Theory and Practice Behind the Course "Designing Enterprise-wide IT Systems"[1]

Alain Wegmann
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
I&C, EPFL, CH-1015 Lausanne, Switzerland
alain.wegmann@epfl.ch

October 3rd, 2003

Abstract:
As a consequence of the co-evolution of business and information technology (IT), the responsibilities of software engineers are expanding. They are becoming much more involved in business-related issues. When defining computing curriculums, this trend needs to be taken into consideration, for example, by proposing courses on business and IT integration. The author presents here a trans-disciplinary, problem-based learning course that addresses business and IT. The target audience is computer science and software engineering students. The course has three modules: a competitive game to illustrate business thinking, role-playing to practice IT requirement analysis, and an IT integration project to present how modern, off-the-shelf technologies contribute to IT system realization. Each module has several sections comprising experiential learning and traditional ex-cathedra lectures. The originality of the course lies in the combination of breadth of the subject and depth on what is taught. The goals of the course and its detailed contents are presented: the emphasis is on the process-related/technical and emotional learning experience by the students and on the author's experiences gained from teaching that course.

Keywords: business and IT alignment, enterprise architecture, experiential learning, problem-based teaching, business process modeling, requirement engineering, IT system integration.

---

[1] Article published in IEEE Transactions on Education, vol. 47, num. 4 (2004), p. 490 - 496.

I. Introduction

The co-evolution of business and technology, in particular information technology (IT), is becoming unavoidable. New business requirements foster the development of new technologies and vice-versa. A consequence of this co-evolution is that software engineers are increasingly exposed to business-related issues. This situation requires a broadening of their skills [1, 2], a broadening that often makes them uncomfortable. This paper presents the course, “Designing Enterprise-wide IT Systems”, which was developed to help future software engineers to bridge the gap between business and IT.

The “ACM/IEEE Computing Curricula 2001” [3] defines or refers to model curriculums for Computer Science (CS), Software Engineering (SE), and Information System (IS). The Computing Curricula recognizes that the curriculums should include “not only an understanding of basic subject matter, but also an understanding of the applicability of the concepts to real-world problems.” Despite this statement, only the IS curriculum [4] acknowledges clearly the need for integration courses between business and IT. The CS curriculum [3] and the SE curriculum [5] do not mention business/IT integration. Alternative SE curriculums, such as [6], do not mention it either. This paper presents the implementation of an integration course, with the same goal as those specified in the IS Curriculum, but targets future software engineers in a CS or in a SE curriculum.

The idea of the course, “Designing Enterprise-wide IT Systems”, came from: (1) the author’s goal to share with students some of the 14 years of his experience in the computer manufacturing industry (software development followed by management of R&D and manufacturing support groups in Switzerland, Taiwan, and US, and then management of a marketing group in US); and (2) a doctoral course, “Integrated Design for Marketability and Manufacturing”, given at Stanford University [7]. In this doctoral course, PhD students in mechanical engineering and MBA students team up to develop products. Through this course, they discover the importance of managing tradeoffs between technical and business issues. Two important differences with the Stanford’s course are: the course described in this paper is designed for an undergraduate curriculum (and not a doctoral / MBA curriculum) and it addresses IT development (and not mechanical development).

The course, “Designing Enterprise-wide IT Systems”, was taught as an elective to be taken in year 3 or 4 of the CS curriculum at the Ecole Polytechnique Fédérale de Lausanne (EPFL) technical university [8]. It was taught from 1997 to 2000. Most of the students had not taken any courses on marketing or finance before taking this one. The course was taught after the databases, middleware, distributed systems, programming languages, and software engineering courses.

This paper is structured as following: Section II defines the course’s goals and its design principles; Section III describes the content; Section IV describes what was learned from teaching the course.

