

4.2.4 "Free" as a business model

In the "Free" business model at least one substantial customer segment is able to continuously benefit from a free-of-charge offer. The non-paying customers are financed by other part of the business model or by another customer segment (Osterwalder and Pigneur, 2010). The free-of-charge offer can be based on multi-sided platforms where one sided consist of advertisers.

Examples

Google`s search engine is a one-sided-free multi-sided platform because the users of the search engine do not pay anything (see also section 4.2.3). Another example is the free newspaper Metro. The two platforms of this newspaper are readers who pay nothing and on the other hand advertisers who must pay for the advertisements in the newspaper.

Other free patterns consist of free basic services with optional paid premium services, better known as 'Freemium'. The Freemium model is characterized by a large user base, benefiting from a free offer. A small portion of users, approximately 10 percent, become a paying customer. This small base of paying users subsidizes the free users.

Examples

The photo-sharing website Flickr is an example of a Freemium business model. Flickr users can subscribe for free to a basic account that enables them to upload and share images. The free service has certain constraints, such as limited storages space and a maximum number of uploads per month. For a small annual fee users can purchase a pro-account and enjoy unlimited uploads and storage space and additional features. The network site LinkedIn use also Freemium as a business model. They provide a basic free account and offer also a payable premium account with more possibilities for users.

4.2.5 Open business models

Open business models are based on the principle of open innovation. Open innovation is a principle of innovation that assumes that enterprises can and should use external ideas as well as internal ideas, and internal and external paths to market, as they look to advance their technology (Chesbrough et al., 2006). Open innovation processes combine internal and external ideas into architectures and systems. Closed innovation, the opposite of open innovation, assumes that innovation take place only within the enterprise without communicating these innovations with the enterprise's outside.

Open business models can be used by enterprises to create and capture value by systematically collaborating with outside partners (Osterwalder and Pigneur, 2010). Chesbrough distinguishes between 'outside-in' innovation and 'inside-out' innovation. Outside-in innovation occurs when an enterprise brings external ideas, technology or intellectual property into its development and commercialization processes. Inside-out innovations occurs when enterprises license or sell their intellectual property or technologies, particular unused assets.

Example

Enterprises with unused software innovations can decide to license or sell this software to other enterprises and make money with their unused software. Another example of open business models is the patent pool of GlaxoSmithKline. This pharmaceutical enterprise make use of the inside-out innovation by placing intellectual property rights into a patent pool, open to exploration by other researchers with the goal to find jointly the right drugs for understudied diseases.

5. Types of valuation methods

This chapter answers research question 4: *What types of valuation methods are available for software?* Valuation is the process of establishing a fair price for a good or service in the absence of having actual sales data (Wiederhold et al., 2009). When tangible goods are transferred from one party to another, a price for the good is usually already established, giving both parties guidance about the value of goods being transferred. For software, off-the-shelf marketable packages have similar characteristics. But a special software disk, containing software to be replicated, or especially develop software programs, cannot be valued by a unit price. Its value will largely depend on the future sales of its contents (Wiederhold et al., 2009). The contents represent the intellectual property (IP) to be valued. The cost of creating the contents, as long as it can be protected, does not determine its value. This observation is applicable to internal use software as well as marketed software products. Consider that a thousand lines of code that generate a report that nobody reads have little value, and a few brilliant lines of code can make an enterprise profitable (Wiederhold et al., 2009).

In this chapter different methods of software valuation are discussed. Also the software specific life cycle is mentioned. The start of chapter five consists of explaining general approaches.

5.1 General approaches

Valuation of software as intellectual property is more difficult than the valuation of tangible property given the intangible nature of the property itself (Chandra, 2005). Valuation methods for software assets are not as developed as valuation methods for tangible assets. Additionally, the uncertainty over the strength of protection given to a software asset influences its valuation. Also the inherent dissimilarity of software and the fact that no public trading markets exist for intellectual property assets, as they do for financial and physical assets, makes is difficult to valuate software (Chandra, 2005). Many valuation experts disagree on which method is best for valuing different types of software. Current methods for valuation of software generally fall within one of the three categories:

- *The Cost Approach* for valuing software based on how much it actually costs to create or recreate that asset and maintain the asset.
- *The Income Approach (Net Present Value)* for valuing software based on the sum of the present value of the software`s future cash flows and the initial cost of the project (by making use of a discounting rate) (Ross et al., 2008).

- *The Market Approach* for valuing software based on the licensing, patent royalty rates (if included in the software product), or sales of similar software products from one enterprise to another (Olson, 2008).

Experts say that while you may use any one of the three approaches to reliably estimate software value, it is best to compare the results obtained by using two approaches to sufficiently challenge the results and determine a likely range of value for the software (Olson, 2008).

5.2 Methods of software valuation

Naturally, valuation of software is largely based on experts' assumptions and estimations. The objectiveness of these is sometimes questioned. That is the main reason for the difficulties of conservative accounting to cope with the recognition of such assets in financial reports. However it is unrealistic to not recognize the role and governance of software assets in the functioning of enterprises in this information economy (Olson and Kolb, 2008). In this section four software valuation methods are described. The methods are:

- Direct assessment of future income
- Research and Development spill-over
- Real options valuation
- Market capitalization

The following sections under section 5.2 are mainly based on the article of Wiederhold et al., published in March 2009.

5.2.1 Direct assessment of future income

The determination of future income requires estimating the income accruing to the intellectual property (IP) in each of all future years over its useful life, i.e., the amount sold and the net income per unit after routine sales and distribution costs are subtracted. If the IP is used internally, then the savings accrued by owning the IP can be similarly estimated.

The estimation of the IP value of marketed software requires estimates of sales volumes over its life (estimates at the unit product level, as the sale price, sales and distribution overhead) and estimates that pertain to the product line, such as marketing costs, likely frequency of future version, and maintenance cost expectations over the life of the software. These estimates can be based on prior experience with the product, or on experience gained with similar products. When offshoring operations of an existing or similar product, prior data will be available and estimates will be reliable. Published information to complement internal experience is sparse, although sales trends of competitive products may be found in competitors' documentation and stockholder reports, and in research reports of industry analysts.

According to Chandra (2005) there are four parameters that need to be looked into for estimation on the future income method. They are:

1. The amount of net income the asset is expected to generate.
2. The time period over which the income is expected to be received.
3. Determination of the present value discount rate for future income.
4. The risk of realizing the future income (Chandra, 2005).

5.2.2 Research and Development spill-over

This valuation method computes the expected income by relying on the leverage of Research and Development (R&D) expenses, aggregated over multiple years. The method employs three key parameters: annual investments in R&D, the period that such an investment will contribute to future income, and the leverage ratio of R&D investments on future corporate income. Published economic benefits of R&D investments vary greatly, so that assigning such a ratio to specific R&D induces significant uncertainty. Determining the start and end of life of R&D benefits is also difficult. Early, high risk R&D investments should have a longer life than investments in short-range product alterations. R&D life values of about seven years have been cited, but these are based on an unanalyzed mix of R&D activities. The cost based R&D spill over method takes into account the physical depreciation and functional obsolescence of an asset in calculating the replacement cost and is useful in determining the maximum value of an asset to the buyer (Chandra, 2005). However, 'cost does not equal value' and the cost of an asset to the seller is irrelevant to the value of an asset to the buyer (Chandra, 2005). Also for starting software enterprises valuation based on R&D spill over is relatively unreliable. For established enterprises, where maintenance is the major component and profit margins are stable, R&D spillover can provide useful guidance.

