

# **Idiosyncrasies of the Software Development Process and their Relation to Software Patents: Theoretical Considerations and Empirical Evidence#**

Knut Blind and Jakob Edler

**forthcoming in: netnomics 2003**

## **Abstract**

In Europe, the future patenting of software-related inventions has been the subject of intensive discussions for some time, since there exists a strong dispute between the supporters of the U.S. practice of allowing patents in order to increase Europe's competitiveness and the opponents postulating negative impacts of patents on the software development process. This paper presents empirical results about the idiosyncrasies of the software development process and tests hypotheses on their impact on the likelihood of patents being obstacles for software development. The paper concludes with the identification of determinants for preferences concerning different possible patent regimes in the future.

Corresponding author:

Dr. Knut Blind

Fraunhofer Institute

for Systems and Innovation Research

Breslauer Str. 48

D-76139 Karlsruhe

Email: kb@isi.fhg.de

Fon: 0049/721-6809-212

Fax: 0049/721-6809-260

---

# The authors acknowledge the financial support of the former German Federal Ministry for Economics and Technology (meanwhile the German Federal Ministry for Economics and Labour) and the helpful comments by an anonymous referee and by the co-editors of a special issue of netnomics. The usual disclaimer applies.

## 1. Introduction

At the European level, the future patenting of software-related inventions has been the subject of intensive discussions for some time (e.g. Commission of the European Community 2000). The member states of the European Patent Agreement have consulted with the European Patent Office about changes regarding the possibilities of patenting software-related inventions. The main question is whether software-related inventions in Europe in future should be more easily liable to patent protection – according to the model in the USA - or whether in view of assumed idiosyncrasies in the development and economic utilisation of software, patents on software should be awarded more restrictively. Background of the discussions is the fear that the lack of standard legal regulation throughout the EU could hamper the competitiveness and economic growth of the European Union. The debate is controversial. The controversy is provoked by the fact that in parts the opinion is being advanced that patents for computer software could promote innovations, as in other technology fields, for they offer the patent-holder an appropriate protection, thus creating greater incentives for further investments in the development of high-performance software. The opposite position recalls that patents, in view of the special specifics of the utilisation and development of software, would undermine fair competition (easier to form monopolies), disrupt development interactions and for this very reason could prevent innovations in the long term.

Up till now, this apparently irreconcilable confrontation has been only very partly informed by theoretically supported empirical work on the relationship of patents in the software area and that of the innovation and market behaviour. The studies available up to the present are either purely theoretical or are constructed on a narrow, rather coincidental empirical basis. Against the background of legal uncertainties, the economic controversy and the European fear of losing competitive edge to the United States in the software industry, various studies were carried out. In 1999, the European Commission commissioned a study on the economic impacts of the patentability of computer programmes. However, Hart et al. (2000) merely depicted the present legal situation in Europe, the USA and Japan, and presented economic arguments pro and contra patenting software on a qualitative level, without validating or quantifying them empirically. Lutterbeck et al. (2000) also discussed, in their expertise carried out on behalf of the German Federal Ministry for Economics and Technology, above all patent law in the economically most important regions, the legal implications of software patents for the Open Source software development and deduce herefrom recommendations for action in future patent policy. Further, the European Commission (PbT Consultants 2001), the British<sup>1</sup> and the French<sup>2</sup> governments carried out open, uncontrolled consultations via the Internet.

---

<sup>1</sup> See Webb (2001), also <http://www.patent.gov.uk/about/consultations/conclusions.htm>, access 11.04.2001.

<sup>2</sup> <http://www.internet.gouv.fr/français/textesref/avisacatec180701.htm> (07/24/2001).

The aim of this paper, on the contrary, is to present analyses of the relationship between the idiosyncrasies of the software development process and patents, based on a well-founded, broad, representative survey among the German software sector.<sup>3</sup> In a first step, the idiosyncrasies of software development and the influence of patents thereon postulated in the theoretical discussion are reviewed, because a close interdependence is presumed. The discussion concentrates on the most significant by far of these specifics, according to the literature, i.e. sequentiality, the utilisation of freely accessible code from the Open Source area and interoperability.<sup>4</sup> In the empirical part of the paper, descriptive statistics will show to what extent the software enterprises really follow this paradigm. Finally, we investigate to what degree the obstruction of the software development process by existing intellectual property rights and the attitude towards the changes in the current IPR regime, like the introduction of software patents, is actually influenced by the actual software development process. Against the background of these results, it is then possible to formulate conclusions that should be taken into account for future initiatives in the area of patenting software innovations.

## 2. Literature Survey on the Role of Patents on the Features of Software Development

Concepts are often differently defined, not only in the area of property laws, but also in the software sector. For this reason we supply our definition of the crucial terms used before the literature review. In Europe, patents are only awarded to inventions in the software area if they have a *technical character*, i.e. in simple language if they fulfil a function by means of a technical apparatus, if they are based on technical considerations, if they cause a technical effect or if they influence a physical characteristic of an apparatus. When we talk of *patents on software- and computer-related inventions*, then those inventions patentable in actual practice are meant. By contrast, patents on „software as such“ („*software patents*“) are not awarded in Europe because of the patent laws and court decisions, i.e. programme codes alone cannot be patented without their technical-functional link, i.e. only as a mathematical algorithm. Many kinds of computer programmes are therefore not patentable in Europe, unlike the USA. If we mean the patenting relevant for the software-developing actors in general, without an exact legal reference, then we talk of *patents in the software area*.

---

<sup>3</sup> The results presented in this paper are based on the results of a study conducted for the German Ministry for Economics and Technology. See Blind et al. (2001).

<sup>4</sup> Further important characteristics, which cannot be described further here, are e.g. the speed of innovation, extreme short innovation cycles, intensive interaction in development, parallel and complementary developments, uncertainty of the origin of partial inventions (code) and cheap worldwide diffusion of products via the Internet.

The transition from the industrial economy to a services- and knowledge-based economy also questions the system of industrial property rights. For in particular the patent protection which was tried and tested in the industrial age can only conditionally be transferred to services or knowledge-intensive markets. The customer-oriented development and the marketing of the knowledge-intensive, „complex product“ software exhibit a number of characteristics which make it necessary to specify the general function of industrial property rights (Smets-Solanes 2000, Besen 2001, Kash and Kingston 2001).<sup>5</sup> Therefore, the central initial presumption, on which the survey is based, is that software and its different characteristics and functions will influence the significance of property rights. The literature is also largely in agreement about the most important idiosyncrasies of software and the innovation processes in the software area: (1) sequentiality, that is the fact that new developments build very strongly on the already existing programme parts or codes, (2) the importance of the possibility to use code for free for development (Open Source) and (3) the interoperability, i.e. the necessity to make individual developments compatible with others. The following literature review will not so much discuss the existence of these three particularities, this is one central subject of the empirical investigation. It concentrates rather on the relationship of these particular characteristics to intellectual property rights, especially patents.

## 2.1 Sequentiality

A focal question for Besen and Raskind (1991) is whether intellectual property rights and especially patents represent the most efficient form of protection, also in the software area. The costs of creating new ideas depend in the software area often on the extent to which the innovators are able to have recourse to earlier work. Patent and copyright protection however tend to limit recourse to already existing ideas, by giving the creator not only the right of the own creation, but also to derivative works, so that the costs for succeeding innovators increase.<sup>6</sup> Besen and Maskin (2000) reach a similar result in their model of sequential innovation processes. They argue that an enterprise which protects its new products by patents, in the software industry with its sequential and complementary innovation processes, could prevent competitors from further developing their products. As the competitors have ideas which the original inventor did not have, for example, the patenting can brake the speed of innovation in the software industry (so also Kash and

---

<sup>5</sup> There are few authors in the literature who completely deny the particularities of the software area. One of the rare examples is Heckel (1996), who interprets all arguments against patenting in the software area as arguments against patents per se and cannot discover any specifics at all which necessitate special regulations for patents.

<sup>6</sup> In addition, it must be reckoned with the direct costs of patenting, which increase the often relatively low development costs by a considerable factor, which is a great strain cost-wise on the many small or one-man enterprises and could make their innovations unusually expensive (Smets-Solanes 2000).

Kinston 2001). Licensing would be possible, it is true, but this creates a higher competitive or price pressure which eats up the profits. Therefore the patent-holders are not interested in licensing.<sup>7</sup> Jaffe (1999) supplements this argumentation and adds as a counter argument against patent protection for software, that developers must procure the necessary licenses for the various software fragments, which causes considerable administrative and financial outlay. Large enterprises are at an advantage in such a constellation, which can attain a comprehensive patent portfolio and thus also achieve a good negotiating position for cross-licensing. The dynamics of the software industry though are carried by small software enterprises, which are correspondingly limited by excessive patenting and so can no longer guarantee the innovation dynamics of the whole sector.

