

The business rationale and legal view points behind the software engineering research - a position paper on economical issues

Olli Pitkänen

Helsinki University of Technology
Laboratory of Information Processing Science
TAI Research Centre
P.O. Box 5400, FIN-02015 HUT, Finland
+358 9 451 3295
olli.pitkanen@cs.hut.fi

ABSTRACT

The paper suggests that the most severe problems in software companies are not technical but related to business and legal issues. Software companies tend to emphasize clever technical details while they forget the economical facts, the real customer needs, and the legal environment. Software engineering research would be more helpful to software business if it took into consideration also issues like economics, intellectual property rights, and commitment management.

Keywords

Software engineering research, economics, business, enterprise networking, intellectual property rights, contracts

1 INTRODUCTION

The main intent of software engineering research is to support software engineering. Most software is developed for business purposes. Normally, at least one of the stakeholders in a software project, e.g. a developer, a customer, or a subcontractor is operating in a business manner.

Especially in the start-up companies, the most severe problems seem to be involved in business and economics. The companies have technical knowledge, but many of them lack business knowledge. Thus business rationale should not be forgotten. The best way to support the innovative, new software companies is to emphasize business rationale in software engineering research.

Of course, business and economics cannot be the only motivation for scientific research. Nor should the

researchers forget the importance of more technical issues. In this paper, however, I am suggesting that business rationale and economics should not be totally ignored either. They represent significant factors that influence all areas in software engineering and thus they should be taken into consideration when focusing research work in this area.

Context and background

This paper is primarily based on my personal experience. It is only partially validated in scientific sense. Furthermore, I am not claiming that my suggestions are universal. My experience is mostly from the Finnish small and medium sized software companies. I have worked with them as a lawyer at Opplex Law Office, as a researcher at Helsinki University of Technology, and formerly also as a software engineer. I am a member of the board of one IT company and a member of the board of Finnish Software Entrepreneurs, the association of the owners and CEOs of small and medium sized software companies. I know personally a number of people in the software business. Thus, I am fairly well acquainted with the advantages and problems of Finnish small and medium sized software companies, but I do not claim anything outside that context.

2 BUSINESS VS. LEADING TECHNOLOGY

Technology enthusiasm overcomes business rationale too often. Most software companies seem to start with a bright technical innovation or an excellent idea of a technical solution to a problem. Typically they also have high knowledge in that technical area. In a word, they know how to write computer programs.

On the other hand it looks as if those start-up companies only rarely have proper understanding of the business rationale even in their own field. They quite often lack e.g. marketing and financing abilities. Furthermore, it is quite typical that the founders of a software company can not see what their software looks like as a product in the markets from the customers' point of view. Instead, they tend to see the software as a promising technical solution or a clever piece of engineering work. Most customers, however, are

not willing to pay for neat algorithms, intelligent data structures, or clever protocols. They want to buy effective solutions to their problems. Of course, technical superiority does not need to be in conflict with solving customers' problems. Certainly, they often go together. Yet, to build a successful software enterprise, the most important thing is not the bright technical detail hidden in the program, but the question whether customers are willing to pay for that software product or not.

It is not only a question of marketing the products right. To give emphasis to the customer needs even at the expense of technical advantages may have some effects on several phases of the software process. That includes requirements management, designing, implementation, and quality assurance, but also drafting the contracts. For instance, many software producers are selling several versions of their products basically because of marketing reasons. They might have for example shareware versions, public betas, or free "light" versions to get people to use their products. [6]

Because of these marketing methods, they need supportive software engineering techniques, but also license agreements, distributor agreements, and other contracts that sustain the rapidly changing variety of product versions.

Furthermore, for instance Cattaneo et al. have suggested that software companies should not focus on improving technical development without considering important dimensions such as the organization and how it relates to the company mission, its marketplace, and the human resources that are at hand. [5] Those dimensions represent more business-oriented approaches than many process improvement methods suggest. Software engineering research should take them very seriously.

3 LONG-TERM COOPERATION

It is quite widely believed that long-term success is more vital to business than taking advantage in short-term opportunities. Not all the companies have fully realized that. For instance, some large companies tend to embrace smaller companies in a way that the larger company gets all benefits from the smaller company in a short period of time, but then strangles it to death. However, it might make more sense to keep the smaller company alive and in good condition to both the companies long-term success.

Networking

Different enterprises have different advantages. For instance, a small company may be flexible, innovative, and rapid. It may also have special competence or capabilities. On the other hand, a large company has a lot of different resources, well established marketing channels, well-known brands and so on. Obviously those companies could gain from cooperation.