5.2.3 Real options valuation

For software and other intangibles that have future income generating ability, and are currently yielding zero or negative returns, real options valuation is an alternative. Real options are derived from financial options and are applied to pricing investment projects. The value of an investment project depends on its investment opportunities in the future and this opportunity is an option (Chang et al., 2005). The option pricing theory of Black and Scholes is a significant contribution to asset pricing theories (Chang et al., 2005). Based on the Black and Scholes stock option valuation methodology, real options views investment in IP as an option to develop the current asset depending on the facts and circumstances at option dates. Dates to be considered would be key development, product release, and profitability milestones. This method still requires an income-based valuation, but adds the optional value of flexibility in spending or cancelling R&D costs associated with development. A drawback is the large amount of variables inherent in options pricing, leading to heightened risk of improper valuation and pricing audits, especially for options not in the public view and marketplace. The relative value of options in the overall financial picture of a corporation is hard to assess in an enterprise with a mix of activities. Divulging specific information to outsiders regarding optional plans for future expansion or cancellation of projects is very unattractive to management. Real option based valuations are hence mainly restricted to analyses performed internally or by informed and trusted experts.

5.2.4 Market capitalization

Subtracting an enterprise` book value from its total market capitalization gives a 'market worth' of the enterprise` intangible assets based on the stockholders assessment of future income. Such an estimate is already discounted for perceived risks because the stockholders influence the price. The value of intangibles that are not related to software must be subtracted as well: management expertise, the value of the workforce that cannot be protected by non-disclosure arrangements, and corporate trademarks that are not related to software.

The portion of the market value allocable to software can be substantial. For a software manufacturer, this portion obviously dominates. Consider a hypothetical case of a logistics enterprise if all proprietary internal use software for scheduling and customer interaction were to disappear. The enterprise would be out of business. Similar scenarios can be drawn for most modern businesses. For a diversified enterprise where the marketed software to be valued is only a part of the enterprise's products, a further allocation must be made. A split by sales volume becomes invalid when the products being assessed differ substantially in type and market from the items being excluded from the transfer.

This comparable market valuation method estimates the value of software assets by looking to the marketplace. According to Chandra (2005) there are four basic requirements for the market method to be functional. They are:

1. An active market must exist for the asset.
2. There must be as sufficient number of similar asset exchanges in the recent past.
3. Price information on similar asset exchanges must be available to the public.
4. The exchanges must be between independent parties (Chandra, 2005).

5.2.5 Combining valuations

If several methods produce similar results, then the valuation carries more confidence. An estimation of R&D spill-over can complement a valuation based on an estimation of future income. Essentially, R&D spill-over measures the input to the IP generation process, and income-based methods estimate the results. Market based valuations include common perceived risks, but not opportunities or risks undisclosed by corporate management. Option recognition can help when assessing differences between income-based methods and market based valuations. For the purposes of valuing an IP no single method is employed

(Chandra, 2005). In fact a combination of various valuation methods are often employed to reach a final determination of value. With different valuation methods, different perspectives on an asset`s value is provide (Chandra, 2005).

The options approach can also be used to assess the value of opportunities and risks causes by expanded off shoring arrangements, which are likely beyond expectations used in other valuation methods. There is an option value associated with maintenance expenses, since costly renewal of IP can be reduced when expectations of software lifetime are threatened by external factors, like the strength of the protection possibilities for software, as the availability of new computational paradigms or competing products.

5.3 The life cycle of software

The characteristics of the value of software over time during its life cycle in the marketplace are unique. At this point software differs crucially from other intangible and tangible assets. In general the valuation of intellectual property (IP) becomes more difficult than other tangible property because of the problem of amortization over the life of an IP (Chandra, 2005). Successful software products typically have many versions, long lifetimes, and corresponding high maintenance cost ratios over their lifetime. Software lifetimes before complete product (not version) substitution are 10 to 15 years, and are likely to increase. The frequency of release new version is usually determined by the rate of required bug fixes needed, and the tolerance of users to dealing with upgrades. Many examples have shown an average rate of months for new versions. With well-maintained software, in active use, it does not wear out, and is likely to gain value. The majority of software costs are incurred during the period after the first release to the marketplace and accepted by industry (Olson, 2009).

Accordingly, successful enterprise software products have many versions, long lifetimes, and corresponding high maintenance cost ratios over their lifetime. Maintenance costs of such enterprise software amount to 60 percent to 90 percent of total costs. These costs are primarily due to software maintenance, which refers both to the activities to preserve the software's existing functionality and performance, and activities to increase its functionality and improve its performance throughout the life-cycle. Thus, over that life, there may be 10 significant versions released. Early in its life, there may have been several versions per year. Software that is significantly dependent on external conditions will require more frequent updates (Olson, 2009).

5.3.1 Original and new code

In “What is Your Software Worth” by Gio Wiederhold (2006), it is suggested that one should measure code sizes of software versions to allocate its relative contribution to the IP. This approach assumes that the value of a unit of the original code is just as valuable as a unit of new code. It is suggests there are valid arguments that code size is not a surrogate for the contents of the IP. One argument is that later code, being more recent, represents more recent innovation, and hence should be valued higher. An argument in the opposite direction is that the basic functionality is represented by the initial code. There may have been a few lines of brilliant initial code, slowly buried in a mass of subsequent system interfaces and later tweaks and fixes, which are critical to the IP. The architectural component of software also represents valuable IP or intellectual assets and changes little over the life of the software. Wiederhold (2006) suggests one might find that much code is inserted later to deal with error conditions that were not foreseen originally. Code that was added during maintenance has its value mainly in terms of providing smooth operation and a high degree of reliability. Usually the original code provided the functionality that motivated the customer's acquisition in the first place. If that functionality had been inadequate the customer probably will move to a subsequent version. However, newer versions of the code will likely include adaptations and perfections that will motivate additional sales. Thus, given that the positives and negatives can balance each other out, it is reasonable to assign the same value to lines of old and new code. As long as the methods to obtain metrics are used consistently and without bias, the numbers obtained will be adequate for the inherently difficult objective of valuing software (Wiederhold, 2006).

5.3.2 Embedded software and tax paying

The producer and vendor of products with embedded software should give an overview of how much software is exactly used in the product for valuation reasons. The problem is that most vendors do not itemize the cost of embedded software on their invoices when they bill their clients. Many manufacturers do not want anyone, most notably competitors, to know the value of the firmware in their products. And plenty of manufacturers do not rightly know the value of the software, or greatly undervalue it (Banham, 2005).

The embedded software generally has in the United States favorable tax treatment at both the state and federal level, to the extent that a taxpayer can establish the software's value (Thomson Tax & Accounting, 2006). This favorable tax treatment comes in two forms. At the state level, software, which generally is considered as intangible property, may be exempt from tangible personal property taxes. At the federal level, enterprises can benefit from faster expensing of the capitalized software's value, which is amortized over 36 months instead of depreciated over five or seven years (the typical asset life of the equipment in which it is embedded). All the taxpayers need to know to capture these benefits of the embedded software part. But consequently, it is difficult for a taxpayer to establish the value of the embedded software and then use that valuation for tax purposes. Only when the bill is split up into a software (intangibles) part of the product and a tangibles part it is possible. Sometimes the value of a software part can be derived from different types of products in a product family. For example, the only difference between a 300 HP engine and a 350 HP engine may be an internal software switch. All physical components of the engine are otherwise identical (Thomson Tax & Accounting, 2006).

