Teaching Software Engineering and Encouraging Entrepreneurship through E-Commerce

Jörg R. Mühlbacher - Michael Sonntag

Institute for Information Processing and Microprocessor Technology (FIM), Johannes Kepler University, Altenbergerstr. 69, A-4040 Linz, Austria

{muehlbacher, sonntag}@fim.uni-linz.ac.at

Abstract. In this paper we report on a project carried out at the University of Linz, initially involving electronic marketing of a software product developed by a team of students. The aim of the project, in part, was to develop in the students an understanding of practical issues of electronic commerce and entrepreneurship. We report on this aspect and comment also on security issues raised concerning the use of electronic commerce in a small-scale marketing operation, using E-Commerce for improving teaching software engineering as well as (non-monetary) benefits a department can get from being actively involved in E-Commerce.

Keywords: E-Commerce, Software Engineering, Entrepreneurship

1. Introduction: the project CodedDrag

From mid 1996 up to the end of 1997 a team of CS-students at FIM of the University of Linz worked on a project concerning the development of convenient user interfaces, in particular for Windows NT using drag and drop metaphors, context menus and integration as a shell extension. These techniques were quite innovative in those days and, recognizing the lack of availability of smart encryption tools "at your fingertips", the product "CodedDrag" (CodedDrag, 1999) was developed. This program allows encryption/decryption of files and folders using DES, TripleDES and Blowfish algorithms taken from a public library (Young, 1999). A single rightclick is sufficient to secure one's files (or complete folders or subtrees) using a key supplied by the user (the passphrase), which then is unloaded after a user defined amount of time automatically by a parallel thread implemented as a DLL. This combines ease of use (not having to re-enter the passphrase for every action) while avoiding the potential security problem of someone getting access to the computer for a short time (e. g. the user left the room) and then being able to decrypt the data without the key.

As encrypted files can no longer be compressed, LZRW-compression is an option to be done before encryption. Another add-on is the possibility to scramble filenames, so absolutely no information on the contained data is revealed.

After having distributed Beta versions of CodedDrag among the academic community (only) as usual, the rapidly growing demand (incoming e-mails, suggestions for improvements and proposals for extensions) encouraged us to go "E-Commercially". In addition we have regarded this initially unplanned additional step as a research opportunity and started a project of its own for this. We wanted to

? learn more about various facts of E-Commerce in real life,

? study feedback from end-users directly to the developers,

? test the liability of distribution channels,

? find facts on efforts necessary for distributing software electronically,

? check the usefulness of low-price tools for E-Commerce

? get general feedback on teaching the software development process with emphasis on documentation and customer support.

Because of the restrictive legislation concerning cryptography, further development of the product is being frozen at this time. This is necessary as the coming into force of laws implementing the „Wassenaar Agreement“ restricts the export of strong cryptography (even within the EU) and the market in Austria alone is too small. Nevertheless, at FIM research in E-Commerce goes on and we are specifying follow-up projects.

2. Commercial aspects and distribution channels

Taking account of the fact that FIM is a non-commercial oriented institute of the University of Linz, primarily involved in research, development and teaching, we decided to use the following approach:

? The software can be downloaded as freeware from the homepage of FIM (CodedDrag, 1999) with full functionality. However the functionality of encryption will be lost after a period, according to a built in timestamp, although the ability to decrypt what has been encrypted before by CodedDrag will remain indefinitely.

? There will always be an up to date version of CodedDrag available on the net, so that every user can redownload a newly timestamped version whenever his or her version is expired. Hence we can still regard our product as freeware. We think this is fair enough. A convenient side-effect is, that customers are encouraged to use the most recent version.

? In order to ease the burden of having to download a newly timestamped version periodically after approximately 3 months, we offer a non-expiring version and free upgrades sent automatically via e-mail attachments. A user can enjoy this comfort if he or she is prepared to send a small donation to FIM explicitly dedicated to fund student projects and further research.

? Requests for this open version of CodedDrag can be made by letter/fax, e-mail or filling in an electronic form provided on our webpages.

