Software Engineering Education: Different Contexts, Similar Contents[*]

Pavol Návrat, Mária Bieliková
Slovak University of Technology,
Department of Computer Science and Engineering
Ilkovičova 3, 812 19 Bratislava, Slovakia
{navrat,bielik}@elf.stuba.sk,
http://www.elf.stuba.sk/~{navrat, bielik}

Abstract

This paper discusses two important aspects of the software engineering related education: the context, both the current one and the historical background of the development and the contents of the education. We concentrate on the situation in Slovakia which has been very similar to other countries in the Central Europe, so it can be considered representative in some sense.

1. Introduction

From a general point of view, one of the key actors in implementing the technological innovations centered around the progress in information technology is a software engineer who analyses, designs, implements and maintains various kinds of information systems of businesses, organizations, etc. The demand of software engineers is a result of the technological progress witnessed throughout the world.

The statement is probably true in general, and it holds in particular also for a typical Central European country like Slovakia which has been going since 1989 through a period of social and economic transition without a precedent in its history. The demand here is also a result of an effort to accelerate the development of the country to overcome the consequences of the past un-natural direction of development that threw the country not only technologically, but also socially, economically, and perhaps, most of all, morally back as well. From this point of view, spreading as widely as possible the information technology and its use is extremely important as it provides a firm platform for a free exchange of information and ideas. In such a way, it strengthens the fundamental attributes of a free society and makes the trend towards it practically irreversible.

It is becoming clear that there are, among other things, at least two important aspects of the software engineering related education to be taken into account and discussed at some length. One of the aspects is

- the context,

both the current one and the historical background of the developments in this region during the last fifty years. It is rather different from what has been the development in the rest of Europe or in North America, to name just two regions with the greatest relevance for the region of Central Europe. We shall discuss some of the differences, such as the systems of higher education, or tradition, but name also other aspects such as availability of the high technology.

The other aspect is

- the contents of the education.

Roughly speaking, it has been similar to a comparable education offered elsewhere. Even more, in some sense it could not have been any radically other way. Despite that, naturally also here some differences can be observed. We shall point at them especially with reference to those aspects mentioned above.

2. Different contexts

For the countries of Central and Eastern Europe, Western Europe and North America are undoubtedly the two regions that are from the point of view of civilization, culture, and level of development the closest ones. Yet during the last fifty years or so, their developments were in many ways radically different. When focusing on software engineering education, the differences in systems of higher education which define the immediate context of it are worth of some attention.

However, it would be a too ambitious task to attempt to compare them here because of the lack of space and more importantly, the lack of comprehensive data, so we restrict ourselves to a few comments. First, there are

[*] This work was partially supported by the Slovak Science Grant Agency, Project 95/5195/605.

not just two or three systems of higher education, but each of the respective regions itself defines a variety of different systems.

North America

From our point of view in Central Europe, perhaps North America, with all the variations and flexible options, is the region with

- the most internally compatible system.

While we are not in a position to compare Canadian and the US systems, or to identify their differences, precisely the fact that there are in the region just two countries, moreover bound so closely by language and culture, makes achieving some level of compatibility easier. Any characteristics of it, no matter how brief, should mention the use of credits and the number of levels i.e., baccalaureate, master's, and PhD. levels of education where the previous level is conceptually intended as a prerequisite for the next level. These characteristics are not necessarily absolutely correct neither they may be considered as the two most distinguishing features of their system by our American colleagues, rather they are a reflection of what we in our current situation see as the features that are most radically different from the system that was maintained here.

Moreover, we should like to stress that we concentrated more on characterizing the aspects of organization of education. A more general context has been obviously more fundamentally different: a free society, university self-rule, strive for individual success along with individual responsibility. It should be emphasized that these aspects of the context are vitally important today for all our higher education, not just the one related to software engineering.

With respect to software engineering, it has been developing in America frequently as a split off from computer science curricula. As Ford [4] points out, it started to be introduced primarily at the master's level. Currently, however, there are already being introduced baccalaureate software engineering courses [7] .

Sooner or later, the development of software engineering education must have reached the stage at which emerges

- the question of accrediting the courses.