6. Protection of software

This chapter answers research question 5: *To what extent can software protection play a role in software valuation?* The different possibilities of software protection are explained. Also the role of protection in software valuation is covered.

In the knowledge economy, information has become the most important resource which enables enterprises and nations to grow. Information is the basis of competitive advantage at the micro and macro levels. As earlier mentioned is software a form of intellectual property (IP). Especially for enterprises in the science-based industries, intellectual property is a key element of the assets of the enterprise. Also for other enterprises is software an important element, for example for a wholesaler who makes use of enterprise resource planning (ERP) and inventory management systems. Without this software it is for this wholesaler very difficult, maybe even not possible, to work. Thus, it is essential that the enterprise is able to master and protect this asset. A definition for 'intellectual property' is: ideas, inventions, discoveries, symbols, images, expressive works (verbal, visual, musical, theatrical), or in short any potentially valuable human product (broadly "information") that has an existence separable form an unique physical embodiment, whether or not the product has actually been 'propertized', that is, brought under a legal regime of property rights (Landes, 2003).

Intellectual property is a collection of many different kind of things. So there are also different kinds of protection methods. Examples are copyright, database right, patent right, trademark law, related rights, domain name right, marks and designs law and trade secret. To protect software as an intellectual property it is not necessary to look at all these types of intellectual property protection because not all of these protection types have a direct link with software (Braunfeld and Wells, 2001). For that reason in this research there is looked to the following protection types:

- Copyright and watermarks
- Licensing, encryption
- Trademarks
- Software Patents
- Trade secret law

The last protection measure, trade secret law, can be used by a producer as protection measure but is not based on official rules. The other types are protection measures based on official rules.

The next sections handles the protection types of copyright and watermarks, licensing and encryption, trademarks, software patents and trade secret law.

6.1 Copyright

In reality, software copyright (and patents also) do not have a long history. Prior to 1960, there was no conflict in terms of intellectual property with regard to manipulating software, because software was not sold as an independent product without a hardware system. Copyright protection for software innovation was singled out in the United States by policymakers during the 1970s as the preferred means for protecting software-related intellectual property. In its 1979 report, the National Commission on New Technological Uses of Copyrighted Works (CONTU), charged with making recommendations to Congress regarding software protection, chose copyright as the most appropriate form of protection for computer software. Because copyright protection adheres to an author-innovator with relative ease and has a long life (now upwards of 120 years for works created for hire). The Commission determined that copyright was the preferred type of intellectual property protection for software. Congress adopted the commission`s position when it wrote: ‘computer program’ into the Copyright Act in 1980 (Cohen, 2003). Subsequently, software protection under copyright system was generally accepted worldwide. The World Trade Organization`s (WTO) Trade Related Intellectual Property Rights (TRIPs) agreement also defined computer software as literary works in Article 10.1, forming a global consensus on computer software as copyrighted objects (Suh, 2009).

The federal judiciary`s application of copyright to software in the aftermath of the CONTU initially promised strong protection for inventors. Apple Computer, Inc. is an early and important case of copyright litigation in packed software. Although the federal judiciary had long held that copyright protected only ‘expression’ in works[1], the court in Apple Computer held that Apple`s precise code was protected by copyright. The court concluded that efforts by a ‘follower’ firm to use the copyright holder`s code for purposes of achieving compatibility with the original software were inconsequential to the determination of whether infringement had occurred. This decision strengthened copyright protection considerable, making it possible for one firm`s copyrighted software to block the innovative efforts of others. Subsequent decisions extended traditional copyright protection of ‘expression’ to such ‘non-literal’ elements of software as structure, sequence, and organization. Subsequent court decisions, however, narrowed the protection provided by copyright for software-related intellectual property. The sweeping interpretation of copyright protection in Apple Computer was narrowed and weakened considerably is a series of copyright infringement cases brought by Lotus Development (Cohen, 2003).

Protection software by copyright is free of charge in the Netherlands and other countries. The copyright belongs to the producer of the software and is valid until 70 years after his or

[1] Historically, a major distinction in the copyright law has been that ideas are not protected, only expressions are. Baker v. Selden, 101 U.S. 99 (1879)

her dead. It gives worldwide protection. Databases can be protected by database right and copyright. The database right is also free of charge and protects the producer (or client) of the database for retrieve and reuse of a substantial part of the database. The protection is valid for 15 years after producing the database and is only valid when a substantial investment for producing the database is done (NL Octrooicentrum, 2011).

6.1.1 Software piracy and watermarks

The person or enterprise that created the work, had full possession of it before publication, published it, and holds the copyright to it. Enterprise A which makes the software can sell it to enterprise B. Enterprise B can work with the software, sell the complete software program or destroy it. But enterprise B is not allowed to copy the software without the authorization of enterprise A (Rub, 2011). For a software producer copyright can help at the protection of the software. By illegally use of the software the copyright holder can start a lawsuit. An example is the lawsuit of the music industry against the website of The Pirate Bay. The Pirate Bay was found guilty in the provision of copyrighted music work.

Software piracy is one of the main threats targeting software development. 35 percent of the software programs installed in 2006 are pirated (Kamel and Albluwi, 2009). With watermarking it is possible to protect the copyright of software codes and multimedia objects like images, video and audio. Watermarking means embedding digital information into original work (Kamel and Albluwi, 2009). The goal is to protect the copyright of the software after releasing it. It is assumed that the owner sells the executable and keeps the source code for himself. By hiding a message, the watermark, the owner can prove that copies of the original software version belong to him. A robust enough watermark is a watermark that required a lot of time to break into the system; cost of breaking into the system exceeds the benefits from breaking into it; and after the destroying or removing the watermark would affect the functionality or the performance of the underlying code (Kamel and Albluwi, 2009). By making use of a watermark producers of software can better protect their products for software piracy after sales.

6.2 Licensing and encryption

There are contractual alternatives to copyright protection for limiting copying. In section 6.1.1 watermarks are introduced as a tool to reduce piracy of software. In this section two other possibilities are treated. One is licensing the original work on condition that the licensee not makes copies of it or discloses it to others in a way that would enable them to make copies. Contractual prohibitions on copying may be costly to enforce and feasible only if there are few licensees. But the feasibility condition is satisfied in an important class of copyrightable expression, namely computer software programs sold directly by the manufacturer to the consumer by means of a license that forbids the consumer to make and sell copies.

Technological fixes can limit copying. Computer files (and digital files more broadly) can generally be quickly, inexpensively, and accurately copies and the copies disseminated both cheaply, sometimes indeed at zero cost, and virtually instantaneously. But encryption software can make the cost of unauthorized copying of computer files prohibitive by physically preventing the purchaser of the software product from duplicating the copy that he buys. The law can make encryption more effective. The Digital Millennium Copyright Act forbids reverse engineering of encryption devices for the purpose of facilitating unauthorized electronic copying of copyrighted recordings (Landes, 2003).

Encryption can actually provide greater protection for expressive work than copyright law does. This is not only because copyrights are often infringed, owing to costs of detection and litigation that make it impossible to achieve 100 percent compliance with copyright law, but also because encryption circumvents the defenses to copyright infringement, notably fair use – not to mention the durational limitation of copyright (Contractual restrictions in a license can do the same thing.). It is also the case, however, that by increasing the cost of sharing copyrighted works encryption can reduce their value. Thus encryption increases the copyright owner`s revenues on one margin but reduces it on another. The net effect is uncertain but, unless encryption itself is very costly, it probably is positive (Landes, 2003). Encryption is a tool to make software code secret and encrypted software cannot be illegally copied.