? The donation can be made by supplying the user’s credit card number, cheque or cash (notes). The latter method, for obvious reasons, is not appreciated by users, and cheques are not welcome by us because of high bank charges (they can be as high as 50 % of the proposed donation in the case of cheques drawn from a foreign bank). Therefore we advise people to send their credit card number (and the electronic webform supports this feature only). For handling FAXes we charge a surplus to cover the overhead.

? The delivery is made either by posting a single 3.5? diskette or sending an e-mail with the installation program as an attachment. Again we focus on the latter method because it is much easier to handle and we will supply registered users with updates only electronically via their e-mail address stored in our customer database.

? Any contact with customers is done via e-mail. Only special site-licenses and acknowledgements of receipt (if explicitly required in addition) are sent via conventional “snail” mail.

? For marketing activities we use our homepage only, which contains hidden keywords for feeding automatic search machines. We also enter the URL into search engines and free software directories etc., and try to keep those entries up to date. We do not send advertising e-mails and have no banner-ads, either on our own or on other sites.

? Nevertheless we have to admit, that at the very beginning of the project the ranking of CodedDrag within the top ten list of Windows - system utilities, published by a distinguished software magazine, was an accelerating kick off and welcome free advertisement for us.

3. Experiences

The project brought a lot of experiences, most of them positive, but also a few negative ones. A selection of them will be reviewed here in detail. Special consideration is given to two areas: security issues (as the product is about data-confidentiality and generally in connection with E-Commerce) and problems appearing in connection with customer support.

3.1. General experiences

Up to the present time we have had 120.000 visits to the related homepage located at our server in Linz. Since 1998 we have had at least 3 mirror sites and the cost-free inclusion of the product on CD-ROMs (exact number unknown) which are distributed as add-ons to related magazines. In addition to "requests for registration" we have received more than 2000 e-mails and have been obliged to respond to most of them. We underestimated the workload necessary for efforts on customer relations, although we are happy about the rewards achieved.

3.2. Security issues

A lot of companies are reluctant to start E-Commerce because they are afraid of potential security problems. Though these threats really exist, we have experienced no problems so far. In part this could be because as a university department there is no specific information which might attract intruders and we are also not a primary target for denial of service or destruction of data attacks based on political or commercial reasons. Moreover, as the department has another focus on network-security, we have a reasonably secure web server and network without the widely known loopholes.

It also is worth mentioning that the rate of attempted fraud by sending an invalid credit card number etc. is less then 1 %. This figure includes both calculated „phantom“ card numbers and card numbers where the owner protested the charge (presumably because of a stolen number and expiry date).

Much more important is the following astonishing fact: though CodedDrag is a utility for improving security and keeping information private, we do not use secure e-mail for distribution or a secure web server. Customers seem to accept this and the (surprisingly very few) people who are reluctant to send their credit card details via the net unprotected, have sent a fax instead.

We have not had complaints concerning the way we deliver the product: nobody seems to fear that someone may have modified the program in the e-mail attachment on its way to the end user’s mailbox. The only problems are certain host-configurations, which do not allow users to receive mails with attachments of this length (just above 1 Mbytes in encoded form). The availability of the software as an attachment eases the installation process too.

One possible explanation for this surprising behavior might be, that belonging to a department of a university we enjoy a reasonable amount of trust a priori. Customers waiting for a reply perhaps think that we will be around a bit longer, if their credit-card is charged but the software has not yet arrived. Also both the FIM and the Johannes-Kepler-University of Linz maintain large websites and have their own domain and identification: this is a difference from a single webpage on a free provider (e.g. GeoCities), where anyone can place almost any webpage without having to identify himself to anybody.

3.3. Customer support

Customer support is an important issue, as we received a very large number of e-mails (approx. 2.000) containing questions, inquiries and comments. Assuming that reading and responding to them takes only 3 minutes per e-mail on average, answering only these e-mails took an unexpectedly long time, especially if one includes the time spent on handling database entries, documentation or compiling FAQs etc. This is a lot of time we had to spend for (mostly) unprofitable replies. A closer look at the content of the mails reveals that most of them could have been avoided: the requested information is either already included in the online documentation (placed at a very prominent position within the webpages and easy to find) or readily available on the related webpages. These posts could therefore possibly be answered automatically, taking away a

lot of work (one of our projects is dedicated to this issue). This is also an important observation for E-Commerce. Customers are reluctant to call for support by phone or find writing a letter too time-consuming but no such barriers exist when e-mail is available: just a few lines, a mouse click and you have transferred your problem back to the supplier. Consequently documentation and passive support (e. g. troubleshooting-FAQ on a webpage) is of much greater importance than in conventional business. Active support, e.g. by automatic reply based on content analysis, also becomes important.