In their model without patent protection, Bessen and Maskin make the basic assumption that the easy imitation of innovations reduces the profits of the innovators, but at the same time new innovations become possible, which once again open up new possibilities for follow-up innovations for the original innovating enterprise. For this reason the incentives to invest in R&D will be higher in a regime without patent protection than in a system with patent protection. The empirical evidence for the sequential innovation model, which Bessen and Maskin present, refers above all to the decrease in R&D investments and the productivity in the American software branch after the explicit introduction of software patents at the beginning of the 80s (Economist 2001). However, besides the aggregated data on the sector level, they have no enterprise information on their theoretically derived causality model.

A counter argument is quoted (Besen and Raskind 1991), that certain regulations of intellectual property rights can also lower the costs for following innovators, because the copyright registration and the patent application force the author and innovators to disclose details of their innovations. This disclosure may supply succeeding innovators with information which lower their own innovation costs; sequentiality is thus increased.

## 2.2 Open Source

The practice of patenting software which has increased in the last years has led especially in the area of the Open Source scene to opposition.<sup>8</sup> By *Open Source soft-*

---

<sup>7</sup> Ordover (1991) therefore suggests complementing a regime of strict intellectual property rights by a liberal licensing practice. Church and Ware (1998) go a step further and propose a mandatory licensing after a relatively short exclusive exploitation period. As a counter example to a functioning licensing practice Jaffe (1999) cites the semi-conductor industry, where small start-up companies have specialised, in developing new chips and have this patented for mass production, but willingly sell the necessary license to large enterprises.

<sup>8</sup> The heated debate mainly conducted via the Internet of the Open Source community about „software patents“ needs not be reproduced here in detail; see on this as an introduction e.g. the petition by Eurolinux on the exploratory paper of the European Commission

*ware* is meant that the source code is available – as a rule free of charge – (e.g. in the Internet). It offers the opportunity to develop the programme further and adapt it to own needs. According to the widely spread Gnu Public License, the further developed, altered or improved programmes – as a rule – must be made accessible free of charge and with source code. By „*Open Source Community*“ we understand the community of developers who disclose and diffuse the source codes of their developments, in order to increase the quality, security, rate of diffusion and user-orientation by means of the transparent interaction with other independent developers. Software which can be used free of charge or for a token sum (typically, less than 50 €), is called shareware or freeware. Open Source is mostly free of charge, shareware/freeware however frequently also consists of proprietary software.

Besides the protagonists of Open Source themselves, a number of scientists have also highlighted the significance of the Open Source movement for the economic and technological development in the software area, and the majority of authors see in the trend to patenting, or possibly new, patent-friendly codifications a problem for the work of numerous Open Source developers. In the following the most important arguments are brought together which are being expressed about the significance of Open Source development and the potential dangers or rather chances presented by patenting. It must be stated however that till now no broadly based investigations exist which validly quantify the economic significance of the Open Source scene.<sup>9</sup> For this reason, the importance of the Open Source scene must be estimated from its wide-spread diffusion and from the theoretically derivable economic and technological functionality.

A first indicator of the economic meaning of the Open Source-based software is simply its diffusion and the trend with which this diffusion develops.<sup>10</sup> The number of users of the operating system Linux, based on Open Source, is continually increasing world-wide<sup>11</sup>. According to statistics of the market research company IDC, 27% of all Internet pages are based on the Linux operating system<sup>12</sup>, meaning the

---

([http://petition.eurolinux.org/consultation/index\\_html?LANG=en](http://petition.eurolinux.org/consultation/index_html?LANG=en)) or the discussion forums of the German Linux community (<http://swpat.ffi.org/>).

<sup>9</sup> Nüttgens and Tesei analyse and explain the fundamental concept and the institutional structure (Nüttgens and Tesei 2000a), the production, organisation and diffusion (Nüttgens and Tesei 2000b) as well as the market models and economic rationality of networks (Nüttgens and Tesei 2000c). However, even these helpful studies supply only few empirical data on the economic significance of Open Source development.

<sup>10</sup> The development and use of Open Source products is meanwhile very differentiated; for a detailed analysis see Dempsey et al. (1999) and Nüttgens and Tesei (2000a-c).

<sup>11</sup> The quoted figures are based on chance registrations by the users themselves and the estimates of the actual total number of users fluctuate clearly <http://counter.li.org/trends.html>, accessed 23.04.2001.

<sup>12</sup> The organisers of the „Linux Counter“, an Internet page, in which Linux users can register themselves, amounted in April 2001 to more than 175,000 registrations, estimate the total number of

system is the number two in this market behind Windows (41%), whereby the growth rates of the Linux system exceed those of Windows.<sup>13</sup> World-wide, the Apache server based on Open Source occupies almost two thirds of the market for Web servers.<sup>14</sup> Also, the diffusion of Open Source programmes as well as the practice of disclosing the source codes in enterprises has greatly increased in the last few years. In Germany the share of software-developing companies, which (also) use Open Source, lies according to a survey by iX magazine at ca. 70% and thus over the diffusion in the USA (Lutterbeck et al. 2000, p. 67). Open Source development is not, as is often assumed, a domain of independent developers and small software companies, but plays a role increasingly also for large, established hardware enterprises. This is seen e.g. in the announcement by IBM to invest from 2000 till 2005 US\$ 200 m in the development of Open Source centres in Europe.<sup>15</sup>

What are the reasons for the economic benefits of the Open Source development? The economic significance of the Open Source development results basically from the fact that private or also industrial actors produce public goods and place them completely at the disposal of the public, free of charge, without property rights (Bessen 2001). This contradicts, especially in the meaning that Open Source development meanwhile possesses, generally held economic basic assumptions, according to which as a rule there is an under-supply of public goods, as their producers cannot appropriate positive externalities (free rider problem) (Arrow 1962). However, the Open Source developers are acting rationally. Their usefulness results on the one hand from the personal profile in open, transparent development forms, whereby the attractiveness for and the flexibility in the job market are increased. For the appropriated knowledge is not only visible to all, but also relatively easy to apply in other enterprises (Lerner and Tirole 2000). On the other hand, Open Source developers generate their profit frequently in novel business models no longer by selling the produced software, but with product-related services (Gross 2001).

The macroeconomic additional benefit from this diffusion of the Open Source development is based in principle on typical interactive and incremental development processes, which rather than closed developments are better able to work the complexity of software products which must provide a multiplicity of applications

---

users at up to 17 m. For Germany, a study by the firm iku-netz comes to the result that 44 % of all computers hooked up to the Internet use the Linux operating system <http://www.iku-netz.de/netscan/>, accessed 21.04.2001.

<sup>13</sup> FAZ (Frankfurter Allgemeine Zeitung) of 16.08.2001, p. 18; Computerwoche No. 10 of 09.03.2001, p. 14. This corresponds with a survey from 1999, whereby ca. 30 % of all Internet pages are based on the Linux system (Dempsey et al. 1999). The market research institute IDC assumes an annual growth rate for the Linux operating system of 25 % (<http://www.heise.de/newsticker/data/cp-03.07.01-000>, accessed 23.06.2001).

<sup>14</sup> <http://www.netcraft.com/survey/>, accessed 23.04.2001.

<sup>15</sup> <http://www.linux-community.de/Neues/story?storyid=72>, accessed 08.02.2001.

adapted for various users. Bessen (2001) speaks in reference to the inclusion of later users in the development of the superior effectiveness of „self-selection“ in the course of the Open Source development, as opposed to the central selection by software firms. In contrast to the classical products of manufacturing industry, the complexity of software and the possible variety of the applications leads to the situation that a greatly differentiated demand and utilisation arises by single adaptations and in principle can also be satisfied. A central argument is that the parallel, interactive and transparent development of software programmes by a large number of developers and testers, even possibly including the later users of the programme, leads to programmes and applications which already in the development process can be better adapted to the needs of specific groups of users than is possible in the traditional mode (Bessen 2001; Gross 2001). At the same time, these products, as a welcome side effect, are relatively free of errors, more reliable and more stable compared with products with closed code.<sup>16</sup> The synergies of parallel development and open communication lead finally to lower costs for „customizing“ and „debugging“ (so too Horns 2000, § 57ff, Lerner and Tirole 2000). Added to this individual satisfaction comes a faster diffusion through acceleration of sequential development processes, a greater variety of development results and new application possibilities (for many Smarr and Graham 2000), as well as an improved interoperability of single programmes (Murillo 1998). Finally, the transparency of Open Source programmes serves as a guarantee for the greatest possible security for software<sup>17</sup>, a postulation that was last year confirmed by a top notch committee of experts convoked by the president of the USA - among others - (Smarr and Graham 2000). In principle, these functions of Open Source are now very seldom denied (see Endres 2001).