The encryption possibilities of the enterprise CrypKey started by 2.095 dollar for a standard encryption version until 4.695 dollar for CrypKey SDK Enterprise with complex licensing and unlimited features (CrypKey, 2011).

6.3 Trademarks

Trademarks are the distinctive signs in the form of a word, device or a label, which are used to distinguish goods or services of one undertaking from those of others. A trademark protection may include letters, numbers, pictures, symbols, colors, the name of the enterprise, sounds, smells or a combination of any of these. For software protection it may be possible to protect the symbols and colors of the logo. Also it is possible to protect the specific letter type. The companies name is also protectable with a trademark (Jain et al., 2010).

Protection something with a trademark is not always possible because of some rules. Generic trademarks cannot be protected. A common name like 'coffee' cannot be protected of a brand of a coffee enterprise because the name is a common name for a good. This rule is also applicable for the service sector. Descriptive marks also cannot be registered for protection. A descriptive mark merely describes the goods it is protecting. A suggestive trademark is not an arbitrary mark but still do not necessarily describe the product or the service associated with the actual trademark. The most suggestive trademarks describe a characteristic of that good or service. This is protectable (Goldmann, 2010).

Arbitrary trademarks are trademarks with common names but that names have not a logical association with the designated good or service. An example of an arbitrary trademark is 'Apple.' Apple is a computer enterprise so there is not a link with the eatable apples. When an apple juice manufacture will use 'Apple' as a trademark this trademark may be either to generic or descriptive. Fanciful trademarks are words or symbols with no direct connection with the goods or services. They are well protectable, an example is 'Google' for a search engine. A trademark is only protectable when there is enough difference with another existing trademark holder, so that there is no confusion between the trademarks (Goldmann, 2010).

When a trademark meets all the requirements it is possible to register the trademark in the trademark register via the Benelux-Bureau voor de Intellectuele Eigendom (BBIE). A trademark for only The Netherlands, Belgian and Luxembourg is possible for approximately 240 euro. The protection is valid for 10 years with the possibility of extending this period (NL Octrooicentrum, 2011). Trademarks which are valid within the European Union costs at least 1.600 euro and international valid trademarks are more expensive, depending on which countries are included (NL Octrooicentrum, 2011).

6.4 Software Patents

There is quite a difference between software patents in the United States and Europe. The major difference is the conditions under which you can apply for a patent. Software patents in the United States are more common than in Europe. Software patents have advantages but also disadvantage compared with copyright. More information about the difference between software patents in the United States and Europe can be found in appendix 1. This section focuses on software patents and the use of them in Europe.

By publishing software, Information and Communication Technology (ICT) enterprises release a product that is entirely made of information and can easily be copied. The traditional intellectual property protection by copyright only protected the literal expression (lines of code) against piracy[2]. Therefore, software enterprises looked for a stronger protection possibility that would also protect the idea (Band, 1995). The search for patents started in the 1970s and intensified in the 1980s and 1990s, mostly in North America, while Western Europe and Japan were still reluctant to extend patent protection to software. Finally, both Western Europe and Japan started to participate in the race to protect software through patents.

Two opposite points of view exist on the opportunity of patenting software. For authors, mostly academics, software should not be patented. The software industry, on the other hand, is in favor of software patents. The next two paragraphs give an overview of the different opinions about software patents.

The cumulative character of software development is such that each advance builds on previous achievements. If future innovators must pay royalties to the previous ones, the chances that new firms will enter the industry will decline rapidly, thus leading to a monopolistic market. The system software of Microsoft can be seen as quit monopolistic in recent years but nowadays there are more alternatives for the software of Microsoft. Furthermore, patents on software risk bringing technological change to a halt in such a dynamic industry: the life cycle of software products is around 18 months, while patents are valid for twenty years. Finally, software is now used in all industries and is vital to their development. Software patents may endanger the viability of many different manufacturing and service industries by denying access to more efficient technologies based on ICTs. Finally the national patent offices may not be equipped to grant software patents. The ensuing surge in litigation may be socially and economically costly (Hall, 2010).

The arguments in favor of software patents are also strong. Large enterprises argue that the cost of developing and marketing software is escalating to hundreds of millions of dollars for the PC software packaged market. Without patent protection these software investments would be prohibitive. Software piracy would be on the rise and affect the profitability of investments in new computer programs. The Business Software Alliance, an American association of software producers, calculates that four out of ten software programs are pirated worldwide. Also, in niche markets, smaller firms will not benefit from any license revenue if patents do not protect their investments from counterfeiters (Hall, 2010; Chabchoub, 2005).

Whatever the theoretical and public policy discussion may be, it is certain that software patents are on the rise, in North America as well as in Europe (Chachoub, 2005). Larger firms

[2] Patents and copyrights protect very different aspects of software. Copyrights are awarded to creators of "original works of authorship" and protect the specific computer code as an "original expression." Copyright does not cover the functions performed by the code. What a software program actually does may be protected by a patent (Hall, 2010).

tend to patent more often, as do firms with more products than services, and enterprises within innovative clusters tend to patent more often than those that are located in more remote regions. Also, once an enterprise has decided to patent, the size of the firm determines the number of patents it requests (Hall, 2010).

The most software patents in the United States are granted to large hardware and software enterprises. Very few small or medium-size enterprises, either independent software or computer hardware corporations, request and obtain patents. In fact, some 90% of North American software patents are held by a total of eighteen enterprises. In the period 1986-2002 a total of 22.254 software patents are registered. IBM takes a leading position in the software patents registration and is responsible for 47 percent of the total amount of registrations (Chabchoub, 2005). This finding has many implications, including implications for the debate about allowing the increase in the number of software patents (Chabchoub, 2005). For Europe there are no figures available.

Patents costs minimal 2.500 euro up to 50.000 euro. This large difference comes mainly from the number of included countries and the possibility of extending the patent. A patent is valid for 20 years but after the third year the owner must pay every year a renewal fee. After 20 years everybody is free to use the patent software technique.

6.5 Trade secret law

Software is ubiquitous in the knowledge economy and the software industry is one of the most important places where intellectual property is concentrated and wealth is created. With trade secret law it is possible to protect valuable business information from abuse by others. The main requirement is that the information must be kept a secret by its owner. Any kind of information, ranging from manufacturing know-how, formulas or devices to marketing intelligence can be protected as a trade secret. An abuser can be convicted to pay damages or to cease using the trade secret information. An example of a trade secret is the recipe of Coca-Cola. Since 1886 Coca-Cola sells its soft drink without publishing the original recipe of it (Lee, 2011).

Trade secrets are a form of intellectual property, but trade secret protection does not offer rights comparable to those offered by copyright, patents or trademarks. In most countries, trade secret misappropriation is regarded as a specific form of unfair competition. Some countries have specific laws on the protection of confidential business information. The TRIPS Agreement requires that countries implement adequate legal protection for "undisclosed information" (Engelfriet, 2007).

