Qualidade e Produtividade de Software

Survey on the Relevance of Topics in Computer Science Education

Christiane Gresse von Wangenheim
Djoni Antonio da Silva

Technical Report LQPS001.09E

Date	06/05/2009
Version	Final
Distribution	Public

LQPS - Laboratório de Qualidade e Produtividade de Software
Universidade do Vale do Itajaí - UNIVALI - São José
São José/SC, Brazil

Abstract

We present the results of our survey on the importance and amount learned of Computer Science topics in the career of software professionals. The survey has been conducted in 2008 among Computer Science graduates in Brazil. For each topic, we asked them how much they had learned about it in their formal education, how much they know now about it and how important the topic has been in their career. The objective of the survey was to provide data that can be used to improve the education and training of software professionals. The results suggest that various topics may be under-represented in Computer Science courses (specifically, with respect to software, software engineering and related areas), whereas others (including mathematics, hardware and computational science) may be less relevant for Computer Science curriculums.

1. Introduction

We plan and execute a survey research on the relevance of topics on Computer Science education following the Survey Research Process as described by (Kasunic, 2005).

Our research is strongly based on the research done by (Lethbridge, 2000) (Lethbridge, 1999), who conducted surveys to identify topics where software practitioners felt they needed more or better education. A replication of this research has also be done by (Kitchenham et al., 2005).

1.1 Research objectives

The goal of our research is to provide information on the relevance of topics in computer science undergraduate courses (specifically, with respect to the software engineering and software process improvement education) that may assist researchers and educators to improve existing curricula. Through this survey, we, specifically, want to know, if our curriculum is appropriate to the needs of undergraduate computer science students.
This survey is also part of our research on software engineering (and software process improvement) education and is intended to point out potential areas that should be strengthen in accordance with practical needs of software professionals today.

The goal of our research is to provide information on the relevance of Software Engineering topics in computer science undergraduate courses that may assist researchers and educators to improve existing curricula in accordance to the needs of software professionals.
Therefore, we want to know:

(1) which CS topics are most important for a software professional;
(2) if the amount of (university) education dedicated corresponds to this perceived importance corresponds;
(3) if software professionals learn on the job what is necessary, and
(4) if the importance of topics has changed in comparison to the (Lethbridge, 1999) and (Kitchenham et al., 2005) survey.

1.2. Target audience

The target audience of the survey is Computer Science Bachelors, who graduated during 1998 – 2005 from a Computer Science undergraduate course in Brazil and who work as software professionals. We selected this target population because it includes graduates who have been in industry long enough to have experienced a reasonably wide range of software jobs and tasks and on the other hand excluded graduates who left university so long ago that they experienced outdated curriculum. Due to the focus of our research on Computer Science, we also excluded graduates from other courses, such as Computer Engineering or Information Systems.

2. Questionnaire design

Revising the questionnaire developed by (Lethbridge, 1999) and (Kitchenham et al., 2005), we updated the questionnaire taking into consideration on one side computing curricula recommendations (The Joint Task Force on Computing Curricula IEEE CS/ACM, 2001), (The Joint Task Force on Computing Curricula IEEE CS/ACM, 2004), (The Joint Task Force for Computing Curricula ACM/AIS/IEEE-CS, 2005) (MEC, 1999) as well as revising, principally, the set of topics related to Software Engineering based on SWEBOK (IEEE Computer Society, 2004) and the CMMI framework (CMMI Product Team, 2006). The list of topics has also been revised by professors of various areas of the computer science course at the UNIVALI/São Jose.

Table 1 presents the topic and categories considered in the survey.

Software Data structures and algorithms File management Databases Specific programming languages Programming language theory Parsing and compiler design Performance measurement and analysis Computational complexity and algorithm analysis Artificial Intelligence Pattern recognition and image processing Computer graphics Human Computer Interaction/user interfaces Security and cryptography Operating systems Data transmission and networks Parallel and distributed processing Real-time system design Web-based programming	**Software engineering** Project management Requirements development Requirements management Formal specification methods Object-oriented concepts and technology Software architecture Software design and patterns Software Testing Software reviews and inspections Software quality assurance Software configuration management Maintenance, reengineering and reverse engineering Software metrics Software reliability and fault tolerance Software cost/effort estimation Software process and process improvement (CMMI, etc.) Software engineering tools
Hardware Digital electronics and digital logic Microprocessor architecture Computer System architecture Analog electronics Robotics Digital signal processing VLSI design	**Mathematics** Differential and Integral Calculus Differential Equations Linear Algebra and Matrices Probability and Statistics Mathematic logic Graph theory Combinatorics Functions, relations and sets
Related areas Physics Chemistry Economics Accounting Marketing Management Entrepreneurship Psychology Philosophy Technical writing Giving presentations to an audience Team work skills Leadership Negotiation Legal/professionalism/ethics and society Statistical process control Scientific methodology Second language	**Computational science** Numerical analysis (Operations research Modeling and simulation

Table 1. List of topics

Changes made to the original list of topics defined by (Lethbridge, 1999) are documented and commented in Appendix B.

Regarding our research questions, for each of those topics, we asked the same four questions as defined in (Lethbridge, 1999) illustrated in Table 2. Each of these questions has responses associated on its own 6-point ordinal scale from 0 to 5 defined in the context of the question.

1. How much did you learn about this during the computer science course? 0 =Learned nothing at all 1=Became vaguely familiar 2 =Learned the basics 3=Became functional (moderate working knowledge) 4= Learned a lot 5=Learned in depth, became expert (learned almost everything)	2. What is your current knowledge about this, considering what you have learned on the job as well as forgotten since you graduated? 0=Know nothing 1=Am vaguely familiar 2=Know the basics 3=Am functional; (moderate working knowledge) 4=Know a lot 5=Know in depth/ am expert (know almost everything)	3. How useful has this specific material been to you in your career? 0=Completely useless 1=Almost never useful 2=Occasionally useful 3=Moderately useful, but perhaps only in certain activities 4=Very useful 5=Essential	4. How useful would it be (or have been) to learn more about this (e.g. additional courses)? 0=Pointless learning more 1=Very unlikely to be useful 2=Possibly helpful 3=Moderately helpful 4=Important to learn more 5=Critical to learn more

Table 2. Questions asked for each of the topics

In addition, demographic questions are included in order to characterize the audience. We also use a set of filter questions in the beginning of the questionnaire in order to restrict the participants to the selected target audience. Two versions of the questionnaire (in Brazilian Portuguese and English) have been developed. The complete questionnaire is presented in Appendix A.

The questionnaire was made available on the web using the open-source survey tool LimeSurvey (http://www.limesurvey.org). To help reduce bias, the order of the question groups and order of topics within a group was automatically changed with each access. We piloted the survey within our research group before starting to gather the data.

3. Execution of the survey

We invited graduates via alumni mailing lists, computer science discussion groups and forums. The invitation included an explanation of the survey and indication to the website access to the questionnaire. Appendix D presents the mailing lists etc. to which the invitation was sent and Appendix C shows the invitation letter.

Due to an error during the survey construction, the item “Parsing and compiler design” was not included into the web version of the survey and, consequently, no data with respect to this topic was collected.

We collected data during the period from 20 October 2008 to 01 December 2008. During this period, we received completed survey responses from 48 participants working as software professionals. Among the participants are Computer Science graduates from public and private higher education institutions as shown in Figure 1, with a majority who graduated from UNIVALI and UFSC.

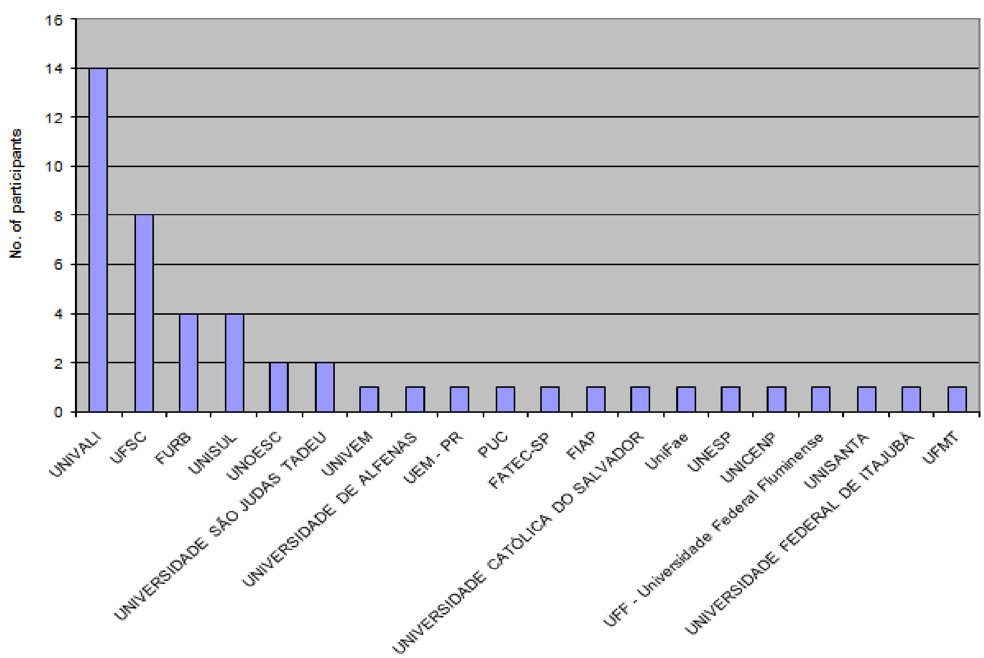

Figure 1. Distribution of participants per institution of graduation

In accordance to the defined period in which the participants graduated, we obtained responses from graduates from each year within the period, with a majority, who graduated in 2005 (Figure 2).