4. Immediate benefits from our engagement in E-Commerce

Getting involved in E-Commerce, even though only on a small scale, brings a number of benefits. We will take a closer look at some of them, which are especially important for a university department and in general.

4.1. Implicit marketing and publicity

As is the case for other homepages of institutes and/or university departments the FIM-homepage contains the information typically related to an university department: invitations to seminars, abstracts and full papers for download, description of current research projects, curriculum information etc. This information addresses the needs of related institutes, colleagues and, of course, current students who have to visit our web-pages for the latest news. But, to be honest and realistic, what is beyond this well-defined community?

Recalling the figure of 120.000 visits to the subpage CodedDrag and taking in account that most of these visitors do not belong to academia, the conclusion and message is obvious. Being involved in E-Commerce now, the FIM-institute can rely on an implicit effective advert. This helps in a number of aspects:

1. The university attracts more students: many students of computer science have already had previous personal experience with the WWW nowadays, so even if they only know the name of the institute or university, this can be an aid when choosing where to study CS (or to study something other at this location). This might also increase the awareness of IT-oriented studies, in combination with other campaigns helping to attract more students for this area.

2. The public gets to know about the university: everybody with an interest in our product learns that universities are no ivory towers and they find first-hand information on new developments in the areas the institute works in. Through this the university enters into „everybody's" life, increasing awareness about it and making it more visible to the public. This can serve as an initial idea for joint projects or increased transfer of knowledge from universities to companies and generally acts as a recurring reminder to thinking about lifelong learning.

3. Motivation for students: a part of the donation is used for further research and other projects and the rest is paid as an incentive to the students, who wrote the program. This ensures that the students work on the program and provide updates even though the course is finished. It is also a motivation for students to produce good results if they can expect to get a financial reward in return.

4.2. Encouraging entrepreneurship

Amongst CS-students, it appears that the inclination to establish new companies or to work self-employed is not overwhelming. All our graduates of last year found a job almost immediately, but only a small minority of them started their own business. This might be slightly exceptional because of the Y2K problem: employed software engineers currently receive top salaries and are very sought after; but this could change in the future. Also students are rather taught to work as software engineers and to deliver their services on behalf of somebody else's business. They are less trained in running their own company and are often not self-confident enough to act as an entrepreneur. We think that an early involvement in customer oriented activities and the confrontation with the odds and evens of a new market are effective steps towards spin off foundations, teamwork in a small company or founding new companies in particular. Or at least: to esteem those who are making this step.

Getting involved in E-Commerce can be an important vehicle in this area, as students of CS have fewer problems (see below) than others with this new way of business. E-Commerce can be a chance for new, and therefore small, companies, which can increase the area where they serve their customers through electronic means.

As software can not only be ordered over distance through electronic communication but also delivered and maintained, the various typical areas of work (producing software, design of software-systems, consultation etc.) of CS graduates are especially suited for E-Commerce. At least it should be an obligation of a university to introduce students to this idea.

4.3. Incentives for follow up research and development

FIM has a long tradition of cooperation with small and medium size enterprises (SMEs) and therefore we can state, with all modesty, that we can draw on extensive application oriented experience in joint projects. But the E-Commerce project CodedDrag has opened an entirely new window. The experiences, lessons and insights have brought an additional field of research, which attracts students in particular. The CodedDrag project has led to a list of spin off projects, e.g.: intelligent agents in E-Commerce, protection against spamming, and security issues for SMEs:

1. Spam: As we need to be in contact with our customers, a number of our e-mail addresses are featured publicly on a lot of webpages, both on our server and on shareware repositories. This leads to large problems with unwanted e-mail advertisements, which get sent to those addresses. Sorting them out takes very long, so we are investigating what can be done to at least ameliorate this problem. Both server-side (filters at the e-mail server sorting out all incoming matching mails) and client-side (automatic sorting by the e-mail client) aids are possible. In addition to this we are looking into possible ways to avoid the problem entirely, examples being deliberately wrong e-mail addresses, which the user must change but an automatic program cannot, e. g. jones_spam@domain instead of jones@domain, or measures to prevent sending, like so-called tar-pits.