Depending on whether more emphasis is given on its roots (i.e., computer science), or its designation (i.e., engineering), two options are available: either to accredit the course with the CSAB, or with the ABET. Besides showing the immaturity of the field, the question is not just a simple flip the coin choice, but has a deeper context and possible consequences. At the moment, the engineering nature of software engineering can still be questioned [16], and rightly so; but the question itself is not only legitimate, it is very productive in provoking discussions on the engineering nature of software engineering [6], or on prospects for this discipline [12, 15]. Moreover, legislations of at least some states place various constraints on engineering education which prevent software engineering from being acknowledged as engineering (profession) there.

Another question is that of requirements that are to be fulfilled in order for the course to be accredited. Formal requirements exist; definition of the contents of such a course doesn't. There has been launched a joint endeavour by the ACM and IEEE to identify the software engineering profession, to define the body of knowledge, and consequently to recommend contents of the curricula.

Western Europe

Europe, or more precisely Western Europe, shares some common characteristics but

- the national systems are in many ways different.

Even if we take European Union which can be considered the core of it, there is no unified system of higher education. Consider just such countries as Great Britain, the system of which is in some aspects closer to the American one, and Germany, which has in some aspects a radically different system of organization, degrees, etc. It should be noted, however, that there are ongoing discussions and some measures taken to bring the national systems of higher education closer to each other so that they would allow e.g., mobility of students across European universities (by establishing a course credit transfer system, and an academic recognition system).

In Europe, software engineering started to be introduced into curricula some time ago, too, although no comparable study to those of Ford is known that would similarly deal so extensively with higher education in software engineering in Europe. In Europe, computer science is frequently called Informatics. This is done not only by the French and the Germans in their languages, but also by most of the other nations, and interestingly also when expressing themselves in English.

Sometimes, they use the terms computer science and informatics almost interchangeably. Interestingly, the word which is considered by some people (most notably, some Americans) and by most dictionaries an English un-word, has been recently accepted by the English to the extent that there is a School of Informatics at the City University of London (which offers courses in software engineering, by the way). However, whether Informatics will establish itself as the precise synonym of computer science is not clear. Rather, the term seems to be understood to describe a wide scope of processes and principles related to storing, processing, communicating, and retrieving information.

Obviously, there does not exist a European accreditation board. On the other hand, for example as far as engineering is concerned, there exists an effort to define

- so called European engineer

as a degree that is awarded by a university and that has been recognised as conformant with requirements of the European Union [13]. Some universities attempt to accredit their courses with the IEE; indeed, there are numerous examples of not British, but overseas universities which successfully accredited their courses. However, the procedure is strict in general requirements, but much less specific in software engineering, or for that matter, any other special branch of engineering related ones.

Central and Eastern Europe

Central and Eastern Europe shared for decades a radically different context characterized by the lack of freedom, strive for collectivism, and the overall responsibility of the state. Although no single unified system of higher education existed,

- the Soviet influence was dominant

also in this area so their model was reflected in the particular national systems of higher education. Again, we do not possess enough data nor know about any comprehensive study to be able to speak about the situation in the region in general.

On the other hand, the situation in Slovakia was very similar to other countries in the region, for the reasons indicated above, so its context can be considered representative in some sense. In Slovakia, there were no credits used and there were only two levels of higher education. First one was master's level in humanities, social, natural, etc. sciences, or equivalently engineer's (Ing.) level in engineering, technology, economics, transport, agriculture etc. It took typically five to six years to complete and included elaborating a thesis. Second level was called somewhat obscurely and misleadingly candidatus scientiarum (CSc.) as it took at least further three years of study, included elaborating a dissertation and was thus comparable to the PhD. level known elsewhere.

A major change to higher education in Slovakia succeeded in 1989. It is perhaps worth mentioning that one of the two most important demands of the democratic movement - besides abandoning the "leading role" of the communist party - was an immediate halt of the Marxist indoctrination in the education, both of which had been enforced by the corresponding clauses of the constitution. Success in these two demands, i.e. deleting the clauses from the constitution, marked the actual moment of collapse of the regime. The change allowed to pass a new higher education in law 1990. The new law has established an entirely new and different context for the higher education. The law guarantees academic liberties and radically widens the extent of academic autonomy. The law introduces three levels of higher education; however, it leaves their introduction, and indeed their definition largely up to the decision of a particular university.