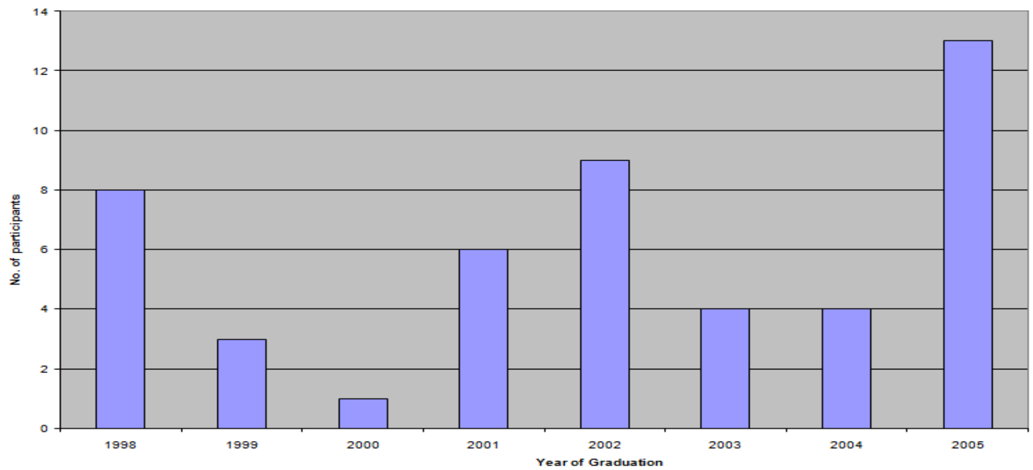

Figure 2. Distribution of participants per year of graduation

Respondents of the survey work on a variety of types of software, with a majority on management information or application software and fewer on real-time embedded or consumer software. Figure 3 shows the types of software in which participants have performed significant work over the last three years. The graph shows the number of participants per category.

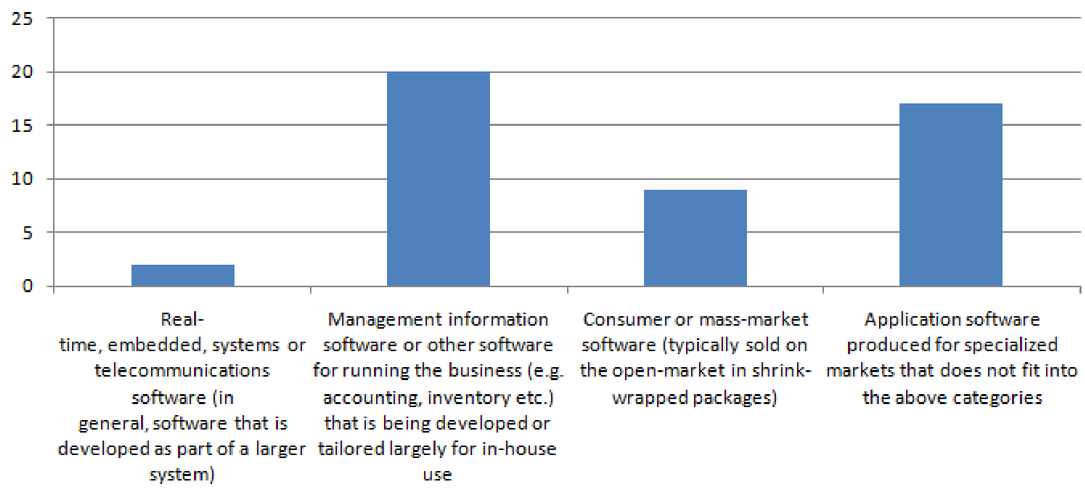

Figure 3. Distribution of participants per type of software worked with during the last 3 years

Respondents are involved in different activities in the software process, with a stronger focus on technical activities as shown in Table 3.

No. participants	none	less than 5%	5% - 10 %	10% - 25%	25% - 50%	50% - 75%	75% - 100%
Management or project management	4	6	9	14	5	7	3
Requirements analysis or specification	1	3	12	12	7	9	4

Software architecture and design	3	8	8	12	10	6	1
Working with source code	5	6	3	10	8	8	8
Testing software written by others	8	6	8	10	5	7	4
Installation, customer support, etc.	17	9	9	9	0	2	2

Table 3. Distribution of no. of participants per percentage of working time spent on activities during the last year

About 29% of the participants o four survey are women.

4. Data analysis and interpretation

The objective of this survey is to analyze (1) which CS topics are most important for a software professional; (2) if the amount of (university) education dedicated corresponds to this perceived importance corresponds; (3) if software professionals learn on the job what is necessary, and (4) if the importance of topics has changed in comparison to the (Lethbridge, 1999) and (Kitchenham et al., 2005) survey. In the following sections, we analyze each of those research questions.

4.1 Which CS topics are most important to software professionals?

We assessed importance based on the median value of the answers to Question 3: How useful has this specific material been to you in your career?

Eq1. Importance = Median of Q3

Table 4 shows the resulting ranking by of clusters of topics with equal levels of importance.

Ranking	Importance (Median of Q3)	Topic	Category
1	5	Data structures and algorithms	Software
	5	Databases	Software
	5	Specific programming languages	Software
	5	Object-oriented concepts and technology	Software engineering
	5	Team work skills	Related areas
	5	Second language	Related areas
2	4	Programming language theory	Software
	4	Operating systems	Software
	4	Real-time system design	Software
	4	Web-based programming	Software
	4	Project Management	Software engineering
	4	Requirements development	Software engineering
	4	Requirements management	Software engineering
	4	Software architecture	Software engineering
	4	Software design and patterns	Software engineering
	4	Software testing	Software engineering
	4	Software quality assurance	Software engineering
	4	Software reliability and fault tolerance	Software engineering
	4	Software cost/effort estimation	Software engineering
	4	Software process and process improvement (CMMI, etc.)	Software engineering
	4	Software engineering tools	Software engineering
	4	Management	Related areas
	4	Giving presentations to an audience	Related areas
	4	Leadership	Related areas
	4	Negotiation	Related areas
3	3	File management	Software

	3	Performance measurement and analysis	Software
	3	Computational complexity and algorithm analysis	Software
	3	Security and cryptography	Software
	3	Data transmission and networks	Software
	3	Parallel and distributed processing	Software
	3	Human computer interaction	Software
	3	Formal specification methods	Software engineering
	3	Software reviews and inspections	Software engineering
	3	Software configuration management	Software engineering
	3	Maintenance, reengineering and reverse engineering	Software engineering
	3	Software metrics	Software engineering
	3	Probability and statistics	Mathematics
	3	Mathematic logic	Mathematics
	3	Entrepreneurship	Related areas
	3	Technical writing	Related areas
	3	Legal / professionalism, ethics and society	Related areas
	3	Scientific methodology	Related areas
4	2	Artificial Intelligence	Software
	2	Computer system architecture	Hardware
	2	Linear algebra and matrices	Mathematics
	2	Graph theory	Mathematics
	2	Combinatorics	Mathematics
	2	Functions, relations and sets	Mathematics
	2	Operations research	Computational science
	2	Modeling and simulation	Computational science
	2	Economics	Related areas
	2	Accounting	Related areas
	2	Marketing	Related areas
	2	Psychology	Related areas
	2	Statistical process control	Related areas
5	1	Pattern recognition and image processing	Software
	1	Computer graphics	Software
	1	Digital electronics and digital logic	Hardware
	1	Microprocessor architecture	Hardware
	1	Digital signal processing	Hardware
	1	Differential and integral calculus	Mathematics
	1	Differential equations	Mathematics
	1	Numerical analysis	Mathematics
	1	Philosophy	Related areas
6	0	Analog electronics	Hardware
	0	Robotics	Hardware
	0	VLSI design	Hardware
	0	Physics	Related areas
	0	Chemistry	Related areas

Table 4. Ranking of topics by importance

The most interesting observations are:

a) The respondents consider data structures and algorithms, databases, specific programming languages, object-oriented concepts and technology, team work skills and second language (other than Brazilian Portuguese) to be the most important topics;
b) All software engineering topics are among the 25 most important topics;
c) Topics from related areas, including also management, leadership and giving presentations, are considered very important.
d) No mathematics, hardware or computational science topics are among the 25 most important topics;
e) Software topics which are considered less important are Artificial Intelligence, Pattern recognition and image processing and computer graphics.
f) Topics considered completely unimportant are: Analog electronics, Robotics, VLSI design, physics and chemistry.

Analyzing the importance of topics within each area separately Figure 4 – Figure 9 illustrate the ranking of topics within each category.