The question is now what impacts the various forms of intellectual property rights, especially patents, have on the Open Source development and thus on the development of these postulated positive effects. Moderate representatives of the Open Source scene themselves, as well as a number of scientific authors, consider the usual patenting practice as incompatible with the Open Source development (Lutterbeck et al. 2000, Horns 2000; Gehring 2000; Smets-Solanes 2000; Smarr and Graham 2000; Bessen 2001). They fear that a stronger patenting in the area of software and computer-related inventions could limit the Open Source model, many Open Source developers even speak of the threatened end of this model.<sup>18</sup> The basic problem for the Open Source developers consists in the fact that even if they do not pursue their activities commercially in the narrower sense - and therefore a pat-

---

<sup>16</sup> Quoted from Bessen (2001), p. 5.

<sup>17</sup> According to a study of the University of Wisconsin, which compared the Open Source products with commercial products (<http://www.ccic.gov/pubs/pitac/pres-oss-11sep00.pdf>), see also Schmidt (2000). The present study does not go into the security aspect of Open Source in any detail. This aspect was the core of the study by Lutterbeck et al. (2000).

<sup>18</sup> For the discussion of the Open Source community in Germany, see <http://www.swpat.ffii.org>.

ent infringement because of commercial dealings is not actually given -, they still run the danger of infringing the patents of other parties. The reasoning is that already the „regular provision and the regular sales“ can offer ground for patent infringements (Horns 2000; § 53ff). The given complexity and sequentiality of innovations in software products make an unnoticed patent infringement more probable than for example in manufacturing industry (Dam 1995).

These basic problems are worsened by further structural problems, for Open Source developers are for the most part not attached to large enterprises and so structurally unprepared for the requirements of patenting, either by information on existing patents, or for own patent applications (Horns 2000 § 55-56; Gehring 2000; Académie des Technologies 2001). Unlike large enterprises, especially from the manufacturing sector, they lack resources, time and know-how. Established, especially large enterprises on the contrary have appropriate resources and knowledge.<sup>19</sup>

The discussed alternatives for protection rights in the area of Open Source development range from an undifferentiated inclusion of all inventions with „technical reference“ in patenting, regardless of the original model, up to a radical refusal to patent software developments generally. Beyond these extreme positions, various differentiated demands are to be found in the literature, which attempt to reconcile the significance of Open Source on the one hand and the existing framework of patenting on the other. They consist in e.g. under certain conditions to remove the Open Source codes from patent protection (Smets-Solanes 2000), with other words, to introduce a kind of „source text privilege“ (Lutterbeck et al. 2000, S. 9; see also Horns 2000, § 80).<sup>20</sup> A further suggestion is to introduce a novelty grace period, in recognition of the short innovation cycles of the software branch and the significance of Open Source for speed of the developments; this would enable the initiator of new code to communicate the same without forfeiting his claim to a patent by this action (in particular, see Gehring 2000 and Horns 2000).

Less authors see in the trend towards more patenting also a chance for the Open Source scene. Nichols (1999) e.g. assumes that stronger patenting will lead to an intensive competition for few, presumably lucrative applications and the Open

---

<sup>19</sup> Cf. e.g. an internal information brochure published by the patent department of Siemens which calls on the own software developers („Do you think that software is not patentable? Read this brochure“).

<sup>20</sup> Without wanting to have a legal discussion at this stage, we point out the problems of this demand. The construction of a source text privilege does not remove the problem of patent infringements, but postpones it. As the framework of *industrial* protection rights otherwise remains and thereby in particular industrial application would establish patent infringements, then a potential industrial user of software, which was developed on an Open Source basis and contains a code which is part of a valid patent, would run the danger of infringing a patent. This would consequently limit the attractiveness of Open Source developments which would continue to be carried out without recourse to patent searches for industrial clients.

Source scene would concentrate on the multiplicity of specific niches. For Smets-Solanes (2000), who basically emphasises the negative consequences of patents for Open Source developers, patents have on software at least the effect, that the codification, publication and thus also the diffusion of technological knowledge, as in other branches or areas also, would be improved by patent applications and publications. In this way Open Source developers can also be informed about algorithms, which are not available to them at present because of the usual secrecy practices of large software companies. A report for the European Commission acknowledges that the balance of positive and negative effects of patents in the software area is somewhat less favourable than in many other sectors, but it considers patents to be fundamentally reconcilable with the conditions of the software sector and therefore also Open Source development (Hart et al. (2000), in particular p. 31ff). The authors stress the possibility, with reference to the USA, that independent developers can also realise economic advantages through patenting (similarly, Murillo 1998, p. 198). In addition, patents on the capital market count as significant measures of the value of an enterprise and thereby possess direct economic value. Hart et al. (2000) see no empirical evidence, that the independent developers would already be hindered in the existing patenting practice in Europe by large enterprises. However they too warn of the danger, that as in the USA, too many trivial patents are awarded and thereby too many algorithms could be blocked.

As a result of this short discussion of the broad literature on Open Source and industrial property rights it can be determined that the economic and technical significance of Open Source is mainly undisputed, even if empirical data are missing. As regards the consequences of an increased patenting in the software area for the development of Open Source activities on the other hand there is no conformity. The review however showed that the problems in this area tend to be seen more strongly than in the software branch generally or even in other branches. This lies especially in the particularly open forms of communication, which many authors and developers regard as threatened by protection rights. But in the area of Open Source development, there is a dearth of empirical data on the role of Open Source software in the software area and how the developers of Open Source software are disadvantaged by patenting.

### **2.3 Interoperability**

After the observations on the role of sequentiality and the contribution of Open Source on software development and the influence of patents on both, the significant role played by network effects, the necessary standardisation and in a further step the interactions with intellectual property rights must be dealt with. For computer users profit from standardised interfaces in multiple ways (Farrell and Saloner 1985; Katz and Shapiro 1985). The greater the interoperability, or degree of standardisation, the greater is the diversity of complementary inputs (software, repairs),

which are at the user's disposal and all the easier the change from one system to another.<sup>21</sup>

Standards have two opposite consequences. On the one hand, they possibly prevent radical innovations up to a new basic standard and reduce first of all the variety of products. Standards though produce at the same time complementarity between the products fulfilling the standard, a „complementary system“ evolves. One effect of this complementarity consists in increasing the combination possibilities of products and herewith indirectly the application variety and product utility are strengthened. Because in such a system only new components and not the entire system must be newly developed, the incentive to innovate is increased and therefore also the competitive intensity and the speed of innovation. Small firms above all see advantages in standards, as they, unlike large enterprises, cannot form own enterprise-internal networks, which build on works standards.

Farrell (1989) was the first to analyse the influence of intellectual property rights on the standardisation in industries, for which compatibility is of crucial significance.<sup>22</sup> The computer and software industry belong to this group. Accordingly, compatibility aspects must be taken into especial consideration when deciding on the design of intellectual property rights. This applies not only for the extent, but also for the type of protection. Mazzeloni and Nelson (1998) argue in favour of re-thinking the present patent protection, especially for so-called system technologies, whose functioning depends largely on compatible components.

In a regime with weak protection rights, a successful innovation will rapidly attract numerous rival imitators, who will push forward on the market with similar and compatible products. This will lead to a competitive situation within the compatible and as standard accepted specification. As the users do not see themselves in a monopoly situation or in the danger of a lock-in, the standard will be further diffused and so produce further positive feedback loops. In a so-called open standard, not protected by property rights, enterprises compete on the parameters price, performance and additional product characteristics.

In a regime with strict protection rights, compatibility between the solutions of the different suppliers is made more difficult and the competition within a standard less probable. Logically, there is more competition between incompatible products, so that no common specification is crystallised as de facto standard, network effects are not possible, and the market becomes fragmented.

---

<sup>21</sup> So a frequent argument against patenting is that the common establishment of interoperability between new software developments would be crucially hampered (so too Smets-Solanés 2000)

<sup>22</sup> Similar statements can be found in Farrell (1995) and in Farrell and Katz (1998).

Thoughts on competition expressed by Katz and Shapiro (1986) concentrate on a model with two technologies. In the competition of two open, i. e. non-protected technologies for market shares, one technology will prevail as the de facto standard. Without possibilities of protection, no penetration pricing strategy is pursued, so that a product will not prevail because of costs which are higher than the utility, and less expectation regarding the market penetration (Varian and Shapiro 1999, p. 179ff).