Practically,

- problems arised with the baccalaureate level

which has not been backed by any widely shared interpretation nor experience nor subsequent legislation that would define the status of the graduates. As a consequence, some universities decided to maintain the "old" master's courses but to introduce new baccalaureate courses running in *parallel*. The idea was to make them largely independent and not transferable with the master's courses intended as technical, non-scientific education preparing for direct entry to industry positions. There seems to be the problem of interpretation still present. The institution that was offering such courses was an university, or an university of technology, and not a *college*, or a *polytechnic* or a *Fachhochschule*. The students were expecting to receive a complete higher education from it and were not prepared to exchange one or two years "saved" for some new title that is largely unknown and lacks any tradition.

Other universities, on the other hand, seized the opportunity offered by the new law and started to offer baccalaureate courses that were in *series* with the subsequent master's and PhD. courses. This involves a far reaching conceptual change also in the master's courses. They do not cover the whole interval of the higher education (apart of the PhD. studies) anymore, but contract to the second level only.

Implementation of this model has opened, for the first time in decades, door for students to design their education with much more freedom because of the inherent flexibility of the scheme. They start realising that they can receive two degrees for example in two different fields, but within a period that would before suffice for just one, even if the higher one of them. Or, to put another way, before this scheme has been adopted, it took twice as long time to acquire two specialisations.

However, the process of shifting the attitudes of students appears to be a rather slow one, because of strong influence of the tradition. Other major benefit of the increased granularity of the higher education scheme is the option to finish the studies after no more than four years (with the baccalaureate degree). At the moment, the problem with this option is once again the lack of tradition; there is no experience with such graduates, and the public will need some time to find a justified and adequate interpretation. With the first baccalaureates graduating now after four years of their studies, it will soon be possible to start gathering experience.

One of the consequences of the new legislation in 1990 was an introduction of

- a permanent accreditation process.

An accreditation board was established that counsels the government mainly by evaluating the universities with respect to the predefined formal standards and quality criteria of the education process and educating staff in the offered courses. Positive outcome of the accreditation process is necessary for the university to acquire (or to keep) the right to award a particular degree in a particular major field of study.

From the above outline of the context of higher education, there are some conclusions to be drawn. The development in Central and Eastern Europe has been un-natural, forced by the communist rule and forcing severe compromises in form and contents. The compromises in engineering, natural and a few other sciences were among all the fields the least severe and were largely reduced to three or four social and political sciences subjects that were turned into Marxist propaganda. The rest of the curricula, i.e. the absolute majority was not contaminated. Here, the biggest problem was the insufficient hardware and software equipment of the laboratories, caused by the export restrictions of the western countries and the technological backwardness of the eastern countries.

3. The contents

The contents of the higher education related to software engineering has been developing in all the mentioned regions basically

- along similar lines and towards similar resulting curricula.

Obviously, there are differences caused by differing needs of the respective industries, by different levels of development of the technologies, or by different amounts of support for science and education in the respective countries, which in turn may have caused varying relative time delays in adopting the fundamental new scientific results or methodologies into the curricula. Obviously, an initiative such as that of establishing the Software engineering institute at CMU turned it quickly into a forerunner in the field and it has been bringing its fruits to the whole US.

One of the crucial questions concerns

- the fundamental skills and knowledge that characterize the professional software engineer [11].

The importance of it is best documented by the establishing of a joint ACM/IEEE Task Force to identify the body of knowledge of the software engineering profession. Implicitly, the assumption behind the effort is that a complete and comprehensive answer to the question is not known to date. Ideally, however, the body of knowledge should be identified before a curriculum can be designed that aims to transfer the knowledge to the learner. Luckily, there are other sources that describe the body of knowledge - here belong obviously monographs, but also e.g., the Ford report [4] which provides a detailed mapping onto a technical content of an undergraduate curriculum. His other report [3] deals extensively with the graduate education.

So the situation in knowing what to teach in Software Engineering is not so entirely pessimistic. However, the problem is of a more fundamental nature, viz. the field itself is not mature enough to have its own theory, scientific principles and engineering practices formulated and standardized the way as other, "older" engineering disciplines have. Some would prefer to strengthen the undergraduate Computer Science curriculum instead of splitting and creating the undergraduate Software Engineering course [16]. Some even deny Software Engineering is a serious discipline [2]. Some envisage software as an engineering discipline [12, 6]. We are strongly committed to developing the discipline of Software Engineering, not least because it quite naturally fits the challenges of a university designated to technology.