Any kind of information can be protected as a trade secret. Something qualifies for trade secret protection if it is kept a secret, and there is economic value because of the secrecy. A trade secret is only protected if it is in fact kept a secret by its owner. Public information cannot be protected as a trade secret, no matter how economically valuable it is. With a non-disclosure agreement (NDA), the owner of a trade secret can share information with a third party without damaging the secrecy of the information. A NDA is a contract in which it is agreed to keep certain information secret under penalty of a fine. Often a NDA is single-sided: one party supplies the information and the other must keep it a secret. A NDA can also be double-sided: both parties exchange information and agree to keep the information they receive a secret (Engelfriet, 2007).

Trade secret law is free of charge. The trade secret it is not registered anywhere (NL Octrooicentrum, 2011). It is very important to keep it secret. Everybody who knows about the trade secret must sign a non-disclosure agreement with a high fine when anybody tells something about the secret.

6.6 Protection and valuation

As stated in the last sections protection is possible in different ways. For software developers copyright is a basic protection possibility which can easily used. The disadvantage of this protection possibility is the high piracy rated of 35 percent. Also for large software enterprises like Microsoft it is difficult and costly to tackle all the individual offenders of the copyright. This is also the case for small enterprises. A solution can be a watermark, licensing or encryption protection method.

Software patents are mostly used by large and innovative enterprises. The use in Western Europe is still emerging compared with the United States. Software patents are costly compared with copyright. Patents can be used for a time of 20 years while the average life cycle of software versions is 18 months. The advantage of patents is the possibility to protect the unique idea of the software product. This is not possible with copyright.

The use of trademarks is a strong protection for names, logo`s and pictures. A total software product with different applications or a website are examples of difficult protectable items with trademarks because a trademark only protects this items when it does not change in color, letter type, etc. Trademarks are very specific protection possibilities.

Trade secret law is a protection way for software and can only be used when the owner of the software kept the software as a secret, for example within the enterprise. With a non-disclosure agreement it is possible to share the software with a third party.

For valuation reasons it is important to protect the software as good as possible. With a good protection the change of illegally copying has diminished compared with no or less protection. The most important thing an enterprise could do is to protect the intellectual property (Braunfeld and Wells, 2001). Without or with less tangible assets, protected intellectual property could ultimately determine the enterprises' valuation and even prevent a competitor from entering the same market (Braunfeld and Wells, 2001).

7. Valuation methods and business models

This chapter answers research question 6: *Which valuation methods and protection possibilities can be applied by the different business models?* In this chapter, information from previous chapters comes together. The focus is on the best way how a business model for software can be valued and protected. A scheme is used to give a better overview of the different possibilities.

7.1 Software types

In chapter 2.2 three different application software types are mentioned. These are mass marketed software, embedded software and internal use software. For valuation reasons it is important to make a difference between these three types.

For <u>mass marketed software</u> it is easier to look to comparable software products sold in the market. Based on the available information in the market a producer of mass marketed software can estimate the value of the software product. Also the value can be established by looking at the future income from sales and the R&D costs. For starting enterprises the R&D costs are not almost clear. For that reason a market approach valuation is easier to use. For the optimal valuation and protection possibilities for mass marketed software table 7.1 can be used. First, it is important to know which business model can be applied for the specific mass market software product. After that the optimal valuation and protection possibilities are given in the table.

For <u>internal use software</u>, and especially the developed software, information of comparable software products in the market are unknown because this software is particular made within or commissioned by an enterprise. So, the use of the market valuation approach is not possible for internal use software. Normally, enterprises will not share information about specific made internal use software with other enterprises. For valuation the lack of information about similar products in the market makes it more difficult to estimate future income. The future income is based on more expectation compared with mass marketed software. R&D spill-over valuation is possible when the producer of the software have enough information about the costs of the software product, and also about the further maintenance costs. For internal use software table 7.1 can be used.

<u>Embedded software</u> is difficult to valuate because producers of the product, which the embedded software included, do not give a specified overview of what software is exactly used in the product. For valuation the products as a whole can by valued on the purchase price and other general valuation approaches. Without a specified overview it is not possible to make use of favorable tax treatment for software. Only when information about the use

of software within the product is known, table 7.1 can be used to define the valuation possibilities and protection choices.

7.2 Business models, valuation and protection

With the information collected in previous chapters a picture is given about different business models, valuation methods for software and protection possibilities. In this section business models, valuation methods and protection possibilities are linked together. Leloux Science & Business can use table 7.1 to decide which valuation and protection methods are optimal in a case. Before using table 7.1 it is important to know which type of software must be valued (see section 7.1) and which business model can applied. To make it easier to decide which business model must be used, examples of enterprises using a specific business model are also given in this table. On the next page table 7.1 is showed. The text below table 7.1 gives an explanation why the specific links are made.

Table 7.1: Overview of business models linked to valuation methods and protection possibilities.

Number	Business model	Valuation method	Choice of protection	Example
1	*Based on one value discipline:*			
1a	Selling ownership rights	Market capitalization + Direct assessment of future income + R&D spill-over	Copyright + Trademarks + Encryption/watermark	Complete takeovers
1b	Selling subscriptions	Direct assessment of future income + R&D spill-over	Copyright + Database right + Trademarks + Licensing	McAfee, Microsoft Office
2a	The Long Tail / Niche products	Real options valuation + R&D spill-over	Copyright + Trademarks + Encryption/watermark	LEGO
2b	Top Businesses	Direct assessment of future income + R&D spill-over + Market capitalization	Software Patents + Trademarks + Encryption/watermark	IBM
3	Multi-sided platforms (included one-side free)	Direct assessment of future income + R&D spill-over	Copyright + Trademarks + Encryption/watermark	Google, Credit card companies, Playstation 3
4	*"Free" as a business model:*			
4a	One-side free (see 3)	-	-	Google, Metro
4b	Freemium	Direct assessment of future income + R&D spill-over	Copyright + Trademarks + Encryption/watermark + Licensing	Flickr, LinkedIn
5	Open business models	Real options valuation + R&D spill-over	Copyright + Licensing + Trademarks	GSK

In general there is no perfect valuation method per software business model. The most reliable valuation of software is possible when different valuation methods carried out and the results are combined to get a better valuation (Wiederhold et al., 2009; Chandra, 2005). When only one valuation method is used for software, for example only the direct assessment of future income, important characteristics like the maintenance costs are not taken into account. When the direct assessment of future income is combined with R&D spill-over the maintenance costs are included and the valuation is more reliable.

The R&D spill-over is a valuation method which can be used within every business model. For a producer this valuation method is understandable and practicable because of the producers' knowledge about the product. Only for starting enterprises this method can be unreliable because starting enterprises no not have always a good view on R&D costs. A combination of market capitalization, direct assessment of future income and R&D spill-over will be the most optimal combination of valuation methods for starting enterprises in the most cases. Trademarks as protection possibility should be used in every business model because it protects basically the name and logo of the product / enterprise. When the products of the enterprise are selling well the name and logo are important for goodwill. Trademarks for only the Benelux are relatively cheap (see section 6.3).

Business models based on one value discipline are different in selling ownership rights (1a in table 7.1) and selling subscriptions (1b in table 7.1). By selling ownership rights of a software product the direct assessment of future income is a good valuation method compared with R&D spill-over because R&D spill-over measures the input to the IP generation process, and income-based methods estimate the results. This combination of valuation methods are also used for the business models with the numbers 1b, 2b, 3 and 4b. Market capitalization as valuation method by selling ownership rights can be used in the case of a complete enterprise takeover and in the case of an active market with enough information about similar asset exchanges.