On the contrary, in a competition between proprietary technologies, a penetration pricing strategy is possible. In the case of two proprietary standards, in the long term the better one will prevail, so that a market advantage which is bought at the cost of quality reduction does not pay off. However, in a competitive situation with one proprietary and an open standard, an inferior proprietary standard can win the upper hand against a better open standard because of the possibility of pursuing a penetration pricing strategy.

Proceeding from this theoretically based analysis, it can be derived that intellectual property rights make the pursuit of a penetration pricing strategy possible. This means on the one hand, that the suppliers of products which must first reach a critical mass, have a greater probability to achieve a successful market introduction. On the other hand, in competition the qualitatively better technology is more liable to triumph. Seen as a whole, intellectual property rights thus contribute to an increase of total welfare, seen from an economic perspective.

If network effects are present on the demand side, as is usually the case with software, then these conclusions must be modified (Farrell 1989, p. 45ff). For these effects enable the suppliers also to apply a penetration pricing strategy<sup>23</sup>, so that the additional award of intellectual property rights is of advantage for the individual enterprise, however in combination with the network effects and switching costs can very easily lead to a monopoly formation, which causes economic welfare losses not only in a static but also in a dynamic regard (Church and Ware 1998, p. 243, Jaffe 1999, p. 41).

The differing estimations of the significance of intellectual property rights in general for the innovation scene in the software industry, but also the assessment of the interaction between copyright and patent protection can be explained by two differing assumptions. Bessen and Maskin (2000) suppose in their model exclusively sequentiality and complementarity in the innovation process of software. Farrell (1989), Farrell (1995) and Farrell and Katz (1998) differentiate on the other hand between incremental innovations, on which Bessen and Maskin concentrate, and radical innovations. Consequently, they prefer basically copyright for the first type

---

<sup>23</sup> A certain cost level is estimated here for costs ensuing from a change from one system to another.

of innovation and patent protection for the last innovation type. Additionally, similar to Messerschmidt and Szyperski (2001) they proceed from strong network effects on the demand side, which Bessen and Maskin (2000) leave out of account in their model. However, this leads on the one hand to Farrell assuming a critical stance towards copyright referring to the interfaces or standard programmes used by many users, because hereby economically desirable net effects can only be exploited inadequately. On the other hand, these net effects represent barriers to radical innovations, because they limit the users' willingness to change over to a new product. For this reason Farrell simultaneously calls for patent protection for radical innovations, which enables the innovators to gain critical mass for their innovative products through intelligent pricing strategies. Without patent protection, this venture would have very little chance of success.

To sum up, in the reality of software development, not only incremental but also radical innovations are principally of elementary significance. There is, however, no empirical evidence on which type of innovation dominates, to what extent and in which manner the net effects cited by so many authors are actually to be observed in software products and programmes and which role intellectual property rights play in this interaction.

### **3. Hypotheses on the Impacts of Patents in the Software Area**

The literature review has documented that the scientific community is undecided about the relationship between software development and patenting. Whereas disagreement arose about the point of patents for all important new technologies, the literature consulted suggests that software development is indeed characterised by special idiosyncrasies which make it appear worthwhile to examine the impacts of patents in this particular area more precisely: sequentiality (i.e. incremental development processes), the process forms unique in the history of industry, of interactive, transparent development of innovations in the form of Open Source, which achieved its undisputed success without or despite patent protection as well as interoperability (i.e. manifold network effects). The picture is further complicated by the fact that these characteristics lead to different consequences in the different sub-markets of the software branches, as companies from greatly varying branches with vastly different traditions, routines and long-term interests regarding industrial property rights confront each other.

In order to survey the postulated idiosyncrasies in the area of software development and to determine the interaction with property rights, an empirical investigation was conducted. The object was first of all to determine the extent of peculiarities in the innovation process of software products, in order then to analyse their significance on patents in the software area. Point of departure for this investigation are the following hypotheses:

- H1: The innovation process in software development is characterised by great sequentiality, which manifests itself in a high rate of code re-use.
- H2: Enterprises which develop software utilise external code to a great extent, and increasingly also code from the Open Source area.
- H3: A central requirement for enterprises, which develop software, is the interoperability of their products with programmes of other origin, especially those of their customers and competitors.
- H4: Proceeding from the theoretical literature, enterprises are already affected by patents covering software innovations. The more they rely on external input in their development process and the more the interoperability of their products with other software is important, the more their software development is hindered by intellectual property rights of other companies. On the other hand, if the company relies primarily on internal development, it has little problems with the intellectual property rights of others.
- H5: The companies develop different positions on possible software patenting regimes, depending on their position regarding the three cited dimensions of the innovation process. The more pronounced the particularities are, the more problematical the extension of patents in the software area is estimated.

#### **4. The Data**

The data for testing whether the above three features of the software development process can be observed in reality and for estimating their influence on the likelihood of experiencing problems with patents in the present and their attitude towards changes of the IPR regime in the future originates from a recent survey of German software developing companies divided into two software-developing sub-branches and of independent software developers. Through the selection of a random sample in the software-developing branches, this investigation differs from the consultations running parallel, e.g. of the European Commission or Great Britain, which cannot claim representativity for the study because of the uncontrolled participation of the consultations and the differing mobilisations in this process via the Internet.<sup>24</sup> The same is true for the French approach of interviewing single representatives of important stakeholders.

---

<sup>24</sup> For results of the EU consultations, see PbT Consultants (2001), the results of the British consultations can be found under <http://www.patent.gov.uk/about/consultatons/conclusions.htm>.

A basic hypothesis of a former empirical study (cf. GFK/ISI/IESE 2000) presupposes significant differences between enterprises whose main corporate aim is to develop and sell software (primary sector), and enterprises of the so-called secondary sector, which besides a traditional main business also produce software for their „hardware“. Software, which is irrevocably integrated in hardware components and can only function together with them, such as e.g. specific control software in mechanical engineering or vehicle construction, we refer to as *embedded software*. For this reason the presentation of the descriptive statistics is differentiated according to these two groups. In addition, the primary sector is split into independent software developers and ordinary software companies.

In the survey, enterprises belong to the primary sector whose corporate objective is the development of software, according to the generally accepted classification NACE (NACE 72.202), supplemented by a number of independent software developers, because these micro-companies play a considerable role in the software area (cf. GFK/ISI/IESE 2000). The branches of the so-called secondary sector were vehicle construction (NACE 34), electro-technology (NACE 30-32), telecommunications (NACE 64) and mechanical engineering (NACE 29). The survey was conducted in May and June 2001 by approaching software companies or companies developing also software via an online questionnaire. In total 149 questionnaires from companies of the primary sector, 39 questionnaires of independent software developers and 68 questionnaires from companies of the secondary were available for the analysis. Since the companies were asked first for the general willingness to participate at the survey, before they have been requested to fill out the questionnaire, two response rates have to be reported. From the companies signalling their willingness to participate, 47% filled out the questionnaire in the primary branch and 22% in the secondary branch. Related to the number of companies approached in the first step, the response rate is significantly lower at 16% in the primary and less than 4% in the secondary branch.

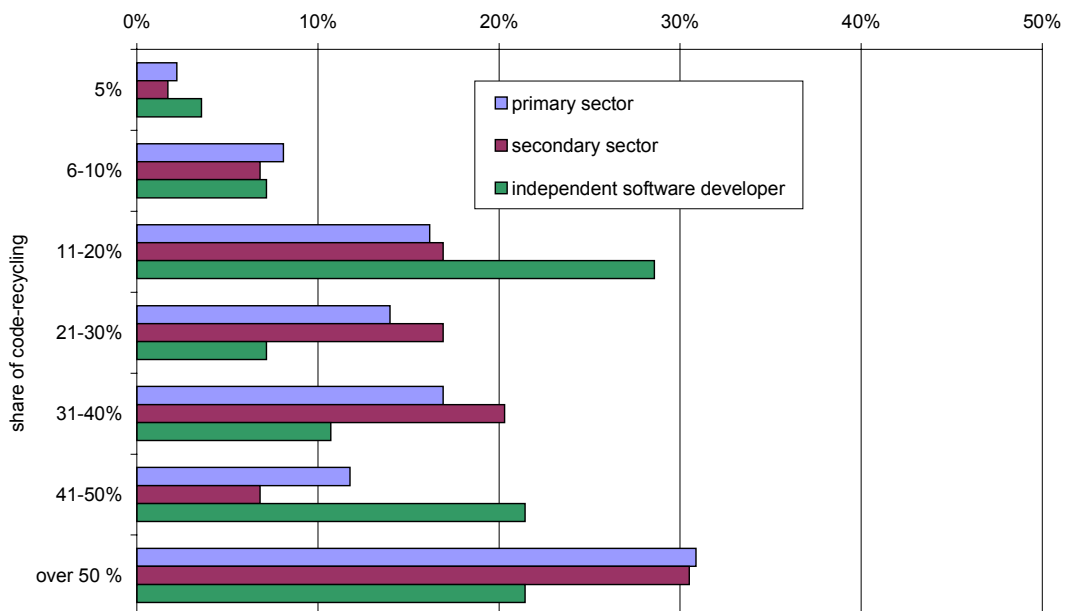
## **5. Descriptive Statistics**

A main argument of many opponents of the patenting of software is based on the fact that specific conditions predominate in the area of software- and computer-related developments, under which patenting would have contraproductive consequences. These specifics concern principally the sequentiality of software developments, the utilisation and disclosure of open code (Open Source) and the resulting own laws, such as the necessary interoperability of software products. These conditions were discussed in the literature analysis. Whether the software development process is really characterised by these three idiosyncrasies will be depicted in the following and subjected to simple hypothesis tests (H1 to H3).