Besides that, the context as outlined above has been influencing the development of curricula. In Slovakia, there has been a rather sharp distinction between "classical" universities and universities of technology. The former traditionally never offer an engineering degree, while the latter never offer other but such degrees. Therefore, the contents of the related curricula have been at the former type of universities very close to pure computer science, and at the latter type a broader computer science and engineering combination.

Integrated Informatics Curricula

Traditionally at Slovak University of Technology, the Informatics curricula have been designed as a blend to cover the broad field of Computer Engineering, Computer Science and Telecommunications Engineering. A relatively major revision in 1989 involved a formal integration of Computer Science (reshaped and renamed to Software Engineering) and Computer Engineering tracks with Telecommunications Engineering, but was at the same time accompanied by increasing the curricula's internal flexibility and by implementing the concept of elective subjects and blocks in a far greater scale than before.

We find the connection of Computer Engineering and Software Engineering very useful for both sides. In fact, the response from industry strongly supports it as there is frequently being expressed an appreciation of the range of principles and techniques mastered by our graduates who even when specializing in software, have a strong background in understanding the models and processes inside the computer architecture. As another positive feature of our course, there is a large amount of practical work required as most of the subjects include labs.

Similar, albeit naturally somewhat looser is the connection of both the Computer Engineering and Software Engineering tracks to Telecommunications Engineering. An internal course's flexibility provides enough room for sufficient autonomy. On the other hand, common Informatics compulsory subjects build a base of knowledge from all three engineerings for all Informatics students. Moreover, electives chosen across the specializations allow students to shape their paths to become software or computer engineers with a relatively strong background in Telecommunications, and vice versa, which is becoming a rather interesting option.

As mentioned above, the Informatics course's revision of 1989 also reflected the shifting focus of interest of the industry by

- changing the name of the Computer Science track to Software Engineering.

Of course, changing a name does not make any difference in the content, but the measure was accompanied by setting a goal of gradually introducing more subjects supporting this shift. Developing and introducing Software Engineering related subjects requires faculty with sufficient expertise and experience. To achieve this, we followed several paths [10]. First, we re-oriented our development activities for local industry to become more involved in typical software engineering projects. Presently, several members of the faculty regularly work on development projects for industry [1].

Second, we try to hire experienced professionals from industry as external lecturers, thesis supervisors etc. whenever possible. Last but not least, we made every possible use of European Union projects (under the TEMPUS programme aiming to foster European transfer of expertise in the educational area) that were focused on Informatics [8, 9] to acquire experience from our partners who were already teaching software engineering related subjects, most notably from the Sheffield University [5].

The last major revision of our course took place in 1993. It was connected to the University's decision to make use of the possibility of

- introducing the baccalaureate level of education.

Introducing the baccalaureate level into a (then) single Informatics course opened several questions that were concerned also with the content of the curricula, thus going beyond the simple acceptance of the new situation of having two Informatics courses. What appeared initially to many to be merely a formal split into baccalaureate and master's levels, induced on second thought serious questions.

It quickly turned out that the splitting revision must first identify the different goals and objectives of the respective courses and consequently to design the curricula as careful transformations of the original one.

Nevertheless, elements from the computing curricula [14] are strongly present in our undergraduate Informatics curriculum. This builds a strong scientific and engineering foundation, but leaves relatively lesser room for subjects specific for software engineering in its "narrow" sense (despite the fact that it typically takes four years to complete). We offer at least a few subjects (such as Principles of Software Engineering) that provide a detailed introduction to software life-cycles, specification techniques, analysis and design methods, implementation approaches including verification and validation, project management issues, and cover structured system analysis and design in a considerable depth to educate the students sufficiently in the software engineering profession so that they acquire necessary skills to be able to assume adequate jobs if they do not continue in graduate study.

Moreover, the context is such - as explained above - that most of the graduates will continue to study the subsequent master's course.

For both of the above reasons, we concentrate the software engineering education mainly to the graduate Informatics course. It is a three semester course with one semester devoted entirely to a thesis project. The first two semesters include one Mathematics, one Computer or Telecommunications engineering subject, a thesis project seminar, and a two semester team project. One humanities elective must be included, and there is also one free elective. The rest are five Software Engineering electives.

4. Conclusions

We attempted to identify some of the differences in contexts of higher education in relation between Central and Eastern European countries on the one side and Western European and North American countries on the other.