The protection possibilities for selling ownership rights are the free-of-charge copyright compared with encryption/watermark because this combination reduce the piracy rate. Encryption and watermarks are not free-of-charge. An enterprise must invest in encryption and watermarks. When the piracy rate is high (research shows a piracy rate of 35 percent) the investment will earn back when encryption and watermarks really reduce the piracy rate. As stated before trademarks are recommended in every business model. For selling subscriptions the free-of-charge protection possibility database right is important when the subscription fee is paid for getting access to a database with information. With a subscription fee the original software is not sold so a licensing contract gives good protection. For selling subscriptions the valuation the optimal valuation method is a combination between the direct assessment of future income and R&D spill-over because the input and output are covered with this combination.

For the Long Tail / niche products (2a in table 7.1) real options valuation is a good valuation method because niche products have future income generating ability, but can be currently yielding zero or negative returns. When niche products are generating a positive income flow direct assessment of future income is also a possible for valuation. Protection possibilities are comparable with selling ownership rights. Software enterprises which are really focused on the top businesses (2b in table 7.1) within a specific branch, and the enterprise made something very unique, then it is a good idea to look at the protection possibility of software patents. Software patents are currently only used be very large software enterprises like IBM, Microsoft, Hewlett-Packard and Sun Microsystems. It is only worth to use software patents when the earnings are higher than the cost of a patent. The combination with trademarks and encryption/watermark gives a stronger protection towards the software products. The valuation of Top businesses is comparable with selling ownership rights. Looking for valuation to the input and output side gives a more reliable valuation.

Multi-sided platforms (3 in table 7.1) and Free as a business model (4a and 4b in table 7.1) have comparable valuation methods. The direct assessment of future income and R&D spill-over valuation methods fits best because R&D spill-over measures the input to the IP generation process, and income-based methods estimate the results. Software products with multi-sided platforms as business model depend on the interest of people or enterprises from both sides. The disadvantage of this business model in starting enterprise is the ‘chicken and egg’ problem which makes valuation tough. When currently zero or negative returns are generate the real options valuation method can be also applied. For protection reasons multi-sided platforms can make use of copyright, trademarks and in some cases encryption/watermarks. Free as a business model, with Freemium as possibility, also licensing can be used as protection. Freemium consist of software with a basic free version and an advanced paid version. For using these versions of software the producer can protect these versions with licensing. When the Freemium gives access to a database also the free-of-charge database protection possibility can be used.

The last business model, based on open business (number 5 in table 7.1), can be valued by R&D spill-over. Future cash flows are less important in this business model because the goal of this model is to get free software products from the inside-out or outside-in principal. When the returns of the open business model software are zero or negative the real options valuation method can be applied for this business model. Protection of open business models are based on copyright, licensing and trademarks. In a licensing contract the producer of the software can make clear under which conditions the software can be used.

8. Summary

This research has been carried out at Leloux Science & Business in Ede (NL). To strengthen the position of Leloux Science & Business in the field of software valuation, the company has to get a better view of the different kinds of software products, the different kinds of business models in the field of software, protection possibilities for software and the way in which software can be valued in a transaction process. To give an answer on this problem, one main question and six sub-questions are defined. In this chapter an answer is given to each sub-question. The next chapter gives an answer to the main question. For Leloux Science & Business table 7.1 on page 43 is a guidance for future advises in the field of software valuation and protection.

Research question (RQ) 1: *What are the characteristics of software and what types of software exist?* This research question is focused on characteristics and types of software. Software is an intangible asset (Ross et al., 2008). Basically software is only source code which provides tasks and gives a specific interface. Software can be divided into system software and application software (Schaefer, 2002). Application software, the starting point for this research, makes use of system software for the communication with the hardware of the computer (Balci et al., 2007). The applications at the computer make it possible to use it for specific tasks.

Types of application software are mass marketed software, embedded software and internal use software. Mass marketed software is the stand alone software which is sold via the internet, in shops or as an optional package available with the purchase of hardware (Kemp, 1987). Embedded software is executed on machines that are not necessarily computers (Lee, 2002). Internal use software is generated software that is created in-house or made to order by a vendor. Developed software is an example of this type, it is designed to solve a unique and specific problem (Schaefer, 2002). All the software types present the same unique problem in valuation, that is, that in order to ensure continued usefulness and applicability, software must be periodically updated so that it remains up to date (Wiederhold et al., 2009).

RQ2: *When is it important to determine a value of software?* This research question is focused on a situation in which valuation of software is important. In modern knowledge-based enterprises intangible assets are the primary business drivers (Kamiyama et al., 2006). Intangible assets contribute for 75 percent of the market value of all enterprises in the United States and grow more and more in importance (Kamiyama et al., 2006).

According to Wiederhold et al. (2009), Chandra (2005) and Chang et al. (2005) it is important to specify a value for IP in the case of a transaction. Examples of transactions are acquisitions, long term supplier or distributor contracts and outsourcing. For Leloux Science & Business software valuation is important in the very early state of the software product, namely when the software idea or product is not yet put on the market.

RQ3: *What types of business models exist in relation to software*? This research question deals with different business model concepts. A business model describes the rationale of how an organization creates, delivers and captures value (Osterwalder and Pigneur, 2010). Business models are formed by nine basic building blocks which are Customer segments, Value propositions, Channels, Customer relationship, Revenue streams, Key resources, Key activities, Key partnerships and Cost structure. Osterwalder and Pigneur (2010) made five business model patterns by looking at the nine building blocks, important for an enterprise. The combination of importance of different building blocks is the bases of the business model patterns.

The first pattern are unbundling business models and value disciplines. The three value disciplines, operational excellence, product leadership and customer intimacy, defined by Treacy and Wiersema (1997), are the starting point for this pattern. Enterprises can sell software ownership rights or selling subscriptions. The second pattern, The Long Tail, has a focus on offering a large number of niche products, each of which sells relatively infrequently, in low volumes (Osterwalder and Pigneur, 2010). The opposite are Top businesses which have a focus on a small number of products, each selling in high volumes. The falling technology costs and the emergence of internet made selling niche products profitable.

Multi-sided platforms, the third pattern, bring together two or more distinct but interdependent groups of customers. Value creation is possible by connecting these groups and playing a role as intermediary (Osterwalder and Pigneur, 2010). An example is Google`s search engine. Also this is an example for the fourth pattern, “Free” as a business model, because Google`s search engine is free-of-charge for one side. The other side, the advertisers paid for the showed advertisements. “Free” as a business model can also consist of free basic services with optional paid premium service, the ‘Freemium’ concept. The last, fifth pattern are open business models based on the principle of open innovation. Open business models can be used by enterprises to create and capture value by systematically collaborating with outside partners (Osterwalder and Pigneur, 2010).

RQ4: *What types of valuation methods are available for software?* This research question deals with different valuation methods for software. Three general approaches can be deducted, which are the Cost approach, Income approach and Market approach. This difference is mainly the base of the more extent software valuation methods like direct assessment of future income, R&D spill-over, real options valuation and market capitalization. Each of these methods has a focus on a part of the software in an enterprise. R&D spill-over is mainly focused on annual investments in R&D, the period that such an investment will contribute to future income and the leverage ratio of R&D investments on future corporate income, while a market approach valuation is functional when information about similar exchanges in the recent past are available and an active market exist.