2. E-Commerce agents: in the Internet it can be very hard for a user to find the cheapest or indeed any shop, selling a special product. Intelligent agents can help in this, as rather unintelligent agents everybody uses (search-engines or directories), demonstrate. More useful in connection with E-Commerce are „comparison-bots“, which know a more or less large number of online-shops and automatically gather product and shipping costs for the user, giving him an easier decision where to shop. In our project we are trying to move one step forward from this by investigating how to provide agents with money, so they can actually buy, for example, the book at the shop providing fastest delivery. In this case security (the agent must not lose the money), prevention/detection of fraud and trust in both own and foreign agents are very important and will be looked into.

3. Intelligent agents for customer support. Selling a software product over the Internet is usually not done by sending the software alone: support is also needed. In our experience, a lot of questions, problems and requests can be easily answered and repeat themselves very often. We are therefore looking into possibilities to answer these E-Mails automatically by employing intelligent agents, which sort out the mails, answer the common ones and forward the unknown ones to manual handling. The data for classification and the answers are taken from a database, which has to be filled manually. In contrast to the agents envisioned in the previous problem, these are stationary agents which are continuously working instead of being charged with a specific task and then stopping or waiting idly. At the same time the different problems are counted to get a statistic for compiling a list of FAQs.

5. Teaching market oriented Software Engineering

Teaching Software Engineering tends to have a bias towards design, coding and testing. The teaching staff is aware of this bias of course. But there are restricted possibilities to emphasize other phases of a (medium sized) software project, including documentation, dealing with end users and doing maintenance upon their requests. Finding the right balance between fulfilling customer wishes and resulting costs is usually beyond what can be covered by a course at the University. Even obvious demands such as online helpfiles, installation instructions and writing handbooks for the non-computer-scientist sometimes remain theory and are

underestimated in both the effort needed to produce them and the time to be spent if one is hampered by customers' questions. There is a great difference between a completed project at the university, which can be submitted to the supervisor, and a final product, which can be sold as it is. The university's responsibility is to make the students really aware of this and, if possible, show it to them through an example.

5.1. Lessons learnt

Frankly spoken, our project has given a lesson to everybody of the involved teams: the team of initial developers, the service staff (bug-fixes and improvements) and the team of those students and staff members, who are responsible for handling the distribution process and all the incoming requests and reports:

1. The initial developers: they found that more testing is needed, if a program is to be distributed to a larger audience. If it is used by a large number of people, even (for developers) rather remote ideas and actions come up (e.g.: scrambling the filename without encrypting the file) and must be handled by the software. A very hard task to teach is producing proper documentation: initial tasks are very simple and larger systems, where documentation is really needed, are only done by students at a very late stage in their course of studies. In this case the students experienced right from the beginning the necessity of documentation, as they personally had to answer the questions resulting from this lack. They also realized the difference between documentation that is written for other developers or computer science professionals (as it is mostly done at the university), and documentation that will be used by „ordinary" users without specific training.

2. The service staff: part of the team of developers has changed during the lifetime of the project. These new staff members had the chance to experience teamwork at the university in a very rare way: instead of forming a team, producing a result and dissolving the team when the project is completed, they entered a group in a later phase and had to build on previous work. They experienced personally the problems resulting from a lack of comments or documentation of design decisions. This personal experience will probably be a much better lesson in proper software development than any lecture. Apart from E-Commerce, this could be an idea to include in the regular curriculum: students working on a larger project in changing teams can learn a lot from this improving previous work. The results are also more interesting and lifelike than the usually rather small programs of student projects. If they are of good quality and of interest to a broader audience, they could probably serve as other objects of study in E-Commerce.

3. The distribution staff: this was a totally new task for students of computer science, who tend to think only of the development side of a product. Now issues like size of the program (one or two disks necessary?), single-file install program (to easily send it as an e-mail attachment), time-limited test versions and distributing updates become issues, not to mention customer databases and organizing the whole process.