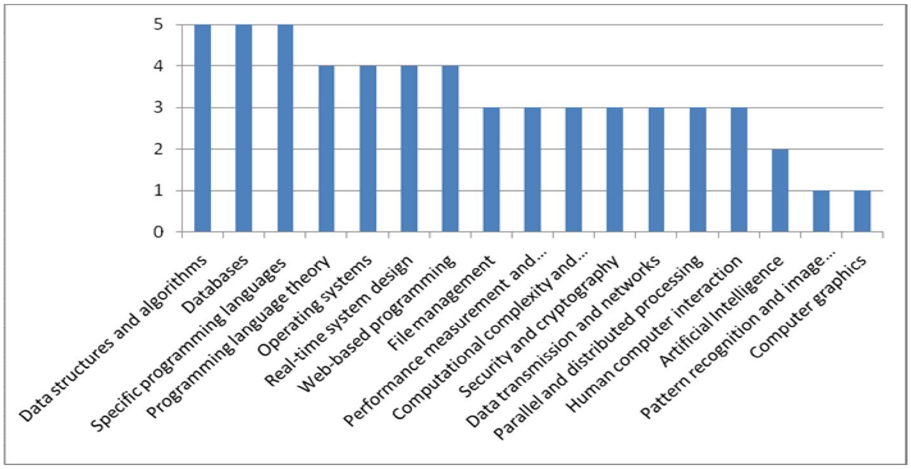

Figure 4. Ranking of importance within the category software

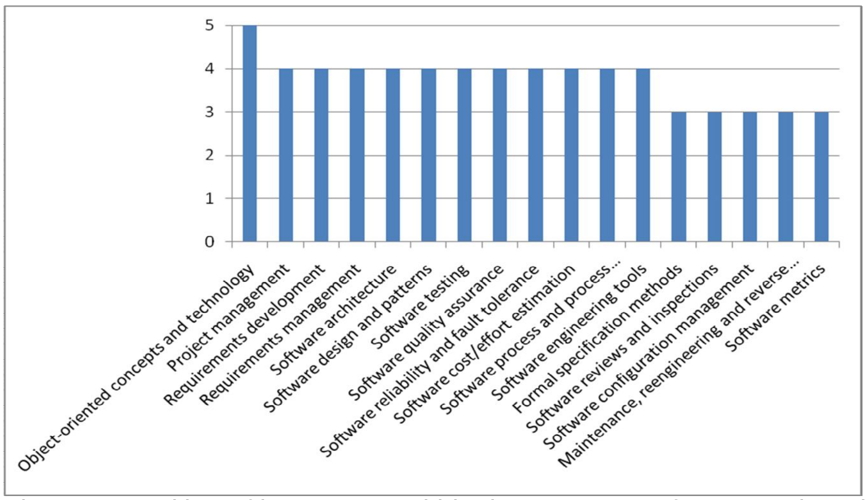

Figure 5. Ranking of importance within the category software engineering

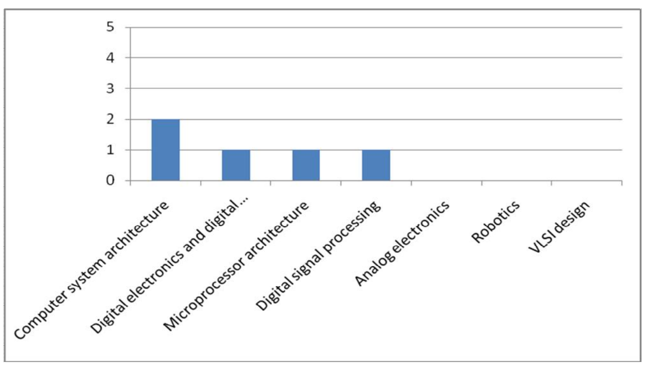

Figure 6. Ranking of importance within the category hardware

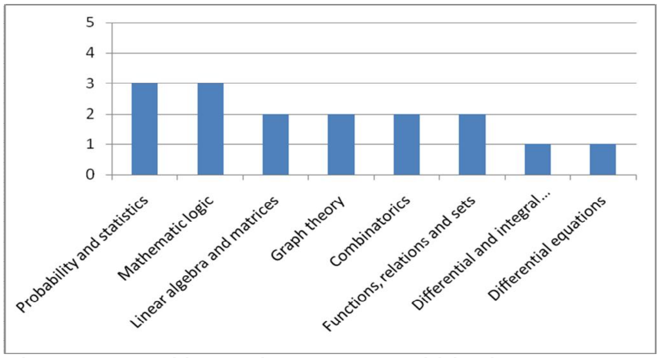

Figure 7. Ranking of importance within the category mathematics

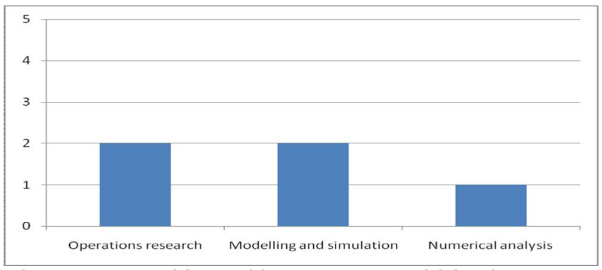

Figure 8. Ranking of importance within the category computational science

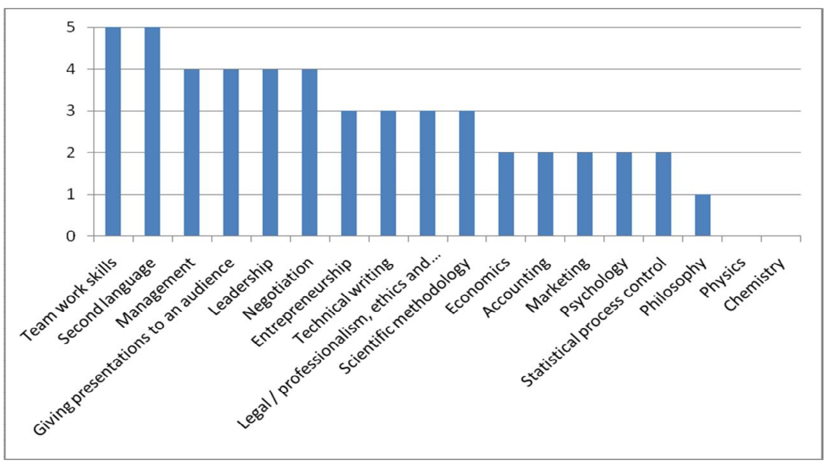

Figure 9. Ranking of importance within the category related areas

4.2 Does the amount of (university) education dedicated to a topic correspond to its importance?

Adapting the definition used in (Kitchenham et al., 2005), we define the knowledge gap to be the difference between importance and educational provision:

Eq.2. Knowledge gap = (median of answers to Q3) - (median of answers to Q1)

Thus, a negative value indicates a lack of education with respect to its perceived importance. Table 5 presents the topics ordered by the knowledge gap.

Topic	**Category**	**Knowledge Gap (Q1-Q3)**

Second language	Related area	-4
Giving presentations to an audience	Related area	-4
Negotiation	Related area	-4
Web-based programming	Software	-3
Project Management	Software Engineering	-3
Team work skills	Related area	-3
Management	Related area	-3
Leadership	Related area	-3
Data structures and algorithms	Software	-2
Databases	Software	-2
Specific programming languages	Software	-2
Real-time system design	Software	-2
Performance measurement and analysis	Software	-2
Object-oriented concepts and technology	Software Engineering	-2
Requirements development	Software Engineering	-2
Requirements management	Software Engineering	-2
Software architecture	Software Engineering	-2
Software design and patterns	Software Engineering	-2
Software testing	Software Engineering	-2
Software quality assurance	Software Engineering	-2
Software reliability and fault tolerance	Software Engineering	-2
Software cost/effort estimation	Software Engineering	-2
Software process and process improvement (CMMI, etc.)	Software Engineering	-2
Software engineering tools	Software Engineering	-2
Software reviews and inspections	Software Engineering	-2
Entrepreneurship	Related area	-2
Technical writing	Related area	-2
Economics	Related area	-2
Accounting	Related area	-2
Marketing	Related area	-2
Psychology	Related area	-2
Programming language theory	Software	-1
Operating systems	Software	-1
File management	Software	-1
Computational complexity and algorithm analysis	Software	-1
Security and cryptography	Software	-1
Data transmission and networks	Software	-1
Parallel and distributed processing	Software	-1
Human computer interaction	Software	-1
Formal specification methods	Software Engineering	-1
Software configuration management	Software Engineering	-1
Maintenance, reengineering and reverse engineering	Software Engineering	-1
Software metrics	Software Engineering	-1
Statistical process control	Related area	-1
Legal / professionalism, ethics and society	Related area	-1
Scientific methodology	Related area	-1
Philosophy	Related area	-1
Artificial Intelligence	Software	0
Pattern recognition and image processing	Software	0
Computer graphics	Software	0
Probability and statistics	Mathematics	0
Mathematic logic	Mathematics	0
Combinatorics	Mathematics	0
Computer system architecture	Hardware	0
Digital signal processing	Hardware	0
Robotics	Hardware	0
VLSI design	Hardware	0
Operations research	Computational science	0
Modeling and simulation	Computational science	0
Physics	Related area	0
Chemistry	Related area	0
Linear algebra and matrices	Mathematics	1
Graph theory	Mathematics	1
Functions, relations and sets	Mathematics	1
Differential equations	Mathematics	1
Digital electronics and digital logic	Hardware	1
Microprocessor architecture	Hardware	1
Analog electronics	Hardware	1

Numerical analysis	Computational science	1
Differential and integral calculus	Mathematics	2

Table 5. Knowledge gap per topic

Analyzing the knowledge gap, we can observe that several topics seem to under-represented in education in relation to their perceived importance, whereas a few seem to be over-represented in accordance with their importance. The main results are:

a) Related skills, such as, team work, second language, management, negotiation, giving presentations, etc. seem to be among the most under-represented topics;
b) With the exception of artificial intelligence, pattern recognition and image processing and computer graphics, all software topics seem to be under-represented, including, mainly, web-based programming.
c) All software engineering topics seem to be under-represented, especially, project management;
d) Some topics from the categories mathematics, hardware and computational science seem to be the only topics which are over-represented in accordance to their perceived importance.

Figure 10 – Figure 15 present the results on knowledge gap per category.