For the purpose of valuing an intellectual property no single method is employed as the best valuation method (Chandra, 2005). Crucial for an optimal valuation is the combination of outcomes of various valuation methods (Wiederhold et al., 2009; Chandra, 2005). Each valuation method focusing on a particular piece of the software product. With combining the outcomes of various valuation methods, the valuation is based on more insights. Wiederhold (2006) suggest valuating original and new software code as the same.

RQ5: *To what extent can software protection play a role in software valuation?* This research question deals with protection possibilities for software. Copyright is a protection possibility which can be used in almost every software case. It is free-of-charge but the piracy level, approximately 35 percent, is high. Protection by copyright in combination with encryption or watermarks gives a better protection result and lower piracy rate. For protection by encryption and watermarks an enterprise must pay. A trademark protects the name and colors of software and is a relatively cheap protection measure. This protection type is for every business recommended. Protection by patents is not commonly used. Only very large software enterprises, mainly based in the United States, are using this relatively expensive type of protection. Protection of software by patents is only recommended for very large enterprises for crucial software inventions. Trade secret law, the last investigated protection type is only recommended for little valuable software types developed and used within an enterprise and not for software which will be sold because of the difficulty to keep trade secrets secret.

RQ6: *Which valuation methods and protection possibilities can be applied by the different business models?* This research question combines information from the previous three questions. Based on that information Table 7.1 on page 43 is drawn up. In this table business models are compared with valuation methods and protection possibilities. For Leloux Science & Business table 7.1 is a guidance for future advises in the field of software valuation and protection.

9. Conclusions and recommendations

This chapter gives a critical evaluation of the research process and describes the conclusions found in this research in light of the problem statement which was defined in the introduction.

Based on the problem statement the following objective was defined in section 1.2.2: *Giving Leloux Science & Business insight in how different kinds of software products can be valued and protected, by investigating the different possibilities of software valuation and protection.* Table 7.1 on page 43 is a guidance for Leloux Science & Business for future advises in the field of software valuation and protection.

After answering the sub-question in the previous chapter the last remaining question can be answered: *To what extent can software be valued in the case of a transaction?* First of all the type of application software is important to determine before looking at valuation. The embedded software type is different to valuate compared with the mass marketed software and the internal use software types. Valuation can be done through different valuation methods. There exists no single best valuation method for software. A combination of various valuation types, like R&D spill-over compared with the income based method or market capitalization, gives the best valuation result (Wiederhold et al., 2009; Chandra, 2005). By using different valuation methods different perspectives on an asset`s value are provided. A very special characteristic of software is the life cycle. Software products versions have an average rate of months. Through maintenance the existing functionality and performance can be improved. For this improvement high maintenance costs are made. This is a point to consider by software valuation, within the R&D spill-over valuation method maintenance costs play are role in valuation. Also the possibility and effectiveness of protection plays a role in valuation. Strong protections make software more valuable. Reducing the piracy rate of software is possible to combine the copyright protection possibility with licensing, encryption or watermarks.

For protection the role of software patents is an upcoming protection possibility. When software patents are more commonly used in Europe it may be a good protection possibility for medium sized software enterprises. Software patents are a protection possibility which must be considered in the next years. Further research can be focused on the question: *Are software patents a real protection option for European enterprises?*

In this research different kinds of software types for application software are considered. The difference between application software and system software for valuation is not investigated and is a possible topic for further research. Further research can be focused on the question: *Are there different valuation possibilities for system software and application software?*

Evaluating the research process it would have been better when the transaction setting, in which the valuation takes place, was defined in a more specific way. In this research the focus on application software was very broad. A large enterprise can make a new application software product which must be valuated but also a little starting enterprise or researcher who made a software application or database and wants to know the value of the product. In this research there is not enough distinction between these two different categories of software applications. Further research can be focused on the question: *Which valuation and protection methods for software are optimal in different transaction situations?* By answering this question table 7.1 can be extended with new information.

10. Reference list

- Adobe (2011), “Adobe ventures, Investing in innovation”, *Company website*, http://www.adobe.com/aboutadobe/adobeventures, retrieved 17 August 2011.

- Bakels, R. and Hugenholtz, P.B. (2002), “The patentability of Computer Programs”, *A Report to the European Parliament.* IViR, Amsterdam.

- Balci, O., Gilley, W.S., Adams, R.J., Tunar, E. and Barnette, N.D. (2007) “Introduction to operating systems”, *Computer science, Virginia Tech,* http://courses.cs.vt.edu/csonline, retrieved 18 July 2011.

- Band, J. and Katoh, M. (1995), “Interfaces on trial, intellectual property and interoperability in the Global Software Industry”, *Westview Press.*

- Banham, R. (2005), "Surprise Inside", *CFO Magazine*, issue 1 October 2005. Available at www.cfo.com/article.cfm/4444459/c_4448927. 28 November 2010.

- Braunfeld, R. and Wells, T.O. (2001), “Protecting your most valuable asset: Intellectual property”, *IT Professional*, Volume 3, Issue 2, pages 11-17.

- Chabchoub, N. and Niosi, J. (2005), “Explaining the propensity to patent computer software”, *Technovation*, Volume 25, Issue 9, pages 971-978.

- Chandra, N. (2005), “Valuation of intellectual property rights”, *9 Corporate law cases*, Mumbia, India.

- Chang, J.R., Hung, M.W. and Tsai, F.T. (2005), “Valuation of intellectual property. A real option approach”, *Journal of Intellectual Capital*, Volume 6, Issue 3, pages 339-356.

- Chesbrough, W., Vanhaverbeke, W. and West, J. (2006), “Open innovation: researching a new paradigm”, *Oxford University Press Inc.*, New York.

- Cohen, W.M. and Merrill, S.A. (2003), “Patents in the knowledge-based economy”, *National Research Concil (U.S.).*

- CrypKey (2011), Company website, Our Products, http://www.crypkey.com/products/sdk.php, retrieved 19 August 2011.

- Dictionary of Media and Communication (2011), First Edition by Daniel Chandler and Rod Munday. *Oxford University Press Inc,* Oxford Reference Online.

- Engelfriet, A. (2007), “Legal protection of trade secrets and know-how”, http://www.iusmentis.com/innovation/tradesecrets/, retrieved 14 October 2010.

- FASAB, Federal Accounting Standards Advisory Board (1998), “Accounting for Internal Use Software”, *Statement of Federal Financial Accounting Standards,* Number 10. http://www.fasab.gov/pdffiles/fasab10.pdf, retrieved 2 December 2010.

- Goldmann, A.J.D. (2010), “Protecting intellectual property*”, Journal of internet banking and commerce*, Volume 15, Issue 1, *http://www.arraydev.com/commerce/jibc/*, retrieved 31 January 2011.

- Hall, B.H. and MacGarvie, M. (2010), “The private value of software patents”, *Research Policy*, Volume 39, Issue 7, pages 994-1009.

- Hand, J.R.M. and Baruch, L. (2003), “Intangible assets”, *Oxford University Press Inc*, New York.

- ‘t Hart, H., Boeije, H., and Hox, J. (2007), “Onderzoeksmethoden”, *Boom onderwijs,* Den Haag.

- IBM (2011), “Investor relations, Our strategy”, *company website*, http://www.ibm.com/investor/strategy/, retrieved 30 August 2011.

- Jain, S. et al (2010) “Intellectual property rights”, *J. Pharm. Sci. & Res.*, Volume 2, Issue 5, pages 314-316.