4. Customer support staff: they observed that what is simple and easy for a computer scientist can be a hard problem for non-CS-specialized users. Any error or even only a very small inconsistency in the documentation has a long life, even though there is only the most up to date version available on the Web and delivered to the users. On shareware repositories (not submitted by us) or CD-ROMs, the old versions will prevail for a far longer time. There is no longer the chance of repairing an error by simply providing a new version: customers will not re-install the software every week and sending updates costs time (and money in the case of disks). Thorough checking of new versions before making them public is therefore of much larger importance.

Putting everything together, the project gave students a chance to run through the *whole* software lifecycle from scratch, turning a couple of rounds (according to the spiral model (Balzert, 1998)) as well. They have faced a number of problems they otherwise would not have come across before leaving university.

5.2. Improving software engineering teaching by E-Commerce

We will now take a look at some of the later phases of software development and present some ideas as to how E-Commerce can help in teaching them:

1. Testing: programming for actual users provides real life feedback. Teachers often have not the time to do a complete test, they rather test only for certain cases, which might be implemented wrong or are special values which have to be handled separately. In addition to this, the common use of the program is tried. In contrast to this, selling the product (or even trying to do this with free test-versions) provides testers „for free", which will use all of the possibilities and usually find even the best hidden bug. This allows the teacher to get a far better notion about the quality of the program, and the students can experience the importance of testing and learn about all the possibilities they should have checked before.

2. Deployment: Both when teaching software engineering and programming, deployment is entirely or mostly ignored (e.g. not even mentioned in (Balzert, 1996): Implementation ✍ Acceptance test ✍ Installation). The focus is on producing the program, but especially in E-Commerce, where it will be transported using electronic means, this is an important phase. A product should not just have a specific size, consist of a number of files and be somehow installed. When using electronic delivery it is, for example, important that it is only one single file, which can then be easily distributed by e-mail. Checking for transfer errors should be possible (or done automatically when installing) and scanning for viruses before sending is mandatory. Modern software is also expected to have not only an easy to use install program, which works on all possible configurations, but also the option to de-install it. This is very important for free test-versions: They should remove themselves completely and leave the system exactly in the same state as before (in some cases this might be different with time-limited copy protection: the time of the first install must somehow prevail). Also compressing the package to reduce transfer time and cost is an issue. Using an E -Commerce project allows experience of these problems and seeking for solutions: software for the different tasks exists, but it must be integrated to a whole. Many of them are only free for non-commercial usage, which allows them to be used for the project while simultaneously calculating the necessary expenses if it should be sold in reality. This teaches the students a more realistic calculation of the costs for the whole project; otherwise these expenses will be usually forgotten.

3. Revisions: Repeated bug-fixes and improvements are only done for projects which are really in use, i.e. they are either used in the university internally or they are products which are sold. E-Commerce can here be the vehicle to give a reason for re-programming, re-engineering or extension of programs. The students can in this phase experience feedback from the users: they can learn how to classify requests for extensions or bug reports according to their importance and schedule their realization. Questions such as "Is it really a bug or only bad documentation? ", "Are bug-fixes more important than improvements?" or "When should the next revision be ready (so we can set the time-limit for test-versions)?" can be shown to them in this way. A more practical example for things to remember during programming when revisions are expected are version numbers. These must be both human-readable, so the user can easily check if he has the most recent version (or e.g. when it will expire if using a time-limited one), and machine-readable, so the installation program can automatically check them and refuse to install an older version over a new one.

4. Maintenance: Producing improved versions of existing software or re-using former software is for most students a novelty. This is different from extending a well-defined and extensively documented framework, as it is not only necessary to learn how to use it without knowing the implementation but also how it works internally (and where the bugs are). If it is a rather successful program, it will exist over a longer period of time and will allow students to experience another aspect of software development, which is usually strange for the university: working on foreign software exists in research, but not in teaching. For many of the students it will be common in business life to either inspect foreign code, build up on it or improve it. Even when creating a completely new program they will have to include code from other members of the development team, a rare aspect in teaching (often resulting in the „lonesome implementer") as large groups of students in courses are hard to manage and not suited for initial training in programming. Maintenance in an E-Commerce project can teach about working in a larger team: using foreign code, perhaps getting short explanations or answers from the previous programmers or using and extending both development and user documentation.