We noted that contrary to the common practice in countries with a long tradition of a two level higher education, where the majority of graduates do not continue towards the master's degree, our tradition dictates - at least for some time - to an overwhelming majority of graduates to proceed towards the master's degree. As a consequence to software engineering education, there is not felt any strong need of more specialising already at the baccalaureate level.

It is not clear at the moment what the trend will be in the long run. Undoubtedly, some baccalaureates will soon realize they can be sufficiently successful without having the traditional higher degree. On the other hand, it is to be expected that one of the basic motivations for splitting the higher education into two levels, viz. allowing student mobility across different fields, will start to function with e.g., students completing their first degree in a different area but wishing to acquire a strong competence in Informatics, too. Our graduate course is open for such students who will be required to study approximately one year longer to complete the necessary prerequisites.

We also report on an interesting experience of developing Software Engineering education within integrated Informatics curricula. This solution avoids making Software Engineering too narrow or too loosely defined due to the lack of sufficiently comprehensive own theory and methodology. Instead, a firm base is created by widening Computer Science foundations with some fundamental principles and theories of Computer and Telecommunications Engineerings with reference to common scientific problem of algorithmic information processing. Building on that, specific software engineering models, architectures and methods are developed.

From among the two Informatics curricula, the graduate one offers more breadth and depth specifically regarding Software Engineering, while the undergraduate one is more general in offering breadth in Informatics and going deeper into Software Engineering only to a relatively lesser extent.

References

[1] Bieliková, M., Galbavý, M., Kapustík, I., Molnár, Ľ. and Návrat, P. Using a CASE tool in developing an information system for Slovak Telecom. In Information Tools and Technologies, 159-164, Moscow, 1996.

[2] Dijkstra, E.W. On the cruelty of really teaching computing science. *Commun. ACM* 32, 12 (Dec. 1989), 1398-1404.

[3] Ford, G. SEI report on graduate software engineering education. Tech. Rep. CMU/SEI-91-TR-2, CMU Software Engineering Institute, 1991.

[4] Ford, G. A progress report on undergraduate software engineering education. Tech. Rep. CMU/SEI-94-TR-11, CMU Software Engineering Institute, 1994.

[5] Holcombe, M. Software engineering. In D. Linkens and R.I. Nicolson, editors, Trends in Information Technology, chapter 2, 17-37. P. Perigrinus/IEE UK, 1988.

[6] Leibfried, T.F. and MacDonald, R.B. Where is software engineering in the technical spectrum? *Int. Journal of Engineering Education* 8, 6 (1992), 419-426.

[7] Lutz, M.J. and Naveda, J.F. The road less travelled: A baccalaureate degree in software engineering. In Twenty-eight SIGCCSE Symposium on Computer Science Education, 287-291, 1997.

[8] Molnár, Ľ., Návrat, P. and Vojtek, V. PARLAB - laboratory for parallel computing and programming. In Proc. TEMPUS Workshop on Engineering and Related Sciences, pages 73-74, Bratislava, 1996.

[9] Molnár, Ľ., Návrat, P. and Vojtek, V. Promoting education in the field of software engineering. In Proc. TEMPUS Workshop on Engineering and Related Sciences, pages 31-34, Bratislava, 1996.

[10] Návrat, P. and Molnár, Ľ. Curricula Transformation in the Countries in Transition: An Experience from Slovakia. *IEEE Trans. on Education* 41, 2 (May 1998), 1-4.

[11] Rösel, A. and Bailes, P. Identifying foundations of software engineering expertise. *SIGCSE Bulletin* 24, 4 (1992), 52-63.

[12] Shaw, M. Prospects for an engineering discipline of software. *IEEE Software* 7, 11 (1990), 15-24.

[13] Simpson, I. Engineering education in Europe. *IEEE Trans. on Education* 37, 2 (1994), 167-170.

[14] Tucker, A.B. A summary of the ACM/IEEE-CS joint curriculum task force report: Computing curricula 1991. *Commun. ACM* 34, 6 (June 1991), 69-84.

[15] Wasserman, A.I. Towards a discipline of software engineering. *IEEE Software* 13, 6 (1996), 23-31.

[16] Wulf, Wm.A. Computer science and software engineering: Splitting is the wrong solution. *Computer Science Education*, 3 (1992), 123-134.