Malone and Laubacher have presented an exciting vision on e-lancers. They ask, if it is possible that large companies

will disappear and ad hoc networks of small companies or e-lancers will replace them. [15] It is an interesting idea, and some studies have shown, that the future trend will probably be towards smaller companies [4]. Probably the large companies will not disappear totally. They represent continuity. Some individuals need them personally, but permanence is also important for business. For example, it takes more than a random network to build a brand or to achieve real credibility.

However, many things in business can be accomplished more efficiently in enterprise networks. There seem to be a great economic demand for networks, and ways to manage them in a reasonable way.

Networking and e-lancers will change the environment for both customers and the software companies. In the future, more and more software will be both developed and used in business networks. Software engineering research should be able to bring on knowledge that software industry will need in the new environment.

4 MANAGING VALUABLES IN SOFTWARE ENGINEERING

One of the fundamental motives behind legal systems is to protect economical values. In software engineering, the valuable things are usually intangible. Thus, the legal tools to protect software engineering work and products are intellectual property rights, like copyright and patents. Those legal instruments were mainly developed long before the computer era. Therefore they do not necessarily fit well to the new forms of intellectual property. For instance, in the context of computer programs, the difference between idea and expression, the questions of novelty, non-obviousness, and usefulness have specific problems. To solve some of those problems, it may require deep understanding of both legal aspects and software engineering. [16],[18]

The legal environment is changing rapidly. For instance, in the USA, article 2B of Uniform Commercial Code is a recently drafted model law that aspires to provide a standard set of rules that will regulate transactions in information products and services. Article 2B has caused a lively conversation about the rules that govern software trading. The model law will have effects on both contract law and intellectual property law. [10],[16],[17],[20]

In European Union, several important directives have been drafted. For example, European Commission has submitted a Proposal for a directive on legal aspects of electronic commerce [7]. Also, a Proposal for a directive on the harmonization of copyright in the Information Society has been submitted [8]. Both of them would have great impact on software industry.

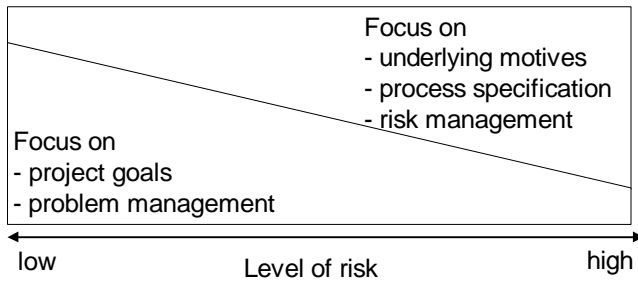


Figure 1: Commitment specification topics and risk

5 DESTRUCTIVE OR SUPPORTIVE CONTRACTS?

Too often the contracts for software engineering are written for possible future trials. A lawyer who drafts a contract knows well what is needed to win a court trial. However, a lawyer typically is not an expert on software engineering: he doesn't know what is the daily life during a software project going to be like. He is not able to draft a contract that would support the accomplishment of the project. At worst, the preparation for possible impending disagreements may even turn the stakeholders of a project against each other without any further reasons.

Commitment management

A software project is a joint undertaking by two or more participants. As a minimum, users and developers are such participants. Cooperation to develop software requires commitment from all participants: users need to commit to providing specifications and feedback, as well as financial compensation; developers must provide resources, technical skills and commitment to schedule. Often these commitments are made early in the project when there is limited information available on many of the details that ideally should be specified. We are developing an approach for defining these commitments at the beginning of development work. [12]

The current state-of-practice in industry seems to be largely unaware of the need to specify and manage commitments. Most contracts and project plans are based on straightforward application of example templates and little attention is given to what commitments in each situation should be defined. Practitioners have little more than their own experience and intuition to support the drafting of better contracts. [12]

In short, the commitment specification framework can be thought of as a series of questions that characterize the project:

- *Why* is the project being done?
- *What* is delivered and accomplished, when and for how much?
- *How* is the project done?
- *What if* something goes wrong?

The *underlying motives* explain why the project is being

done. It refers to motivation and higher-level business goals that parties have for participating in the project. [12]

Project goals define what is delivered and what is accomplished by the project, when the work takes place and finishes, and how much resources and money is spent. In other words, project goals include project output, quality goals, schedule, cost, intellectual property rights, and any other type of attribute that is associated with the project. [12]

Process specification defines how the goals should be achieved, how the parties cooperate and how the software is to be developed. [12]

The *risk and problem management* topic covers proactive responsibilities to identify and avoid potential problems, as well as defining a priori what to do if some problems occur. [12]

In most projects it is unrealistic to expect that all commitment specification topics can be defined exhaustively. Usually there is neither enough time nor information to do this. Instead, one should focus on topics that (i) are most relevant and (ii) can be specified. [12]