Generally speaking, E-Commerce allows the students to experience real-life software development instead of theory only while avoiding a lot of the initial and continual costs of a regular software company. This allows using the best teacher available: practice.

Another idea is to create a common software pool at the university, where all students must place their projects and the documentation for free download. In addition to simply placing it there, they must also create a short presentation for it, so everybody can search for relevant existing software easily. When downloading, the user has to enter how much he would pay for the program. This provides real feedback to the author and could be used for incentives for the students to improve future maintenance for the program. If the program is used extensively, it can then be further improved by projects building up on the first version. The students can in this way gather the benefits of producing their software for „real" E-Commerce while avoiding legal complications and being competition for regular companies.

5.3. E-Commerce as a tool for teaching virtual teamwork

In conventional teamwork, the participants of a group work closely together and meet at the same time at the same place. In contrast to this, members of a virtual team won't meet physically and sometimes not even at the same time; they might never have met the other participant personally. Instead they use electronic communication for coordination and collaboration. Both in research and in business this mode of working grows ever more important as national boundaries get less important and international companies increase in number and size. The improving possibilities (telephone ✍ e-mail ✍ video-conferencing) and quality (more bandwidth available) of electronic communication makes this ever more easy. As computer science produces the means for this virtual teamwork, computer scientists should be the first users of this emerging option. Many of their tasks (e.g. software development or design of networks) are suited very well for this, as their data is already available electronically. Because of this, the students must also be taught how to use common tools and what special adaptations in contrast to conventional teamwork are necessary. E-Commerce is especially suited for teaching and practicing this, as producer and consumer are already separated and interact only by electronic means. So the extension to virtual cooperation during the development of the product, the design of the documentation and advertisements, the marketing and the bookkeeping is easier to explain.

In this context a joint project between different fields of study is possible: Designing a product, deciding on a marketing strategy and selling it through E-Commerce. This is especially suitable, as many non-computer science aspects are touched, like marketing, accounting, creating a price structure, calculation of costs, statistics, writing license agreements etc. This allows the possibility to integrate students of other departments like industrial management, jurisprudence or statistics. This has an advantage for all participating persons: non-computer scientists learn about using computers, electronic communication and E-Commerce and CS-students get a glimpse of the other problems relevant to actually selling a product. All of them learn how to cooperate with other persons, who are sometimes thinking in considerably different ways. As these other departments might be located somewhere else, working together in virtual teamwork is an essential part of the project. So close virtual teamwork is used within the groups working on a special aspect (e. g. selecting the different payment methods, arranging it with the credit-card companies , ...) while looser cooperation exists between the different teams, for example when agreeing on a common time-plan or discussing the special features of the product to be shown in the marketing and generally when exchanging or combining results. This way of having to solve a real task in coordination and cooperation with other distant people is the best motivation for the students to learn and practice virtual teamwork.

6. „Small-scale" E-Commerce

In our experience also „small-scale" E-Commerce is possible. Although CodedDrag alone would not be sufficient as a single source of income, starting like this is a good way to sell smaller products with little investment. The big advantage is that although a lot of work is needed, the personal financial risk is very low. Because of this, E-Commerce can offer a chance to successfully start a new company. There is also the possibility to reach more customers if an already established company starts selling its products online. In this case, however, E-Commerce is only viable if a working IT infrastructure already exists, or the costs of starting this new branch will be rather high (see below).

In particular, we will take a closer look at a few specific problems and give some possible solutions:

? Marketing: If electronic means are the only way of selling the product, getting an audience is the most important task. A single webpage is easily overlooked in the huge WWW and it is therefore necessary that it can be found as fast as possible by potential customers. Good starting points for this are entries in search engines, shareware listings and links on other sites.

? IT infrastructure/expertise: Introducing E-Commerce to an existing company is possible only if the staff already has expertise in the use of computer networks and a lot of data is available electronically. To be successful, electronic shopping must provide additional value for the customer. Only in this case will he use (and later re-use!) these online offers. Typically this would be a lot of information on the product, including links to other products or further information on other websites. Providing this data requires a lot of work if it is not readily available, producing high initial start-up costs. Also regular maintenance is necessary (information updates, replies to inquiries over e-mail, new links , …), which must be done in house.