## 5.1 Sequentiality

A first dimension for the idiosyncrasies of software development is the re-use of code and thus of sequentiality. One assumption by critics of patents is that patents limit the access to existing codes. An indicator for sequentiality in the software development is the degree of re-utilisation of existing code for own developments. It states the percentage of the input for new developments attributed to code re-utilisation (figure 1). The result is unambiguous: in almost a third of all enterprises, more than 50% of the input for new developments consist of already existing code. Almost two thirds of the enterprises quote a share of over 30% for code re-utilisation. The significance of sequentiality in software development is thus unanimously confirmed.

Figure 1 Share of Code Re-utilisation in Self-developed Software



In order to estimate the significance of sequentiality for the question of industrial property rights, not only the extent of code re-utilisation, but also and above all the origin of the programme components which flow into the developments of enterprises, is relevant. It appeared at first (see figure 2), that the greater share of the code that was re-used stemmed from own developments and thus was not affected by industrial property rights. This applies to both branches, is however in the secondary sector with 70% slightly higher than in the primary sector. In general, figure 1 confirms that on average around one third of software code is re-used for further developments, which indicates a certain degree of sequentiality in software development. Nevertheless, this sequentiality refers mostly on self-developed and not on code from third parties.

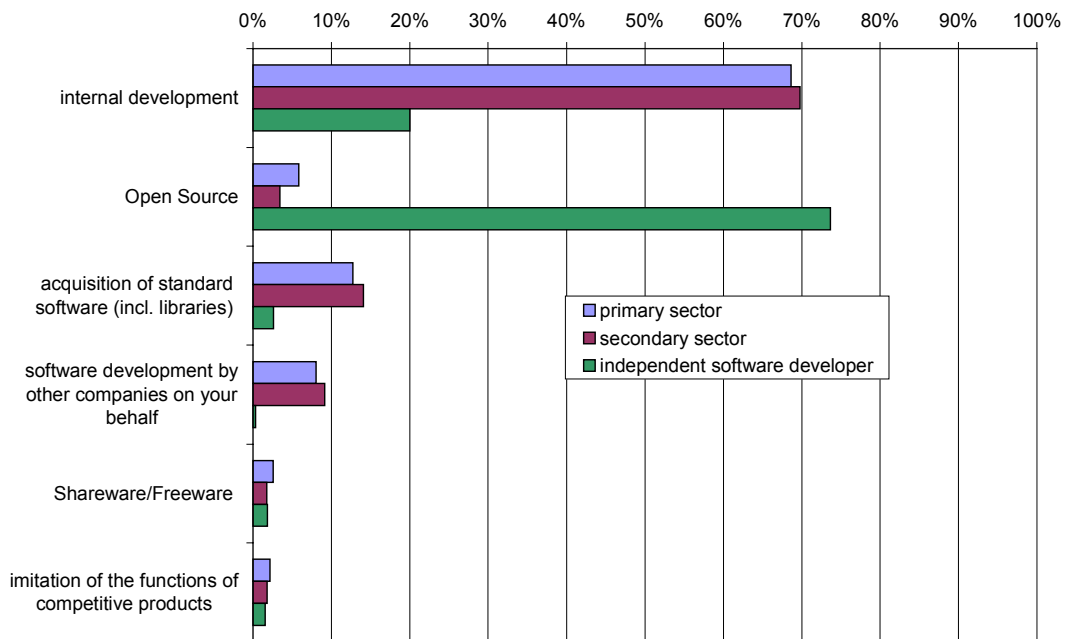
## 5.2 The Importance of Open Source Software

The result significant for the importance of industrial property rights is obtained from the analysis of the "foreign" components in own developments (figure 2). According to this analysis, Open Source software accounts for over 70% of the input independent software developers use and thus almost five that of bought standard software, and ten times as much as specially commissioned software. The imitation of foreign software plays a subordinate role, just like the use of free-of-charge software. Of all foreign components in the software developments in the primary and secondary sector, acquisition of standard software and specially commissioned software are the most significant. The great role which Open Source plays for the independent software developers cannot be observed for the primary and secondary sector, because these shares of Open Source barely reach 6%.

In the secondary sector, the extent to which Open Source is used (3.5% of the whole input for software developments) is significantly lower in comparison with the whole primary sector, while the shares of other "foreign" components are at similar levels.

The estimation of future development is important to mention. More than 60% of the enterprises in the primary sector claim that in future the importance of Open Source will increase. In the secondary sector, 70% even of the enterprises expect that the importance of Open Source will increase as an input for own development. And for the secondary sector it also applies that the dependence on external sources will increase and also the exchange of software components between enterprises. For this reason, influence on Open Source development through patenting policy in the future is of increasing importance also for the secondary sector.

Figure 2 Origin of Software in Software Products

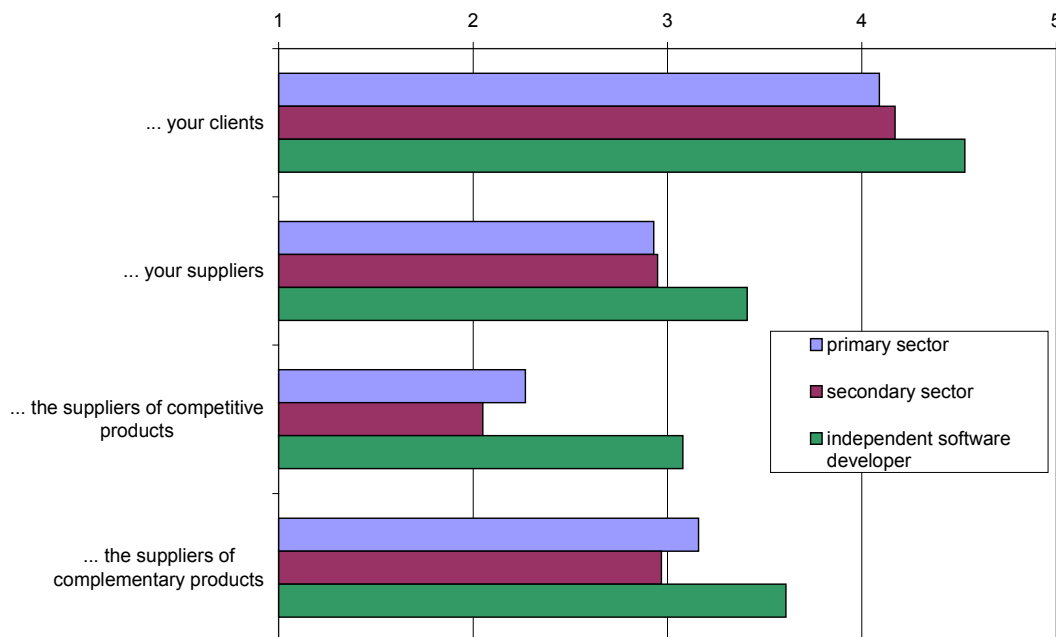


### 5.3 Interoperability

One last idiosyncrasy of software development in contrast with other products is the interoperability necessary between systems and applications or between various applications. An argument of the opponents of patenting in the software area points out that patenting could prevent the rapid establishment of interoperability, especially by means of Open Code.

Four dimensions must be differentiated regarding interoperability. Figure 3 shows that the interoperability with customer software is by far the most important criterion for both sectors. The interoperability with supplier products is of medium importance and is approximately as significant as the interoperability with products of complementary suppliers. The significantly higher figure of the primary sector in the interoperability with competitive products reveals the importance attributed to the functional compatibility of the own product with others on the market. This strategic motive on the other hand is not pronounced in the secondary sector.

Figure 3 Significance of Interoperability with Software of Various Actor Groups (1 = very low, 5 = very high)



In particular, software which is a partial component of other software is decisively dependent on interoperability. Therefore enterprises which produce software as „partial components“ attribute a much higher significance to interoperability with the software of clients, suppliers and competitors than the other enterprises.