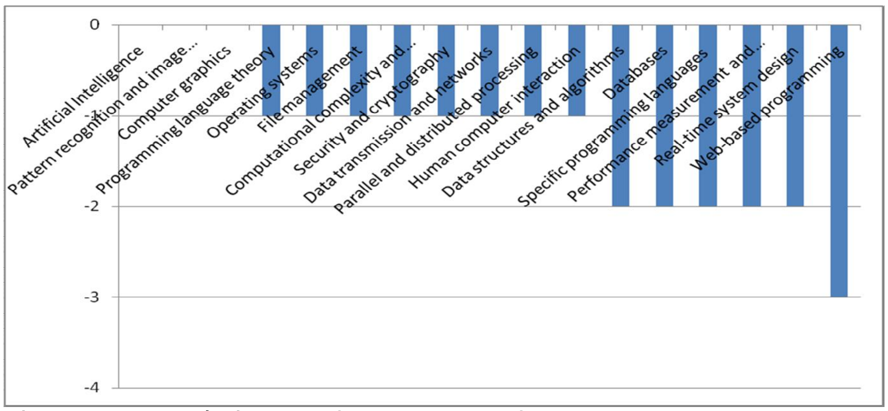

Figure 10. Knowledge gap for category software

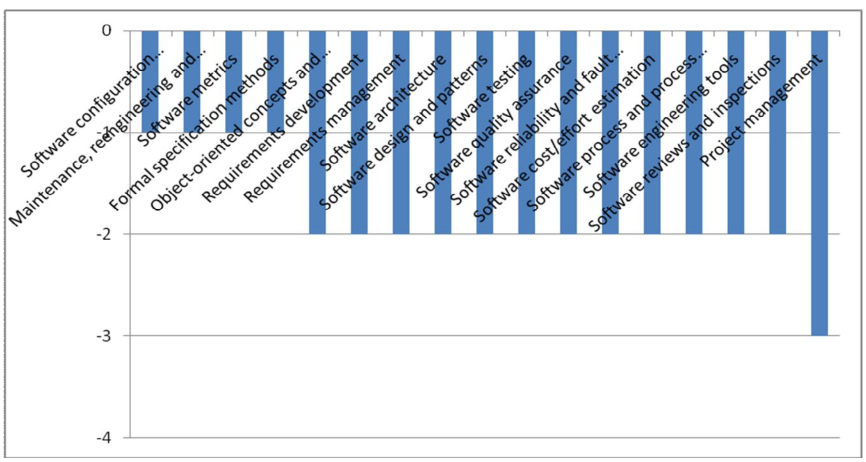

Figure 11. Knowledge gap for category software engineering

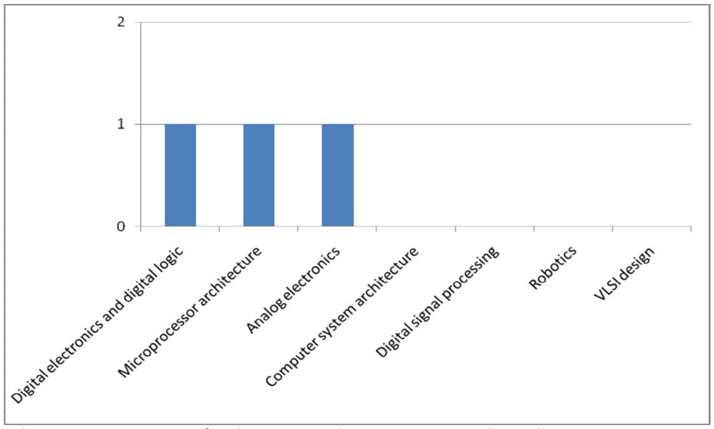

Figure 12. Knowledge gap for category hardware

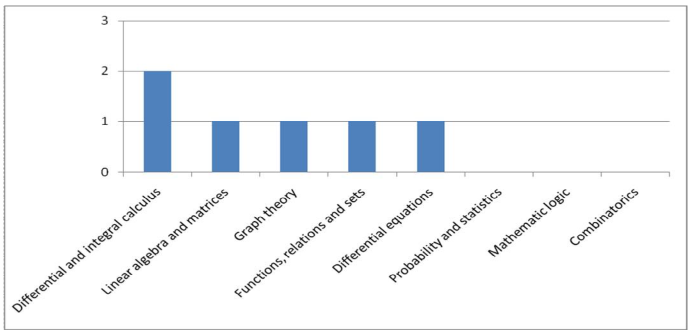

Figure 13. Knowledge gap for category mathematics

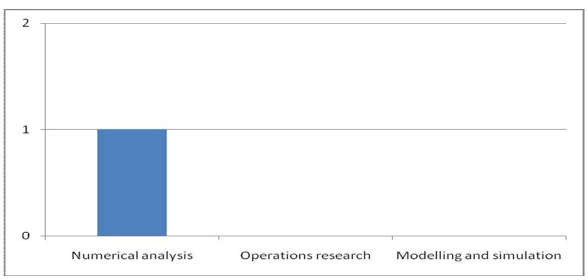

Figure 14. Knowledge gap for category computational science

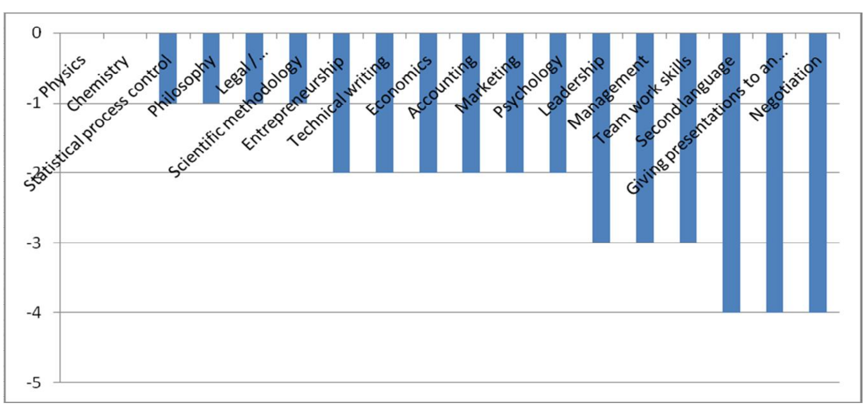

Figure 15. Knowledge gap for category related areas

4.3 Do software professionals learn on the job what is necessary?

In accordance to (Lethbridge, 1999), we also analyzed the amount of topics learned on the job (or forgotten since education) based on the difference between the responses to question 1 and 2.

Figure 16 – Figure 21 present the results in relation with the perceived importance and amount learned in education.

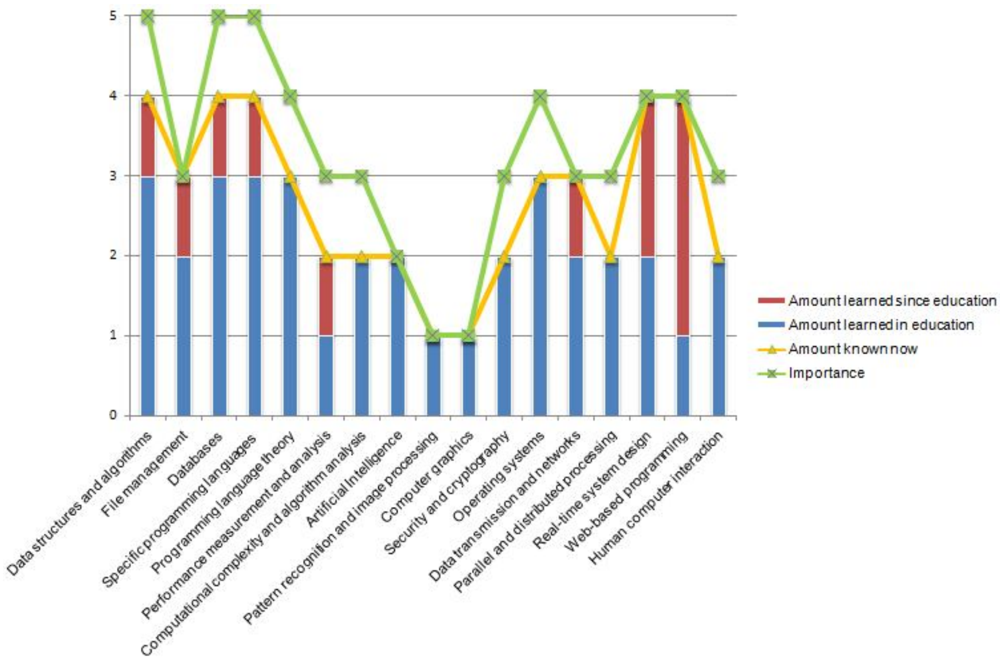

Figure 16. Results with respect to category: software

Regarding topics in the software category, we can observe that professionals learned more since graduation mainly with respect to web-based programming and real-time system design. We can also observe that considering the perceived importance of several topics, such as, security and cryptography, professionals did not learn sufficient neither in during university education nor later on, pointing out a still existing deficiency with respect to those topics.

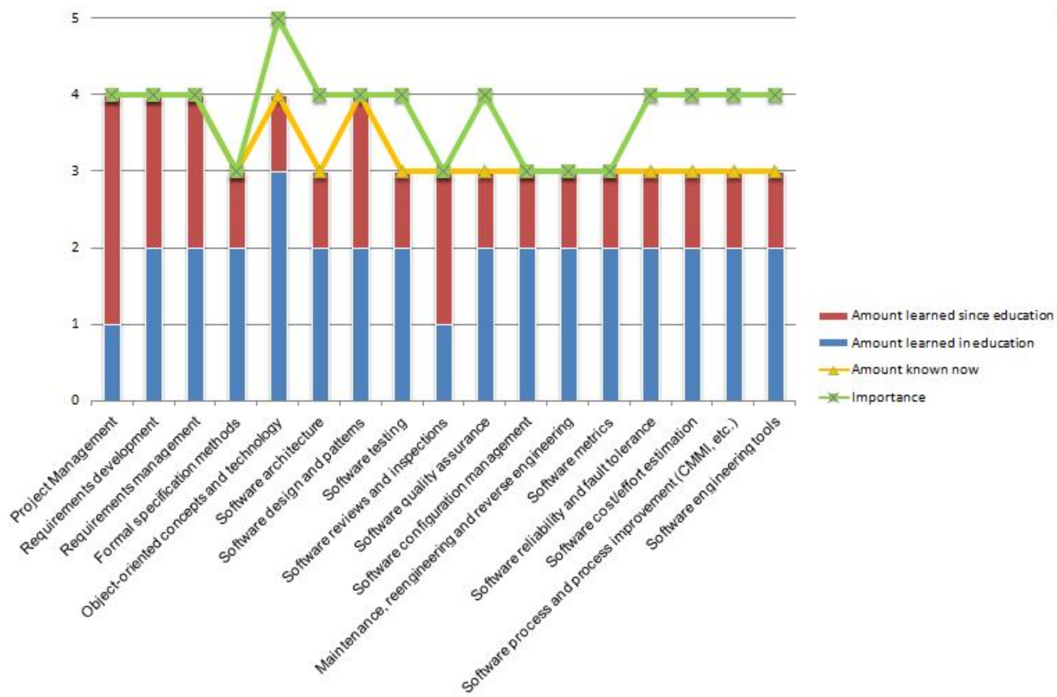

Figure 17. Results with respect to category: software engineering

Considering topics in the software engineering category, professionals indicated that they learned more in each topic since education. Especially, the topic project management is emphasized as one of the topics which professionals learned most since education. Yet, the amount known now still does not reach the level of perceived importance with respect to several Software Engineering topics.