- Kamel, I. and Albluwi, Q. (2009), “A robust software watermarking for copyright protection”, *Computers & Security*, Volume 28, Issue 6, pages 395-409.

- Kamiyama, S., J. Sheehan, and C. Martinez (2006), “Valuation and Exploitation of Intellectual Property”, *STI Working Paper 2006/5, Statistical Analysis of Science, Technology and Industry*, http://www.oecd.org/sti/working-papers

- Kasper, H., Helsdingen, P. van, and Gabbott, M. (2006), “Services marketing management, A strategic perspective”, second edition, *John Wiley & Sons Ltd*, London.

- Kemp, D. (1987), “Mass Marketed software: The legality of the form license agreement”, *Louisiana law review*, Louisiana State University.

- Kotler, K. and Armstrong, G. (1994), ‘Principles of marketing’, *Prentice-Hall International, Inc.*, London, Sixth edition.

- Kwan Yuk, P. and P. Stafford (2007), “Study Urges IT Valuation Rethink”, *The Financial Times*, November 4th , 2007, http://www.ft.com.

- Landes, W.M. and Posner, R.A. (2003), “The economic structure of intellectual property law”, Harvard *University Press.*

- Lee, E.A. (2002), “Embedded software”, *Advances in computers,* Academic press, London, Volume 56.

- Lee, J. (2011), “Revealed at last: The secret recipe for Coca-Cola”, *news.co.au Business*, http://www.news.com.au/business/secret-coca-cola-ingredient-list-found-in-newspaper/story-e6frfm1i-1226006104069, retrieved 16 August 2011.

- Leloux, M. (2011), Personal communication with Leloux, M. and Gemici, T., *Leloux Science & Business office*, Ede, July and August 2011.

- Leloux Science and Business (2011), *Company website,* http://www.scienceandbusiness.nl/en/index.php, retrieved 10 July 2011.

- McAfee (2011), *Company website,* http://home.mcafee.com, retrieved 16 August 2011.

- NL Octrooicentrum (2011), “Hoe beschermt u uw idee?”, *Agentschap NL, Ministerie van Economische Zaken, Landbouw en Innovatie,* Company website, http://www.agentschapnl.nl/divisie/octrooi-merk-model, retrieved 17 August 2011.

- Nutt, G. (1997), “Operating Systems: A Modern Perspective”, *Addison-Wesley*, *Reading,* Massachusetts.

- Olson, D. (2008), “Software & Valuation in the information society”, Les *Nouvelles*, June 2008, pages 120-122.

- Olson, D. and Kolb, D. (2008), “Software & Valuation in the information society. Part two: The software inventory”, *Les Nouvelles*, December 2008, pages 272 – 279.

- Olson, D. (2009), “Software & Valuation in the information society. Part three: The software inventory valuation – TSV (OV)”, *Les Nouvelles*, June 2009, pages 130-136.

- Osterwalder, A. and Pigneur, Y. (2010), “Business model generation”, *John Wiley & Sons, Inc.*, Hoboken, New Jersey.

- Ross, S.A., Westerfield, R.W., Jaffe, J., Jordan, B.D. (2008), “Modern financial management”, *McGraw-Hill/Irwin*, New York, eighth edition.

- Rub, G.A. (2011), “Contracting around copyright: The uneasy case for unbundling of rights in creative works”, *University of Chicago law review*, Volume 78, Issue 1, pages 257-279.

- Schaefer, D. (2002) “Software: Systems and application software”, *CSU Sacramento*, Chapter 4, http://www.csus.edu/indiv/s/schaefer/presentations/ch04.pdf, retrieved 18 July 2011.

- Suh, D. and Hwang, J. (2009), “An analysis of the effect of software intellectual property rights on the performance of software firms in South Korea”, *Technovation*, Volume 30, Issue 5-6, pages 376-385.

- Thomson Tax & Accounting (2006), “Embedded software trends. Software taxation developments”, *The Journal of Taxation*, published by Thomson Tax & Accounting.

- Treacy, M. and Wiersema, F. (1995), “The discipline of market leaders”*, Perseus Books*, London.

- Verschuren, P. and Doorewaard, H. (1999), “*Designing a research project*”, *Lemma.*

- Wiederhold, G. (2006), “What is your software worth?”, *Communications of the ACM*, Volume 49, Issue 9 pages 65-75.

- Wiederhold, G. et al. (2009), “The valuation of technology-based intellectual property in offshoring decisions”, *Communications of the Association for Information Systems,* Volume 24, Issue 1, pages 523-544.

- Wilkinson, A.M. (1991), “The scientist’s handbook for writing papers and dissertations”, *Prentice Hall,* Englewood Cliffs, New Jersey.

11. Appendix

Patents for software in the United States versus Europe.

There is quite a difference between software patents in the United States and Europe. The major difference is the conditions under which you can apply for a patent. In this appendix the difference between software patents in the United States versus Europe is explained.

Under U.S. patent law "Whoever invents or discovers any new and useful process, machine, manufacture of composition of matter [...] may obtain a patent there for [...]" (35 U.S.C. § 101). Section 102 sets out the novelty requirement, whereas Section 103 clarifies that patents are granted only for non-obvious subject matter. Unlike the European Patent Convention (EPC), the U.S. Patent Act does not comprise a list of subject matter that is excluded from patentability. Also, under U.S. law there is no statutory requirement of 'technical character' (Bakels, 2002).

European patents are governed by the European Patent Convention, a treaty between all member states of the European Union and several other European countries. Co-existing with the European patent system, national patent laws in Europe offer additional opportunities for patent application at the national level. National patent laws and the European Patent Convention are similar in structure. Patents are granted for inventions on certain subject matter, if certain, substantive and formal, requirements are met.

The law usually does not define explicitly what constitutes an invention. According to Art. 52 (1) of the European Patent Convention, 'European patents shall be granted for any inventions which are susceptible of industrial application, which are new and which are not obvious.' Subsequent subsections of Art. 52 EPC list subject matter that cannot be the object of an invention. Some subject matter is inherently unpatentable, such as discoveries and scientific theories (Art. 52(2)(a) EPC). There is also subject matter that is excluded for reasons of social policy, such as medical methods in Art. 52(4) EPC. Computer programs are explicitly mentioned on the exclusion list of Subsection (2). Subsection (3) however specifies that subject matter listed in Subsection (2) is only excluded from patentability 'as such'. Chemical theories for instance cannot be patented "as such", but a chemical theory leading to a new medicine can indeed be patented in the context of a pharmaceutical patent claim. Similarly, a computer program can be patented if it is part of diagnostic equipment patent claims (Bakels, 2002).

From Art. 52 it follows, that a statutory invention must meet other requirements in order to qualify for a patent. Firstly, the invention must involve what is called an inventive step - which makes it non-obvious. The requirements for an invention to be novel are further set out in Article 54. The requirement of inventive step is elaborated in Article 56, which states that an invention must not be “obvious to a person skilled in the art”. ‘Industrial application’ is clarified in Article 57, which requires that an invention must be ‘made or used in any kind of industry’.

In addition, it is generally assumed that an invention must also be technical in order to qualify for a patent. This requirement is not mentioned explicitly in the EPC, but rather derived from European Patent Convention Rule 27, which explains that the description of the invention shall specify the technical field to which the invention relates. There is no generally agreed legal definition of the word ‘technical’ within this context. In particular, there is considerable debate to what extent computer software is to be considered ‘technical’ for the purpose of the patent law (Bakels, 2002).