## 6. Idiosyncrasies of the Software Development Process and their Interference with Intellectual Property Rights

The presentation of the descriptive statistics about the three idiosyncrasies of the software development process confirmed impressively the postulated first three hypotheses of chapter four. This chapter has firstly the aim to test the fourth and fifth hypotheses. Therefore, it first determines the importance of the characteristics of the software development process on the likelihood of being disturbed by intellectual property rights, especially patents by using a multivariate approach. Secondly, this multivariate approach is used to investigate the determinants of the „voting“ behaviour of the companies concerning three different changes of the IPR regime for software in the future.

The dependent variable „hindrance in software development“ is based on answers on the question whether the software development process of the companies was restrained by intellectual property rights of other companies (Model 1). The answer possibilities are recoded and divided into yes and no. Based on the posed questions,

three further differentiations can be made. In version two, a software development project has been restrained, became more expensive or has been prolonged due to the intellectual property rights of competitors (Model 2). In the more drastic version, a project has been cancelled (Model 3). Finally, a new project has even not been started because of assumed intellectual property rights (Model 4).

The descriptive statistics underline that over one quarter of the companies have experienced problems with foreign intellectual property rights which disturb their software development process. However, less than ten percent had to cancel a project. In most cases the project became more expensive or took longer than originally planned.

Table 1: Share of companies with problems caused by foreign intellectual property rights

	Share of Companies in %
Problems in general	26.32
Project extended	22.18
Project cancelled	7.89
Project not started	11.65

The explanatory variables are based on the three features of the software development process. Sequentiality is represented by the share of code-recycling multiplied by the share of input of self-developed software into the software product (SEQU). The higher this value, the less important and therefore less dangerous are intellectual property rights on the software of competitors. The second dimension discussed is the use of Open Source for the own software development process (OPSO). The higher the share of this input, the larger the likelihood of problems with patents and other property rights of third parties. Finally, interoperability is the third characteristic which has been taken into account. Since there are several independent dimensions of interoperability, of which four are covered in the survey (see figure 3), it is reasonable not to construct a single variable, but to include all four dimensions in the model. Therefore, INTCUS represents the importance of interoperability with the software of the customers, INTSUP with their suppliers, INTSUB with the suppliers of substitutive software, and INTCOMP with the suppliers of complementary software products. Concerning problems with intellectual property rights it is expected that interoperability on the vertical dimension is less a problem than on the horizontal dimension, whereas especially problems with suppliers of complementary software products are more probable.

Since a sector dummy representing the independent software developers correlates very closely with the use of Open Source, sector dummies are omitted in the regression. Furthermore, the size dimension turned out to be insignificant in the regres-

sions and did not contribute to the power of the model. Consequently, this dimension was also neglected. However, besides the three input dimensions, an additional explanatory variable is included which represents the ability of the produced software to function independently (IND). This variable is calculated based on the share of turnover realised with stand-alone software solutions and should be dichotomous to the interoperability variables.

In table 2, the results of the probit estimation are presented. A priori, no assessment was made about the quantitative impact of the three distinct features of the software development on the likelihood of problems with intellectual property rights of third parties. However, the results confirm impressively that especially companies which use significantly Open Source as input into their software development process are more likely to be restrained by foreign intellectual property rights. Coming back to the descriptive statistics about the use of Open Source, the independent software developers indicate an already high usage. Consequently, the inclusion of a sector dummy for the independent software developers makes the variable indicating the use of Open Source as input insignificant.

The expected signs of the other variables can be verified, but the coefficients are not always significant. Companies relying on own inputs and with a high share of code-recycling (SEQU) are less likely to be confronted with the inputs and consequently with the intellectual property rights of third parties. In addition, the variable indicating a high degree of independence concerning functionality of the own software from other software (IND) is at least significantly reducing the general likelihood of problems and of the likelihood of project delays.

The influence of the four dimensions of interoperability on problems with foreign intellectual property rights on software is heterogeneous. On the vertical dimension, the importance of interoperability both with the software of the customers (INTCUS) and those of their suppliers (INTSUP) reduces – although not significantly – the likelihood of problems. The close connection with those two groups fostered by regular co-operations and common projects helps to avoid IPR-related problems. However, it is worthwhile to mention the exception that a high importance of interoperability with the suppliers is likely to increase the probability that projects are not even started. This result suggests that if the suppliers of software possess patents or other protection rights, the companies are reluctant to start new software development projects which interfere with these claims.

On the horizontal level, the importance of interoperability is a driving force for the likelihood of problems with foreign IPRs. Whereas the interoperability with suppliers of competitive, potentially substitutive products (INTSUB) increases the probability only slightly and not in a significant manner, the interoperability with the software of companies producing complementary software (INTCOMP) is in all four models at least as important as the use of Open Source in explaining positively

the chance of IPR problems in the software development process. The access to complementary software and the realisation of compatibility is very often decisive for the user benefit and consequently for the competitiveness and the market success of new software products. This result is an indication of the importance of compatible interfaces between complementary software products which are often blocked by intellectual property rights. Besides the message that the use of Open Source increases the likelihood of IPR-related problems in the software development process, the dilemma already discussed in the theoretical section of intellectual property rights for interfaces between complementary software products is impressively confirmed. Both these aspects have to be borne in mind concerning the development of policy conclusions.

Table 2: Results of a Probit analysis

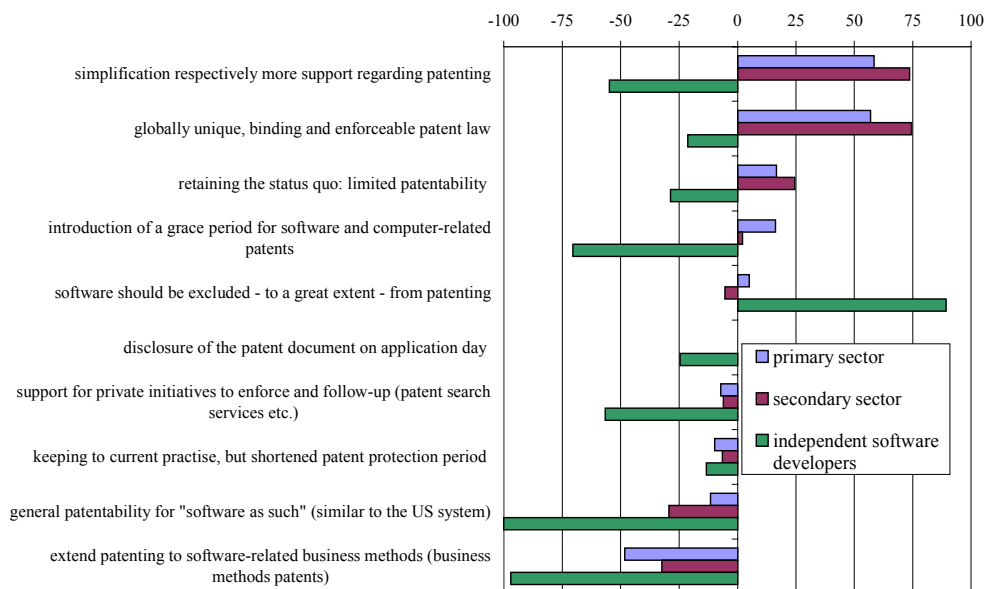
	Model 1 General problems		Model 2 Delay		Model 3 Cancellation		Model 4 Ex ante renuncia- tion	
	coeffi- cient	t-value	coeffi- cient	t-value	coeffi- cient	t-value	coeffi- cient	t-value
SEQU	-0.00036	-0.61	-0.00030	-0.48	-0.00116	-1.32	-0.00031	-0.41
OPSO	0.01191***	3.01	0.01087***	2.72	0.00426	0.92	0.01221***	2.83
INTCUS	-.00310	-0.04	0.00025	0.00	-0.00374	-0.03	-0.02337	-0.22
INTSUP	-0.058350	-1.01	-0.09267	-1.55	-0.04723	-0.60	0.05922	0.84
INTSUB	-.017210	-0.20	0.07835	0.91	0.05486	0.52	0.04436	0.44
INTCOMP	0.23704***	3.08	0.14344*	1.83	0.25787**	2.24	0.18369*	1.90
IND	-0.00567**	-2.31	-0.00468*	-1.88	-0.00151	-0.42	0.00096	0.30
Constant	-0.97893**	-2.26	-1.01342**	-2.31	-1.98717***	-3.07	-2.22698***	-3.61
PseudoR2	0.1387		0.1221		0.1347		0.1670	
Log Likeli- hood	-132.03888		-123.57276		-63.572559		-79.759651	
Number of Observations	266		266		266		266	

Finally, we test hypothesis 5, thus returning to the discussion of a change in the IPR policy regime with which the paper began. The aim is to determine which features of the current software development process are important for the attitude of the companies towards different options to change the IPR regime concerning software in Germany. In figure 4, the votes of the addressed companies on different policy options are illustrated. The scale reaches from  $-100$ , representing total opposition, to  $+100$ , representing total agreement. Results around zero signal a strong antagonism between two equally strong groups. Ambivalent votes are neglected. In the following discussion, we leave out the attitudes towards different supportive measures, like simplification of application procedures, and small changes of the existing regime (see figure 4).