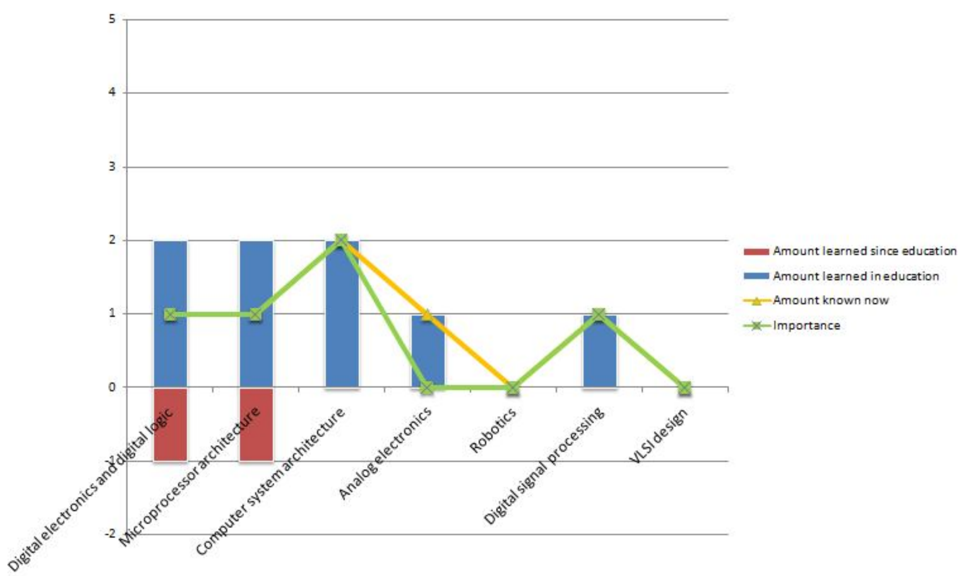

Figure 18. Results with respect to category: hardware

In this category, we can observe a different pattern. Regarding those topics, respondents even indicated a loss of knowledge since education with regard to digital electronics and digital logic and microprocessor architecture. Several of those topics also seem to be over-represented in Computer Science education, as their importance is perceived lower than the considered amount learned.

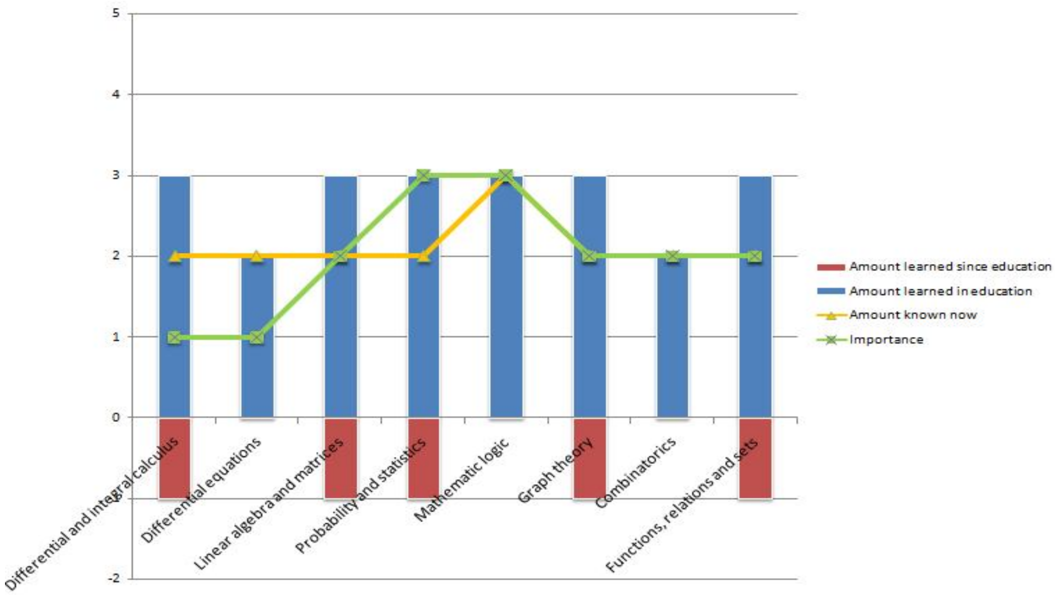

Figure 19. Results with respect to category: mathematics

Although considered important, mathematics seems to be also over-represented in computer science education. Exceptions are probability and statistics, mathematic logic and combinatorics, for which the degree of perceived importance and amount learned in education correspond. This is also stressed by the observation that respondents indicated a loss of knowledge to most of the mathematics topics since graduation.

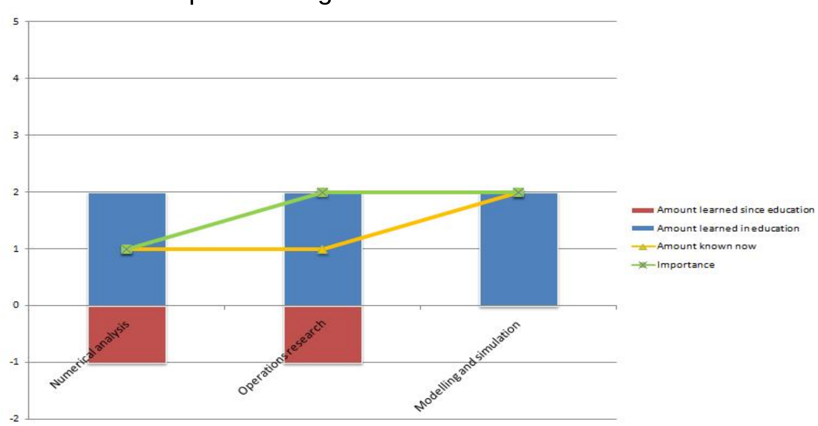

Figure 20. Results with respect to category: computational science

A similar pattern can be observed for topics in the category computational science. Being considered of low importance, the amount learned in education matches the perceived importance or as in case of numerical analysis may be over-represented.

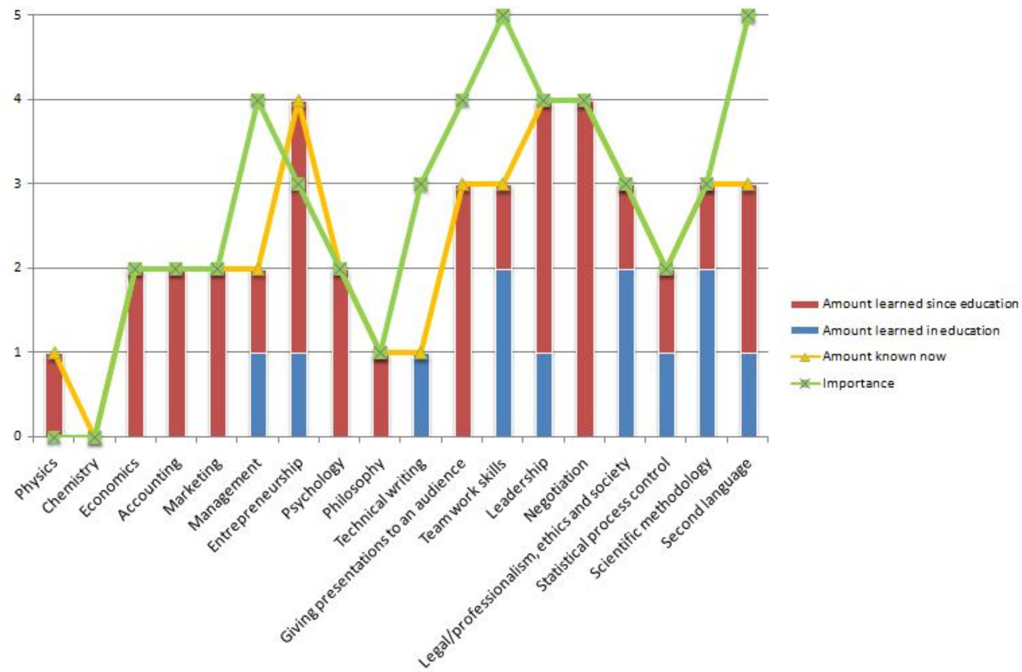

Figure 21. Results with respect to category: related areas

One of the most interesting results is the perceived importance of those related topics and the lack of amount learned in education. Extreme examples are second language (other than Brazilian Portuguese) which seems to be essential for a software professional in Brazil, as well as negotiation, leadership, giving presentations to an audience and management. Topics from this area are also among those for which the respondents learned a large amount since education – indicating a need for additional education as those topics do not seem to be adequately represented in computer science education. Yet, the amount of knowledge with respect to several topics, such as, management, technical writing, giving presentations and team work skills still does not reach their perceived degree of importance.

4.4 Did the importance of topics change in comparison to the (Lethbridge, 1999) and (Kitchenham et al., 2005) survey?

In order to analyze this question, we compare the ranking of importance of the topics in our survey with the ranking of the importance of topics in the Lethbridge and Kitchenham survey. Yet, it is important to point out that the importance has been assessed differently in each of the surveys. Here, as described before, we assess importance on the median value for question 3.

In the Lethbridge study, several ways of assessing the importance we applied (Lethbridge, 1999). Yet, the presentation of the full list of topics ranked (see Table 7) is provided by computing the average (mean value) of questions 3 and 4.
On the other side, in the Kitchenham study, importance is assessed on the proportion of subjects scoring three (indicating at least moderate usefulness) or more for question 3 (see Table 8).

Another issue to be considered is the sets of topics analyzed in each of the studies. Although, strongly based on the Lethbridge study we also modified the set (as described in Appendix B) in accordance to our needs and advances in the field of CS – including e.g., web based programming.

On the other hand, the scope of the Kitchenham survey is focused on software and software engineering topics, not covering, e.g., related areas or hardware topics.

Table 6 show a the topics ranked by importance based on our survey, Table 7 shows the ranking by importance of the Lethbridge study and Table 8 presents the ranking by importance of the Kitchenham study.