? Payment methods (Sonntag, 1999): Our experiences showed that only credit-cards or bank transfers are used for payment. Accepting bank transfer of invoices has some risks: either you send the product and have to hope for the money (suing for it in a foreign country is probably more expensive and risky than it is worth), or risk angry customers, who already have transferred the money but have not yet received the software (bank transfer might take more than a week if SWIFT is not used). Cash on delivery is also not a suitable option if doing international business. Electronic cash has currently a much too small audience and too high prerequisites on the technical side to be an issue for small-scale E-Commerce. So credit-cards are the only really convenient way for payment, both for customers and online shops. An exception from this rule is, if you already know your customers and just transfer business from personal handling to electronic communication or operate only in one country.

? Delivery methods: whenever possible try to avoid changing the media. With respect to E-Commerce this means to deliver your services or goods through electronic means if feasible. This is the cheapest version as handling disks or CD-ROM's (duplicating, labeling, writing an accompanying letter , ...) can easily take more time (and therefore cost for staff) than the cost of the materials the mselves.

? Setting up a web-shop (Lohse, Spiller, 1998): in the internet the comparison between shops is much easier for the customer, so even very small shops are compared to huge ones. It is therefore important that the electronic marketplace is very easy to navigate and use and that the information is extensive and correct (no dead links!). The shop is the only part of the company visible to the customer so it should be both graphically attractive and provide value for the user. A lot of internet-shops we visited suffer from a lack of usability, so they can hardly be successful. It should also be noted, that for a small shop it is not necessary to provide a shopping basket, multiple pages for delivery/payment options, etc. A simple webform with explanations, as we used, can be more than sufficient for selling a small number of diffe r-ent products.

? Secure server (Garfinkel, Spafford, 1997): using a secure server seems not to be necessary at the mo-ment but widespread use of certificates and digital signatures will make this an important issue in the future. The problem is that hosting a secure site is more complicated and elaborate compared to a co n-ventional one. But the option for migration should be included in the plans for further extensions. This not only provides authentication of the customers to the shop and the shop to the customers, increasing mutual trust, but offers also the possibility to include e ncryption for the communication.

We can therefore conclude that a company which is already equipped with a working and extensively used IT infrastructure can rather easily open up a new distribution channel through E-Commerce for certain goods and delivery/payment methods. In contrast to this, „conservative" companies are not immediately suited for this. Starting a company by doing only E-Commerce will probably be seldom successful. On the other hand, using it as a job on the side can provide a start for later full-time business.

7. Summary: Worth doing it

E-Commerce provides a rather new field of activities, even (or in particular) for computer scientists. We have decided to accept this challenge. We did not restrict ourselves to watching what other people do and just commenting on or reviewing their efforts; we have been and are being actively involved. By first hand experience we can state that it was worth doing it. Students are supported by monetary incentives, they are learning additional features of the complex software development process and the department can welcome new visitors to our home page daily. Last but not least new research projects have been started, based on these experiences made so far.

References

H. Balzert. "Lehrbuch der Software-Technik: Software-Entwicklung." Spektrum Akad. Verlag, 1996

H. Balzert. "Lehrbuch der Software-Technik: Software-Management, Software-Qualitätssicherung, Unternehmensmodellierung." Spektrum Akad. Verlag, 1998

CodedDrag Homepage: http://www.fim.uni-linz.ac.at/codeddrag/codedrag.htm (1.7.1999)

S. Garfinkel, G. Spafford. "Web Security & Commerce". O´Reilly, 1997

G. L. Lohse, P. Spiller. "Electronic Shopping. Designing online stores with effective customer interfaces has a critical influence on traffic and sales." In *Communications of the ACM* 1998, Volume 41 Number 7: 81-87

M. Sonntag. "Rechtliche und Technische Fragen des E-Commerce." http://www.fim.uni-linz.ac.at/staff/sonntag/ecommerce.htm (1.7.1999)

E. Young. "SSLeay". http://www.psy.uq.oz.au/~ftp/Crypto/ (26.7.1999)