There are many potential situation attributes that influence what commitment specification topics should be defined. However, at an early stage in a project the overall level of risk is often the most critical situation attribute. Thus it should have the most influence on commitment specification. [12]

It may be futile to make fixed commitments to goals in a high-risk project as these risks may make goals unreachable. A better strategy in such a situation is to focus on *how* the project is done: frequent meetings, change management procedures, reporting, information exchange, etc. As Figure 1 illustrates, project with low level of risk can commit to goals more firmly. High-risk projects can have less emphasis on project goal definition and, instead, motives, process, and risk and problem management are relatively more important. [12]

6 FUTURE WORK

The problems described above do not fit nicely under any traditional science. There are both technical, legal, and business aspects to be taken into consideration. At least some of the problems require multidisciplinary research.

In software engineering research and education, the business rationale should be emphasized. Software engineers need first and foremost technical skills. However, business and legal issues may ruin software companies, if they are ignored. Thus it is important to direct the research so that all the important viewpoints are covered.

REFERENCES

- [1] B.W. Boehm and Bose P., A Collaborative Spiral Software Process Model Based on Theory W. *Proceedings of the 3rd International Conference on the Software Process*. IEEE Computer Society, Washington, DC.
- [2] B.W. Boehm and Ross R, Theory W Software Project Management: Principles and Examples *IEEE Transactions on Software Engineering*, vol. pp. 902-916, 1989.
- [3] B. W. Boehm, *Software Engineering Economics*, Prentice-Hall, 1981.
- [4] E. Brynjolfsson, T. W. Malone, V. Gurbaxani, A. Kambil. Does Information Technology Lead to Smaller Firms? *Management Science*, vol 40, No 12, December 1994.
- [5] F. Cattaneo, A. Fuggetta, and L. Lavazza. An experience in process assessment. *Proceedings of the 1995 International Conference on Software Engineering*, IEEE Computer Society, Los Alamos, 1995.
- [6] J. B. De Long, A. M. Froomkin. The Next Economy. <http://www.law.miami.edu/~froomkin/articles/newecon.htm> To be published in: Deborah Hurley, Brian Kahin, and Hal Varian, eds., *Internet Publishing and Beyond: The Economics of Digital Information and Intellectual Property*, MIT Press, Cambridge.
- [7] European Union. Proposal for a European Parliament and Council Directive on certain legal aspects of electronic commerce in the internal market. 1999/C 30/04. 1999.
- [8] European Union. Proposal for a European Parliament and Council Directive on the harmonization of certain aspects of copyright and related rights in the Information Society. 98/C 108/03. 1998.
- [9] C. Kaner. Quality Cost Analysis: Benefits and Risks. *Software QA*, Volume 3, Number 1, 1996
- [10] C. Kaner. The Law of Software Quality. *Tutorial Notes of the 10th International Software Quality Week*. Software Research Institute. San Francisco, 1997.
- [11] C. F. Kemerer. Progress, Obstacles, and Opportunities in Software Engineering Economics. *Communications of the ACM*. Vol 41, No 8, August 1998.
- [12] J. Kontio, O. Pitkänen, and R. Sulonen, Towards Better Software Projects and Contracts: Commitment Specifications in Software Development Projects *Proceedings of the 1998 International Conference on Software Engineering*, IEEE Computer Society, Los Alamos, 1998.
- [13] R. Gulati and H. Singh, The Architecture of Cooperation: Managing Coordination Costs and Appropriation Concerns in Strategic Alliances. *Administrative Science Quarterly*, Vol. 43 Issue 4, December 1998.
- [14] J. T. Landry. Supply Chain Management, The Case for Alliances. *Harvard Business Review*, November–December 1998.
- [15] T. W. Malone and R. J. Laubacher: The dawn of the e-lance economy. *Harvard Business Review*; September–October 1998
- [16] P. Samuelson, Good News and Bad News on the Intellectual Property Front. *Communications of the ACM*, vol. 42 no. 3, March 1999.
- [17] P. Samuelson, Intellectual Property and Contract Law for the Information Age. *California Law Review*, vol. 87 no. 1, January 1999.
- [18] P. Samuelson, *On Author's Rights in Cyberspace: Are New International Rules Needed?* <http://www.firstmonday.dk/issues/issue4/samuelson/>
- [19] O. T. A. Soronen. *Electronic Markets in Open Networks*. Licentiate thesis, Helsinki School of Economics and Business Administration, Helsinki 1998.
- [20] Uniform Commercial Code. Article 2b. Licenses. Draft. <http://www.law.upenn.edu/library/ulc/ulc.htm> February 1, 1999.