Ranking	Topic
1	Data structures and algorithms
	Databases
	Specific programming languages
	Object-oriented concepts and technology
	Team work skills
	Second language
2	Programming language theory
	Operating systems
	Real-time system design
	Web-based programming
	Project Management
	Requirements development
	Requirements management
	Software architecture
	Software design and patterns
	Software testing
	Software quality assurance
	Software reliability and fault tolerance
	Software cost/effort estimation
	Software process and process improvement (CMMI, etc.)
	Software engineering tools
	Management
	Giving presentations to an audience
	Leadership
	Negotiation
3	File management
	Performance measurement and analysis
	Computational complexity and algorithm analysis
	Security and cryptography
	Data transmission and networks
	Parallel and distributed processing
	Human computer interaction
	Formal specification methods
	Software reviews and inspections
	Software configuration management
	Maintenance, reengineering and reverse engineering
	Software metrics
	Probability and statistics
	Mathematic logic
	Entrepreneurship
	Technical writing
	Legal / professionalism, ethics and society
	Scientific methodology
4	Artificial Intelligence
	Computer system architecture
	Linear algebra and matrices
	Graph theory
	Combinatorics
	Functions, relations and sets
	Operations research
	Modeling and simulation
	Economics
	Accounting
	Marketing
	Psychology
	Statistical process control
5	Pattern recognition and image processing
	Computer graphics
	Digital electronics and digital logic

	Microprocessor architecture
	Digital signal processing
	Differential and integral calculus
	Differential equations
	Numerical analysis
	Philosophy
6	Analog electronics
	Robotics
	VLSI design
	Physics
	Chemistry

Table 6. Importance ranking of our study

	Ranking based on (Lethbridge, 1999)	Overall importance (Average of Q3 and Q4)
1	Specific Programming Languages	3.8
2	Data Structures	3.6
3	Software Design and Patterns	3.5
4	Software Architecture	3.4
	Requirements Gathering & Analysis	3.4
5	HCI / User Interfaces	3.3
	Object Oriented Concepts & Tech.	3.3
	Ethics and Professionalism	3.3
	Analysis and Design Methods	3.3
	Giving Presentations to an Audience	3.3
	Project Management	3.3
6	Testing, Verif. & Quality Assurance	3.2
	Design of Algorithms	3.2
7	Technical Writing	3.1
	Operating Systems	3.1
	Databases	3.1
8	Leadership	3.0
	Configuration and Release Management	3.0
	Data Transmission and Networks	3.0
9	Management	2.9
10	File Management	2.8
	Software Reliability & Fault Tolerance	2.8
	Systems Programming	2.8
	Network Architecture & Data Trans.	2.8
	Negotiation	2.8
11	Performance Measurement & Analysis	2.7
	Maintenance, Reeng. and Rev. Engg.	2.7
	Programming Language Theory	2.7
	Computer System Architecture	2.7
12	Comput. Complexity & Algor. Analysis	2.6
	Probability and Statistics	2.6
	Software Cost Estimation	2.6
13	Real-Time System Design	2.5
	Information Retrieval	2.5
	Software Metrics	2.5
14	Formal Languages	2.4
	Formal Specification Methods	2.4
	Process Standards CMM / ISO 9000	2.4
	Predicate Logic	2.4
	Entrepreneurship	2.4
15	Simulation	2.3
	Security and Cryptography	2.3
	Telephony and Telecommunications	2.3
	Parsing and Compiler Design	2.3
	Parallel and Distributed Processing	2.3
16	Microprocessor Architecture	2.2
	Digital Electronics & Digital Logic	2.2
	Set Theory	2.2
17	Automata theory	2.1
	Data Acquisition	2.1
	Marketing	2.1
	Comput. Methods for Numeric Probs.	2.1
	Psychology	2.1
	Accounting	2.1

	Economics	2.1
18	Linear Algebra and Matrices	2.0
	Philosophy	2.0
	Second Language Other than English	2.0
	Physics	2.0
	Information Theory	2.0
	Graph Theory	2.0
19	Queuing Theory	1.9
	Computer Graphics	1.9
20	Digital Signal Processing	1.8
21	Control Theory	1.7
22	Pattern Recognition and Image Proc.	1.6
	Differential and Integral Calculus	1.6
	Combinatorics	1.6
23	Artificial Intelligence	1.5
	Analog Electronics	1.5
24	Laplace and Fourier Transforms	1.3
	Differential Equations	1.3
	Chemistry	1.3
	Robotics	1.3
25	VLSI	1.2

Table 7. Importance ranking of the Lethbridge study (Lethbridge, 1999)

	Ranking based on (Kitchenham et al., 2005)	**Importance (Q3) %**
1	Human computer interaction/user interfaces	93.33
2	Project management	83.33
3	Databases	76.67
4	Operating systems	75.86
5	Requirements gathering and analysis	73.33
	Specific programming languages	73.33
	Data structures	73.33
6	Software architecture	70.00
	Data transmission and networks	70.00
7	Analysis and design methods	66.67
	Testing, verification and quality assurance	66.67
8	Software design practices	65.52
9	Web-based programming	60.71
10	Object oriented concepts and terminology	60.00
11	Systems programming	58.62
12	Information retrieval	57.14
13	Software design patterns	51.72
14	Configuration and release management	48.28
15	File management	46.67
16	Security and cryptography	46.43
17	Design of algorithms	44.83
18	Performance measurement and analysis	43.33
19	Computer graphics	39.29
20	Programming language theory	37.93
21	Multi-media	34.48
22	Maintenance, reengineering and reverse engineering	33.33
	Formal specification methods	33.33
23	Software cost estimation	28.57
24	Software reliability and fault tolerance	27.59
	Parallel and distributed processing	27.59
25	Real-time system design	23.33
26	Computational complexity and algorithm analysis	20.69
	Parsing and compiler design	20.69
27	Process standards (CMM/ISO 9000 etc.)	17.24
28	Simulation	11.11
29	Computational methods for numerical problems	10.71
30	Artificial intelligence	10.00
31	Software metrics	6.90
	Pattern recognition and image processing	6.90

Table 8. Importance ranking of the Kitchenham study (Kitchenham et al.,2005)

Comparing the results of our study regarding the ranking of the topics by importance in relation to those two earlier studies, the main observations are:

a) Topics considered very important in our study, including, data structures and algorithms, databases, specific programming languages and object-oriented concepts and technology are also among the most important topics of both other surveys.
b) Team work skill – a very important topic in our survey – has not considered in any other of the other two surveys.
c) Second language (in our case other than Brazilian Portuguese) is considered a very important topic, whereas this topic is considered less important in the Lethbridge survey, where this is understood as a second language other than English.
d) In comparison – especially with the Kitchenham survey – we can observe an increase in the importance of software engineering topics.
e) A topic considered very important in the Lethbridge and the most important topic in the Kitchenham study – human computer interaction – has been considered of medium importance in our survey.
f) In comparison to the Lethbridge survey, topics such as ethics and professionalism, and data transmission and networks have been considered less important in our survey.
g) Our survey results confirm the high importance perceived of topics from related areas, specifically related to skills such as giving presentations, leadership, management etc.
h) Our survey also confirmed a lower importance perceived on topics from the categories of hardware and mathematics.

5. Survey weaknesses and limitations

We identified various weaknesses and limitations of our survey, which may be a threat to the validity of the results.

One of the major weaknesses in this survey is the low response rate. This means that the results have to be interpreted with caution as they may not present a representative result for the target population. Especially as the participation was voluntary, the respondents may share similarities and interests, not generally present. We also identified the need to obtain more information on the context of the respondents, mainly, on the type and application domain of the software systems on which they work in order to analyze the results in the context of specific industry sectors.

Another issue regarding the generalization of the results may be our propositional limitation in our research to Brazilian software professionals. Most probably there exist international or even regional differences also due to the kind of predominant software industry and application domain present. Therefore, also the comparison of the results of our survey with the Lethbridge and Kitchenham studies has to be done with caution. This comparison has also to be done carefully, as our survey is not an exact replication of neither the Lethbridge nor the Kitchenham study – we modified (partly) the research questions, target audience, questionnaire and analysis.

Another weakness may be the construction of the survey. One problem was that due to an error the topic Parsing and Compiler Design was not included in our survey. Another issue may be a lack of explanation for each of the topics being considered. For example, software professionals may not have a correct and/ or consistent understanding of the topics, such as, e.g. the difference between requirements development and requirements management, resulting unintentionally in the collection of erroneous answers.

In order to reduce the provision of intentional untruthful responses, the provision of personal information, such as, name and email was voluntary.

6. Discussion

It is important to point out that the focus of our research is on the relevance of topics of Computer Science undergraduate courses for software professionals. In our research, we do not cover other courses, such as, Information Systems or Computer Engineering, for which, certainly, another set of topics may be relevant.

Among the main observations of the survey are:

- The perceived importance of teaching on one side fundamental concepts, such as, data structures and algorithms and object-oriented concepts and technology and, on the other side, specific programming languages.
- The outstanding importance and need to teach required skills, such as, management, team work, leadership, giving presentations, etc. This is an interesting result, especially, as the teaching of those skills often is not emphasized in Computer Science courses.
- A specific result of this survey is also the outstanding importance and existing knowledge gap considering a second language (other than Brazilian Portuguese). This result may indicate the essential importance of the English language in the software domain for non-native English speaker, probably as a pre-requisite in order to obtain access to literature and information available only in English as a basis for the acquisition of any of relevant topics.
- An increased perceived importance of Software Engineering topics which seems not to be sufficiently attended by university education.
- Hardware, mathematics and computational science topics seem to become less important to Computer Science graduates, although certain topics, such as, e.g., mathematical logic, are being perceived as of medium importance. Yet, often these topics are extensively taught in Computer Science courses. Considering the perception of respondents that they tend to forget about those topics later on, are an additional indication that those topics are less important and may be over-represented in a Computer Science curriculum.
- The survey also shows that a considerable amount on those topics is learned after graduation. This also stresses the importance of professional trainings and learning on-the-job. Section 4.3 shows that the greatest amount of learning since education occurred with respect to web-based programming, project management and giving presentations, leadership, negotiation and entrepreneurship. Yet, the amount currently known correlates well with their perceived importance, indicating that it currently is possible for software practitioners' to acquire the necessary competence. On the other hand, we identified various topics for which the currently known amount lies under the importance of the topic. Therefore, it may be interesting on the short term to focus professional training offerings specifically to those topics as well as consider them more strongly in Computer Science curriculums on the long term.