In the literature, several authors (Thurow 1997, Murillo 1998, Stolpe 2000, Nalley 2000, Smets-Solanes 2000, Holmes 2000) postulate a technology- and market-specific differentiation of intellectual property rights, which would represent a so-called „sui generis“ solution. Oz (1998) finds in his survey among patent attorneys and companies also a broad agreement for such an IPR regime. Moderate deviations from the current patenting practice request a privilege for source texts (Lutterbeck et al. 2000, Horns 2000) or specific grace periods in connection with obligatory licenses (Gehring 2000). Other authors disagree completely with „sui generis“ solution for software (cf. Dam 1995) or point to the juridical problems of such a comprehensive differentiation, like Horns (2000).

In general, the primary and secondary sectors tend to have a sceptical attitude towards a further increase of patenting in the software sector, whereby the attitude of the independent developers is decisively against an extension in the direction of the US model. The independent developers are even in favour of a general exclusion of software from patenting, the other enterprises of the primary and secondary sectors are ambivalent on this option. They are slightly in favour of the status quo, which is also denied by the independent software developers.

Figure 4: Attitudes towards Different Options to change the Current IPR Regime concerning Software Patenting



In order to determine the most important motives behind the “voting” behaviour, we duplicated the probit analysis above by substituting the dependent variable by the agreeing votes and non-agreeing votes concerning the three general options of a change in the IPR software regime: status quo, restriction of patenting, extension of

patenting.<sup>25</sup> Table 3 presents just the share of agreeing votes differentiated by the three options. The disagreement about the preferences for an IPR regime covering software is obvious. No alternative finds a majority. The option to reverse the recent practice and to restrict software patenting reaches at least almost 40 percent, whereas an extension finds only the support of around one fifth of the respondents. The satisfaction with the status quo is also not very high with less than one third of the votes in support.

Table 3: Share of Companies agreeing to Three Different IPR Regimes of Intellectual Property Rights

	Share of Agreeing Companies in %
Status quo	31.95
Restriction of software patenting	39.85
Extension of software patenting	20.30

In addition to the already defined independent variables characterising the software development process, the experience with the current patent regime (PATEX), measured by the use of patents as protection means, is considered as an independent variable since it is essential for the attitude towards the status quo, but also towards restrictions or extensions of the patentability of software. The greater the familiarity with the existing institutional settings, the less likely a vote for a restriction of the already available options becomes and more support for a further extension of the current possibilities can be expected.

Table 4 presents the results of the probit analysis of the voting behaviour for the three selected options of a possible IPR regime for software. The model to explain the voting behaviour concerning the status quo is not satisfactory. Except for the disagreement of the Open Source users with the existing regime, other variables not included in the model must determine the voting behaviour. However, the models to explain the voting behaviour concerning both the option to restrict patenting and the possibility to extend software patenting deliver more significant and convincing results. The Open Source users are definitely in favour of a restriction of software patenting and reject totally an extension of the patentability of software. The respondents already using patents are not willing to lose this strategic option and support an extension of software patentability such as already realised in the United States. In addition, those companies with a high share of turnover with independent software not embedded in other soft- or hardware definitely support a restriction of software patenting. They are obviously afraid of being restricted in their development processes by an invasion of software patents. Whereas under the current regime companies with a high priority for interoperability with the software of suppli-

---

<sup>25</sup> Answers indicating an ambivalent attitude towards the proposed option are counted as non-agreement.

ers of complementary software are more likely to run into problems with intellectual property rights during their development process, neither this dimension nor the other dimensions of interoperability are able to explain the voting behaviour of the companies. Surprisingly, the tendency of companies which assign a high importance to interoperability is to support an extension of software patenting. In general, the analysis of the voting behaviour brought up more puzzling results compared to the analysis of the determinants of the likelihood of current problems with already existing intellectual property rights in the software development process.

Table 4: Results of a Probit Analysis

	Status quo		Restricting software patenting		Extending software patenting	
	coefficient	t-value	coefficient	t-value	coefficient	t-value
SEQU	-0.00828**	-1.97	0.02061***	3.65	-0.044670***	-3.03
OPSO	-0.00056	-1.07	-0.00083	-1.52	-0.00010	-0.17
INTCUS	-0.06697	-0.93	0.04901	0.66	0.08489	0.91
INTSUP	-0.03134	-0.57	0.02916	0.49	0.08606	1.32
INTSUB	-0.00548	-0.07	0.10471	1.22	0.01825	0.20
INTCOMP	0.07356	1.07	-0.00047	-0.01	0.03870	0.51
IND	-0.00085	-0.38	0.00498**	2.08	0.00210	0.83
PATEX	0.11475	0.55	-0.55738**	-2.54	0.45021**	2.07
Constant	-0.00763	-0.02	-0.93930**	-2.59	-1.49922***	-3.19
PseudoR2	0.0226		0.1853		0.1370	
Log Likelihood	-162.89766		-145.71566		-115.82227	
Number of Observations	266		266		266	

## 7. Conclusion

The empirical survey among German software companies impressively confirmed the existence of idiosyncrasies in software development. Both sequentiality, the actual use and the increasing importance of Open Source and the necessity to ensure interoperability in the vertical and horizontal dimension are essential features of software development processes. However, these three factors influence to a different degree both the likelihood of problems with foreign intellectual property rights and the priorities of respective different IPR regimes concerning software patents. Whereas sequentiality is not significant in explaining the likelihood of IPR problems during software development and the votes for different IPR regimes, the intensity of Open Source is the decisive independent variable in all models. In addition, the interoperability of the own software products with complementary soft-

ware is already at present seriously hampered by intellectual property rights, as postulated in the literature (cf. Farrell 1995).

From these results it becomes clear that future IPR regimes in the software sector have to take into account the impact of their institutional arrangements on the use and development of Open Source. Whereas the negative impacts may be negligible in the short run, in the long run the progress of Open Source as a kind of public good, which promotes innovation dynamics in the sense of Romer's endogenous growth theory (1990) especially in the so-called New Economy, will be seriously injured. The more so since Open Source will not remain a mode of operation for independent developers or certain niche companies, but rather will become a key feature for larger enterprises both in the primary and secondary sector. Therefore, Open Source needs some sort of protection against intellectual property rights. How this kind of protection can be realised effectively and efficiently, however, represents a challenge for further interdisciplinary research in the future, integrating legal, economic and technical aspects. One solution to save the future of Open Source and solve the detected interoperability problem may be the introduction of a reward system, under which innovators are paid for innovations directly by the government and innovations pass immediately into public domain<sup>26</sup>, since obligatory licensing may obstruct the incentives of innovators or lead to other even more destructive protection strategies.

---

<sup>26</sup> See Shavell and Ypersele (2001) for a general comparison between a reward system and intellectual property rights.

## References

- Académie des Technologies (2001): Avis de l'Académie des Technologies concernant la brevetabilité des inventions mises en oeuvre par ordinateur; <http://www.internet.gouv.fr/francais/textesref/avisacatec180701.htm> (24.07.2001)
- Arrow, K. J. (1962): Economic Welfare and the Allocation of Resources for Invention, in: National Bureau of Economic Research (Ed.): The Rate and Direction of Inventive Activity: Economic and Social Factors, Princeton: Princeton University Press.
- Besen, S. M. and Raskind, L. J. (1991): An Introduction to the Law and Economics of Intellectual Property, in: Journal of Economic Perspectives Vol. 5, No. 1 (Winter 1991), pp. 3-27.
- Bessen, J. (2001): Open Source Software: Free Provision of Complex Public Goods, working version 4/01, <http://www.researchoninnovation.org/opensrc.pdf> (25.04.2001).
- Bessen, J. and Maskin E. (2000): Sequential Innovation, Patents, and Imitation, MIT Working Paper, No. 1/2000, Cambridge.
- Blind, K.; Edler, J.; Nack, R.; Straus, J. (2001): Mikro- und makroökonomische Implikationen der Patentierbarkeit von Softwareinnovationen. Geistige Eigentumsrechte im Spannungsfeld von Wettbewerb und Innovation, Report on Behalf of the German Federal Ministry of Economics and Technology, Berlin 2001 (<http://www.bmwi.textonly/Homepage/download/technologie/Softwarepatentstudie.pdf>)
- Church, J. and Ware, R. (1998): Network Industries, Intellectual Property Rights and Competition Policy, in: Competition Policy and Intellectual Property Rights in the Knowledge-Based Economy edited by Robert D. Anderson and Nancy T. Gallini, University of Calgary Press, Calgary, pp. 227-286.
- Cohen, W.; Nelson, R.R.; Walsh, J. (2000): Appropriability Conditions and Why Firms Patent and Why They Do Not, National Bureau of Economic Research, Working Paper 7552, February 2000.
- Commission of the European Community (2000): Patentability of Computer-related Inventions: Discussion Paper of DG Internal Market, Brussels 2000.