The obtained results of our survey confirm in many points the observations of the Lethbridge survey. In general, the importance rating has been quite similar, with few exceptions, which are emphasized even more in the importance ranking of the Kitchenham survey, including, principally, human computer interaction – being considered much more important in those surveys. On the other hand, the topic second language is considered of much less importance in the Lethbridge survey (most probably as the survey has been conduct in English speaking countries, reducing the need for a second language) and in the Kitchenham survey a much less importance for real-time system design. Comparing our results to the ones in the Lethbridge survey, we also can confirm the indication of less importance perceived for topics in mathematics, hardware and computational science. In general, our results also continue to show the same observations with regard to the amount learned in education.

Thus, the results of the survey may indicate that currently important topics are not adequately taught, whereas some less important topics may be over-represented in Computer Science curriculums. Therefore, we compare the survey results (considering the perceived importance) with the summary of the Computer Science body of knowledge as presented as part of the curricular guidelines for undergraduate programs in computing (The Joint Task Force on Computing Curricula, 2001). Table 9 shows the topics ordered by importance mapped to topics of the Computer Science body of knowledge, indicating also the number of suggested minimum core hours to be dedicated to each topic.

		IEEE/ACM CS Curriculum guidelines (XXX)	
Ranked by Importance	**Topic**	**Computer science body of knowledge**	**Number of suggested min. core hours***
1	Data structures and algorithms	PF. Programming Fundamentals	38
	Databases	IM. Information Management	10
	Specific programming languages		
	Object-oriented concepts and technology	PL6. Object-oriented programming	10
	Team work skills		
	Second language		
2	Programming language theory	PL1. Overview of programming languages PL2. Virtual machines PL3. Introduction to language translation PL4. Declarations and types PL5. Abstraction mechanisms PL7. Functional programming PL8. Language translation systems PL9. Type systems PL10. Programming language semantics PL11. Programming language design	11
	Operating systems	OS. Operating Systems	18
	Real-time system design		
	Web-based programming	NC1. Introduction to net-centric computing NC4. The web as an example of client-server computing NC5. Building web applications NC6. Network management NC7. Compression and decompression NC8. Multimedia data technologies NC9. Wireless and mobile computing	5
	Project Management	SE8. Software project management	3
	Requirements development	SE5. Software requirements and specifications	4
	Requirements management		
	Software architecture Software design and patterns	SE1. Software design	8
	Software testing	SE6. Software validation	3
	Software quality assurance		
	Software reliability and fault tolerance	SE11. Software reliability	-
	Software cost/effort estimation		
	Software process and process improvement (CMMI, etc.)	SE4. Software processes	2
	Software engineering tools	SE3. Software tools and environments	3
	Management		
	Giving presentations to an audience		
	Leadership		
	Negotiation		
3	File management	*as part of OS. Operating systems*	
	Performance measurement and analysis		
	Computational complexity and algorithm analysis	AL. Algorithms and Complexity	31
	Security and cryptography	NC3 Network security	3
	Data transmission and networks	NC2. Communication and networking	7
	Parallel and distributed processing	AR7. Multiprocessing and alternative architectures	3
	Human computer interaction	HC. Human-Computer Interaction	8
	Formal specification methods	SE10. Formal methods	-
	Software reviews and inspections	*As part of SE6. Software validation*	
	Software configuration management		
	Maintenance, reengineering and reverse engineering	SE7. Software evolution	3
	Software metrics		
	Probability and statistics	DS6. Discrete probability	6
	Mathematic logic	DS2. Basic logic	10
		DS3. Proof techniques	12
	Entrepreneurship		
	Technical writing		
	Legal / professionalism, ethics and society	SP. Social and Professional Issues	16
	Scientific methodology		

4	Artificial Intelligence	IS. Intelligent Systems	10
	Computer system architecture	AR2. Machine level representation of data AR3. Assembly level machine organization AR4. Memory system organization and architecture AR5. Interfacing and communication AR6. Functional organization	27
	Linear algebra and matrices		
	Graph theory	DS5. Graphs and trees	4
	Combinatorics	DS4. Basics of counting	5
	Functions, relations and sets	DS1. Functions, relations, and sets	6
	Operations research	CN2. Operations research	-
	Modeling and simulation	CN3. Modeling and simulation	-
	Economics	SP9. Economic issues in computing	-
	Accounting		
	Marketing		
	Psychology		
	Statistical process control		
5	Pattern recognition and image processing		
	Computer graphics	GV. Graphics and Visual Computing	3
	Digital electronics and digital logic	AR1. Digital logic and digital systems	6
	Microprocessor architecture	*as part of AR. Architecture and Organization*	
	Digital signal processing		
	Differential and integral calculus		
	Differential equations		
	Numerical analysis	CN1. Numerical analysis	-
	Philosophy	SP10. Philosophical frameworks	-
6	Analog electronics		
	Robotics		
	VLSI design		
	Physics		
	Chemistry		

* *Note: The numbers represent the minimum number of hours required to cover this material in a lecture format as suggested by (The Joint Task Force on Computing Curricula IEEE CS/ACM, 2001). No allocation of core hours indicates advanced course content for which no min. number of hours is specified.*

Table 9. Mapping of importance of topics with Computer Science body of knowledge

Overall, the importance ranking based on the survey results and the consideration of the respective topics in the Computer Science body of knowledge guidelines show partially a good correspondence. The most highly ranked survey topics are well represented, with exception of specific programming languages and soft skills, such as, team work and second language. In general, relevant soft skills seem to be under-represented as explicit cited topics. Yet, as those skills can be understood to be orthogonal to the specific Computer Science topics a stronger integration of instruction dedicated to those topics within the existing topics/courses may be possible easily.

The comparison also confirms a lack of dedication to certain software engineering topics considered important.

On the other side, it also confirms an existing strong focus on hardware related topics, which may be over-represented. As these results confirm also findings in other studies, it may be interesting, to revise current Computer Science curriculums in order to fulfill more adequately the needs of software practitioners' today.

7. Conclusion

In this report, we presented the survey on the relevance of Computer Science curriculum. We principally identified the perceived importance and the existing knowledge gap after education in order to provide a feedback for the development of undergraduate curriculum and professional training development. In general, we confirmed the results of earlier studies. Yet, in order to obtain a more representative result and in order to accompany changing needs from industry, such surveys should be conducted regularly and on a larger scale.

Acknowledgements

We would like to thank Timothy Lethbridge for providing us with material used in the 1998 survey. Thanks also to our colleagues at the UNIVALI for reviewing the questionnaire. We would also like to thank the research group CYCLOPS at the Federal University of Santa Catarina (UFSC) for hosting the survey system.

This work was supported by the CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico), an entity of the Brazilian government focused on scientific and technological development and by the Universidade do Vale do Itajaí - UNIVALI.

References

CMMI Product Team. CMMI for Development, Version 1.2. Technical Report CMU/SEI-2006-TR-008, Carnegie Mellon University/Software Engineering Institute, 2006.

IEEE Computer Society. SWEBOK - Guide to the Software Engineering Body. of Knowledge, 2004.

M. Kasunic. Designing an Effective Survey. Handbook CMU/SEI-2005-HB-004, Carnegie Mellon University/Software Engineering Institute, September 2005.

B. Kitchenham, D. Budgen, P. Brereton, P. Woodall. An investigation of software engineering curricula. The Journal of Systems and Software 74 (2005) 325–335.

T. C. Lethbridge. The Relevance of Education to Software Practitioners: Data from the 1998 Survey. Technical Report TR-99-06, University of Ottawa - Computer Science, Canada, 1999.

T. C. Lethbridge. What Knowledge Is Important to a Software Professional? IEEE Computer, May 2000.

Ministery of Education . Curriculum Guidelines for Courses in the Area of Computer Science and Informatics (in Portugues). Brazil, 1999.

The Joint Task Force on Computing Curricula IEEE CS/ACM. Computing Curricula 2001 - Computer Science. Final Report, 2001.

The Joint Task Force on Computing Curricula IEEE CS/ACM. Software Engineering 2004 - Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, 2004.

The Joint Task Force for Computing Curricula ACM/AIS/IEEE-CS. Computing Curricula 2005 - The Overview Report, 2005.

APPENDIX A. Questionnaire

The Software Quality and Productivity Laboratory of the UNIVALI/São José conducts a survey to gain a better understanding of the relevance of topics to computer science education. The survey is run among graduates from the computer science course at the UNIVALI/São Jose, who graduated between 1998 and 2005.

This questionnaire is designed to discover what aspects of your educational background have been useful to you in your career. The results of the survey will be used to help improve curricula and to direct our research on software engineering education. All the information you provide will be kept confidential. In particular, we have no intention of judging you as a person - we are merely interested in learning about the relevance of certain topics to your work.

If you don't understand a question, you may leave it blank or state an assumption. There is a field at the end where you can type any additional comments you may have. If you have any questions at any time regarding this activity, please contact Gisele at the LQPS – *Laboratório de Qualidade e Produtividade de Software/UNIVALI* via e-mail (giselepsn@gmail.com) or phone +55 (48) 3281 1505.

Your responses to this questionnaire are very important to us. They will help us develop the most complete and accurate understanding possible of education needs. The questionnaire should take no more than half an hour to complete.

Thank you for taking the time to complete this questionnaire.