- Dam, K. W. (1995): Some economic considerations in the intellectual property protection of software; in: Journal of Legal Studies XXIV, June 1995, pp 321-377.
- Dempsey, B. J. et. al. (1999): A quantitative Profile of a Community of Open Source Linux Developers; SILS Technical Report TR-1999-05.
- Economist (2001): Patently absurd?, The Economist, 21. Juni.
- Endres, A. (2000): „Open Source“ und die Zukunft der Software; in: Informatik Spektrum, 23. Oktober 2000; pp. 316-321.
- Farrell, J. (1989): Standardization and Intellectual Property, in: Jurimetrics Journal 1989, pp. 35-50.
- Farrell, J. (1995): Arguments for weaker Intellectual Property Protection in Network Industries, in: Standards policy for Information Infrastructure herausgegeben von Brian Kahin and Janet Abbate, MIT Press, Massachusetts, pp. 368-377.
- Farrell, J. and Katz, M. L. (1998): The effects of antitrust and intellectual property law on compatibility and innovation, in: The Antitrust Bulletin Fall-Winter 1998, pp. 609-650.
- Farrell, J. and Saloner, G. (1985): Standardization, Compatibility, and Innovation, in: Rand Journal of Economics, Vol. 16, pp. 70-83.
- Farrell, J. and Saloner, G. (1992): Converters, Compatibility, and the Control of Interfaces, in: Journal of Industrial Economics, March 1992, pp. 9-36.
- Gehring, R. (2000): Berliner Ansatz zu Open Software Patents; <http://130.149.19.71:8080/Think-Ahead.ORG/Cyberlaw>, 15.03.2001.
- GFK/ISI/IESE (2000): Analyse und Evaluation der Softwareentwicklung in Deutschland. Report on Behalf of the German Federal Ministry for Education and Research; Dezember 2000.
- Gross, G. (2001): Leserbrief zu Albert Endres, „Open Source und die Zukunft der Software“, Informatik-Spektrum, 24th Februar 2001, pp. 38-39.
- Hart, R.; Holmes, P.; Reid, J. (2000): The Economic Impact of Patentability of Computer Programs, Report to the European Commission London 2000.

- Heckel, P. (1996): Deunking the Software Myth, in: Ludlow, P. (Ed.), High Noon on the Electronic Front: Conceptual Issues in the Cyberspace, Boston, pp. 63-108.
- Holmes, W. N. (2000): The Evitability of Software Patents; in: Computer, Vol 33, No. 3, pp. 30-33.
- Horns, A. H. (2000): Der Patentschutz für softwarebezogene Erfindungen im Verhältnis zur „Open Source“-Software, JurPC Web.Dok.223/2000, Paragraphs. 1-80.
- Jaffe, A. B. (1999): The U.S. Patent System in Transition: Policy Innovation and the Innovation Process, National Bureau of Economic Research Working Paper 7280, Cambridge, MA, 1999.
- Kash, D. E. and Kingston, W. (2001): Patents in a World of complex technologies, in: Science and Public Policy February Vol. 28 (1), pp. 11-22.
- Katz M. L. and Shapiro, C. (1986): Technology Adoption in the Presence of Network Externalities, in: Journal of Political Economy 94, pp. 822-841.
- Katz, M. L. and Shapiro, C. (1985): Network Externalities, Competition, and Compatibility, in: American Economic Review, Vol. 75, pp. 424-440.
- Lerner, J. and Tirole, J. (2000): The Simple Economics of Open Source, National Bureau of Economic Research Working Paper 7600, Cambridge, MA, 2000.
- Lutterbeck, B.; Gehring, R.; Horns, A. H. (2000): Sicherheit in der Informationstechnologie und Patentschutz für Softwareprodukte – ein Widerspruch? Report on Behalf of the German Federal Ministry for Economics and Technology, Berlin, Dezember 2000.
- Mazzoleni, R. and Nelson, R. A. (1998): The benefits and costs of strong patent protections: a contribution to the current debate, in: Research Policy Vol. 27, pp. 273-284.
- Mennell, P. (1989): An Analysis of the Scope of Copyright Protection for Application Programs, in: Stanford Law Review Vol. 41, pp. 1045-1104.

- Messerschmitt D.G. and Szyperski, C. (2001): Industrial and Economic Properties of Software: Technology, Processes, and Value; University of California at Berkeley; Computer Science Division Technical Report UCB//CSD-01-1130, Jan. 18, 2001, and Microsoft Corporation Technical Report MSR-TR-2001-11, Jan. 18, 2001 [<http://divine.eecs.berkeley.edu/~messer//PAPERS/01/Software-econ/>].
- Murillo, G. (1998): Institutional Development in the Software Industry: Intellectual Property Protection; UMI Dissertation Abstract, Ann Arbor, Michigan.
- Nalley, E. T. (2000): Intellectual Property in the Computer Programs, in: Business Horizons 43, 4, pp. 43-51.
- Nichols, K (1999): The Age of Software Patents; in: Computer, Vol 32, No. 4, pp. 25-31.
- Nüttgens, M. and Tesei, E. (2000a): Open Source - Konzept, Communities und Institutionen. Working Paper of the Institute for Informatics (Iwi) Saarbrücken; No. 156, Januar 2000.
- Nüttgens, M. and Tesei, E. (2000b): Open Source - Produktion, Organisation und Lizenzen. Working Paper of the Institute for Informatics (Iwi) Saarbrücken; No. 157, Januar 2000.
- Nüttgens, M. and Tesei, E. (2000c): Open Source - Marktmodelle und Netzwerke. Working Paper of the Institute for Informatics (Iwi) Saarbrücken; No. 158, Januar 2000.
- Ordover, J. A. (1991): A Patent System for Both Diffusion and Exclusion, in: Journal of Economic Perspectives Vol. 5, Number 1 , pp. 43-60.
- Oz, E. (1998): Acceptable protection of software intellectual property: a survey of software developers and lawyers, in: Information & Management Vol. 34, pp. 161-173.
- PbT Consultants (2001): The results of the European Commission consultation exercise on the patentability of computer implemented inventions; Notts.
- Romer, P. M. (1990): Endogenous technical change, in: Journal of Political Economy, Vol. 98, pp. 71-102.
- Schmidt; J. (2000): Dasein oder Nicht-Dasein. Analyse der Ausfallzeiten von Web-Servern, in: c't, 8/2000, p.179.

- Shapiro, C. and Varian, H. R. (1999): *Information Rules: A Strategic Guide to the Network Economy*, Harvard Business School Press, Boston.
- Shavell, S. and van Ypersele, T. (2001): Rewards versus Intellectual Property Rights, in: *Journal of Law and Economics*, Vol. 44, pp. 525-547.
- Smarr, L. and Graham, S. (2000): *Recommendations of the Panel on Open Source Software for High End Computing*; President's Information Technology Advisory Committee, 11. September 2000.
- Smets-Solanes, J.-P. (2000): *Software Userright: Solving Inconsistencies of Software Patents*. Contribution „Second Nordic European/USENIX Conference“, Malmö, Schweden, 8.-11.2.2000.
- Stolpe, M. (2000): *Protection against Software Piracy: A Study of Technology Adoption for the Enforcement of Intellectual Property Rights*; in: *Economics of Innovation and New Technology* Vol. 9, pp. 25-52
- Thurow, Lester C. (1997): *Needed: A New System of Intellectual Property Rights*; in: *Harvard Business Review*, September-Oktober 1997, pp. 95-103.
- UK Patent Office (2001): *Software Patenting Consultation*; <http://www.patent.gov.uk/news/softpat.htm> (30.06.2001)
- Webb, R. (2001) *Software & business methods patents: the UK consultations*; Contribution to a Workshop organised by the German Federal German Ministry for Economics and Technology, Berlin 2001.