Prof. Dr. rer. nat. Christiane Gresse von Wangenheim

Universidade do Vale do Itajaí - UNIVALI - Campus São José
LQPS - Laboratório de Qualidade e Produtividade de Software
Rodovia SC 407, Km. 4 - Bloco 1, Sala 401
88122-000 - Sao José - Santa Catarina – Brasil
http://www.univali.br

Filter questions (If any of these questions is answered with no – terminate questionnaire)

Did you graduate from a Computer Science undergraduate course? [yes, no]

Did you graduate in the period of 1998 to 2005? [yes, no]

Do you currently work as a software professional? [yes, no]

Substance questions

	1. How much did you learn about this during the computer science course at UNIVALI? 0 =Learned nothing at all 1 =Became vaguely familiar 2 =Learned the basics 3 =Became functional (moderate working knowledge) 4= Learned a lot 5 =Learned in depth, became expert (learned almost	2. What is your current knowledge about this, considering what you have learned on the job as well as forgotten since you graduated? 0=Know nothing 1=Am vaguely familiar 2=Know the basics 3=Am functional; (moderate working knowledge) 4=Know a lot 5=Know in depth/ am expert (know almost	3. How useful has this specific material been to you in your career? 0=Completely useless 1=Almost never useful 2=Occasionally useful 3=Moderately useful, but perhaps only in certain activities 4=Very useful 5=Essential	4. How useful would it be (or have been) to learn more about this (e.g. additional courses)? 0=Pointless learning more 1=Very unlikely to be useful 2=Possibly helpful 3=Moderately helpful 4=Important to learn more 5=Critical to learn more

	everything)	everything)		
Software Data structures and algorithms File management Databases Specific programming languages Programming language theory Parsing and compiler design Performance measurement and analysis Computational complexity and algorithm analysis Artificial Intelligence Pattern recognition and image processing Computer graphics Human computer interaction/user interfaces Security and cryptography Operating systems Data transmission and networks Parallel and distributed processing Real-time system design Web-based programming				
Software engineering Project management Requirements development Requirements management Formal specification methods Object-oriented concepts and technology Software architecture Software design and patterns Software testing Software reviews and inspections Software quality assurance Software configuration management Maintenance, reengineering and reverse engineering Software metrics Software reliability and fault tolerance Software cost/effort estimation Software process and process improvement (CMMI, etc.) Software engineering tools				
Hardware Digital electronics and digital logic Microprocessor architecture Computer system architecture Analog electronics Robotics Digital signal processing VLSI design				
Mathematics Differential and integral calculus Differential equations Linear algebra and matrices Probability and statistics Mathematic logic Graph theory Combinatorics Functions, relations and sets				
Computational science Numerical analysis Operations research Modeling and simulation				
Related areas Physics Chemistry Economics Accounting Marketing Management Entrepreneurship Psychology Philosophy Technical writing Giving presentations to an audience Team work skills Leadership Negotiation Legal/professionalism/ethics and society Statistical process control Scientific methodology Second language				

Demographic questions

These will help us find out if people with different types of background have different needs.

Gender: [male/female]

Institution where you graduated in Computer Science [________________________]

Year of graduation from the Computer Science: [2005, … 1998]

Country where you work: [..,Brazil, …]

How many years do you have worked professionally in the field of computer science/software development? [number]

Please indicate the approximate percentage of your total working time that you have spent on the following activities during the last year.

a) Management or project management: [none, less than 5%, 5%-10%, 10%-25%, 25%-50%, 50%-75%, 75%-100%]
b) Requirements analysis or specification: [none, less than 5%, 5%-10%, 10%-25%, 25%-50%, 50%-75%, 75%-100%]
c) Software architecture and design: [none, less than 5%, 5%-10%, 10%-25%, 25%-50%, 50%-75%, 75%-100%]
d) Working with source code (writing code, understanding code etc.): [none, less than 5%, 5%-10%, 10%-25%, 25%-50%, 50%-75%, 75%-100%]
e) Testing software written by others: [none, less than 5%, 5%-10%, 10%-25%, 25%-50%, 50%-75%, 75%-100%]
f) Installation, customer support etc.: [none, less than 5%, 5%-10%, 10%-25%, 25%-50%, 50%-75%, 75%-100%]

On which of the following types of software have you performed significant work over the last three years? You may select more than one category:

a) Real-time, embedded, systems or telecommunications software (in general, software that is developed as part of a larger system).
b) Management information software or other software for running the business (e.g. accounting, inventory etc.) that is being developed or tailored largely for in-house use.
c) Consumer or mass-market software (typically sold on the open-market in shrink-wrapped packages).
d) Application software produced for specialized markets that does not fit into the above categories.

Comments: [free text]

Identification (optional)

Name:

Email:

Once again, thank-you for participating! You can contact us for the results in 2009.

APPENDIX B. Changes made to the questionnaire with respect to (Lethbridge, 1999)

Software
Data structures and algorithms (design of algorithms)
File management
Databases
Specific programming languages
Programming language theory
Parsing and compiler design
Performance measurement and analysis
Computational complexity and algorithm analysis
Artificial Intelligence
Pattern recognition and image processing
Computer graphics
Human Computer Interaction/user interfaces
~~Information retrieval~~
Security and cryptography
Operating systems
~~Systems programming~~
Data transmission and networks
Parallel and distributed processing
Real-time system design
Web-based programming

Software engineering
Project management
Requirements development ~~Requirements gathering and analysis~~
Requirements management
Formal specification methods
Object-oriented concepts and technology
Software architecture
Software design and patterns
~~Analysis and design methods~~
~~Testing, verification and quality assurance~~
Software Testing
Software reviews and inspections
Software quality assurance
Software configuration management ~~Configuration and release management~~
Maintenance, reengineering and reverse engineering
Software metrics
Software reliability and fault tolerance
Software cost/effort estimation ~~Software cost estimation~~
Software process and process improvement (CMMI, etc.) ~~Process standards (CMMI/ISO 9000 etc)~~
Software engineering tools

Hardware
Digital electronics and digital logic
Microprocessor architecture
Computer System architecture
~~Network architecture and data transmission~~
~~Telephony and telecommunications~~
Analog electronics
Robotics
Digital signal processing
VLSI design
~~Data acquisition~~

Mathematics
Differential and Integral Calculus
Differential Equations
Linear Algebra and Matrices
Probability and Statistics
Mathematic logic ~~Predicate logic~~
~~Set theory~~
Graph theory
~~Information theory~~
~~Automata theory~~
~~Queuing theory~~
Combinatorics
~~Control theory~~
~~Formal Languages~~
~~Laplace and Fourier Transforms~~
Functions, relations and sets

Computational science
Numerical analysis ~~(Computational methods for numerical problems)~~
Operations research
Modeling and simulation

Related areas
Physics
Chemistry
Economics
Accounting
Marketing

Management
Entrepreneurship
Psychology
Philosophy
Technical writing
Giving presentations to an audience
Team work skills
Leadership
Negotiation
Legal/professionalism/ethics and society ~~Ethics and professionalism~~
Statistical process control
Scientific methodology
Second language

APPENDIX C. Invitation Letter

Olá,

Nós, do Laboratório de Qualidade e Produtividade de Software estamos realizando uma pesquisa para obter uma melhor compreensão da relevância dos temas da educação para o curso de ciência da computação.

Então, se você se formou no curso de ciência da computação entre os anos de 1998 e 2005, nós gostaríamos de convidá-lo para participar da nossa pesquisa.

Este questionário foi concebido para descobrir quais os aspectos da sua formação acadêmica têm sido útil para você na sua carreira. Suas respostas serão importantes para nos ajudar a obter uma compreensão exata das necessidades educacionais.

O questionário desta pesquisa estará disponível até o dia 1 de dezembro de 2008. Para responder este questionário você deverá utilizar menos de 30 minutos. Todas as informações que você fornecer será mantido confidencial.

Se você tiver alguma dúvida sobre a pesquisa, entre em contato com Gisele no LQPS - Laboratório de Qualidade e Produtividade de Software / UNIVALI via e-mail ou telefone +55 (48) 3281 1505.

Agradecemos o seu tempo para concluir este questionário.

Para acessar o questionário clique no link abaixo:
Versão em português:

http://ead.telemedicina.ufsc.br/survey/index.php?sid=22372&lang=pt-BR

Versão em inglês:

http://ead.telemedicina.ufsc.br/survey/index.php?sid=22372&lang=en

Prof. Dr. rer. nat. Christiane Gresse von Wangenheim, PMP
Universidade do Vale do Itajaí - UNIVALI - Campus São José
LQPS - Laboratório de Qualidade e Produtividade de Software
Rodovia SC 407, Km. 4 - Bloco 1, Sala 401
88122-000 - Sao José - Santa Catarina – Brazil
http://www.univali.br

Agradecimentos: Gostaríamos agradecer o Prof. Timothy Lethbridge por fornecer informações referentes a um survey anterior e o grupo de pesquisa CYCLOPS/UFSC por hospedar o sistema de survey.

APPENDIX D. Distribution List

2000email@yahoogrupos.com.br
4CIECO_2002@yahoogrupos.com.br
alunos@comp.uniformg.edu.br
alunos-cco@inf.ufsc.br
alunos-cco-owner@inf.ufsc.br
alunos-ppgcc@inf.ufsc.br
alunos-ppgcc-owner@inf.ufsc.br
atc2003-1@inf.ufsc.br
atc2003-1-owner@inf.ufsc.br
atc2003-2@inf.ufsc.br
atc2004-1@inf.ufsc.br
atc2004-1-owner@inf.ufsc.br
bachareis-cico2000@yahoogrupos.com.br
bccfema2004@yahoogrupos.com.br
cc_unip_2002@yahoogrupos.com.br
comp_turma_2000@yahoogroups.com
comp_turma_2000-subscribe@yahoogroups.com
digitz@yahoogroups.com
digitz-subscribe@yahoogroups.com
egressos@listas.univali.br
egressos-bccomp-l@ucs.br
egressos-ccomp-sj@listas.univali.br
egressos-comp@listas.dcc.ufba.br
egressos-comp-owner@listas.dcc.ufba.br
egressos-l@www.inf.unilasalle.edu.br
ex_alunos@comp.uniformg.edu.br
fic_computacao@yahoogrupos.com.br
FORMADOS_COMP@yahoogrupos.com.br
formadosarea1@yahoogrupos.com.br
mocondo@yahoogroups.com
pucsp98@yahoogroups.com
pucsp98-subscribe@yahoogroups.com
qualidade_soft_facig@yahoogrupos.com.br
turma99-1_di_ufpe@yahoogrupos.com.br
ufmg_dcc_1998@yahoogroups.com