II. Course’s Goals and Design Principles

Based on interviews with industrial partners, on the standard curricula, and on his personal experience, the author identified three main teaching goals for the course. First, the students need to understand how the different specialties taught at universities fit together as a whole [5]. Second, the students need to be aware of how a company works, how people collaborate within it, and how a company can sustain and improve its competitiveness. Third, the students need to learn how to work under stress and with incomplete, sometimes inconsistent, information.

To structure the content of the course, in collaboration with a group of Compaq[2], the author analyzed standards and best practices used in business and IT design.

The teaching goals and the content of the course led to using *experiential learning* as the pedagogical method of choice (in complement with traditional ex-cathedra lectures). According to [12], experiential learning is commonly attributed to two authors: John Dewey (1859 - 1927) and Kurt Lewin (1890 - 1947). Dewey [13] claims that learning is a result of a combination of the analyses made of difficulties that arise in practical experiences. Thus the instructor has two roles: (1) to set up the experimental conditions for the students so that they can experience and conceptualize what they learn; and (2) to confirm that the student's new knowledge is valuable by relating it to the state of the art.

The course uses the experience the students gain from the competitive game and role-playing [14]. The game and role-playing are based on the hard goods industry for the following reasons: (1) The hard goods industry deals with concrete, physical products. (2) Since the hard goods industry is at an extremely high level of maturity, proven practices exist. (3) These proven practices are becoming a standard for the service industry. (4) Concrete physical examples from different companies in the hard goods industry can illustrate the course.

III. Course Contents

The course consists of 84 periods over one semester: i.e., 14 weeks with 2 classes of 3 periods per week. The class size is between 12 and 20 people. The course has three modules: (1) the business level module addresses business goals and processes; (2) the operation level module addresses the role of IT in business; (3) the IT level module presents IT system integration using standard off-the-shelf technologies. The IT level module is purposely constructed to last as long as the first two modules to maintain an emphasis on the engineering aspects. The course outline is presented in Table 1.

---

[2] The Compaq group is now part of HP. They are active in enterprise architecture [9]. *Enterprise architecture* is the discipline whose purpose is to align more effectively the strategies of enterprises together with their processes and their resources (business and IT) [10]. The described course can also be considered as a course on enterprise architecture. The author's research is in this field [11].

Module	Iteration	Topic	Week
I. Business Level Module		Engineering	Week 1
		Finance	Week 2
		Quality	Week 3
		IS & ERP	Week 4
II. Operation Level M.		Interview	Week 5
		Contextual interview	Week 6
		Guest speaker	Week 7
III. IT Level Module	Iteration 1	n-tier	Week 8
		application server	Week 9
		database connector	Week 10
		web interface	Week 11
	Iteration 2	message queuing	Week 12
		xml	Week 13
		object trans-action monitor	Week 14

Table 1: Course Outline

The students are in teams of four. In the course, each team "works" for a different company. The company is an aircraft engine provider. All the engine providers are in competition and have the same customer called "NPI" (for "NewPlane Inc"). NPI makes flying cars (similar to a "real" company making a "real" flying car called "Skycar" [15, 16]). In the first module, the students play the role of the management team of an engine provider. In the last two modules, the students play the role of an IT group within the same company.

III.1 Business Level Module

The instructor's specific goals for the business level module are to teach the students: (1) to understand how a company operates and how its operation is related to its business goals, (2) to understand product development (i.e., its process, its importance for the company's competitiveness), (3) to realize the importance of stress management, interpersonal skills, and working methods; and (4) to understand how the IT system can support the business processes.

This module lasts four weeks. In the first three weeks, the instructor presents the main business process of the company, and in the last week he/she gives a demonstration of an information system supporting these processes. In week one, work is on the product development process; in week two, on production planning and finance; and in week three, on quality assurance. The course format is the same for the first three weeks.

In the first class of the week, the students play the business game, and the course ends with two debriefings - a "process debriefing" in which students identify and discuss the facts that made an impact and an "emotional debriefing" in which students identify and discuss their feelings and emotions.

In the second class of the week, the students conceptualize what they learn from their experience. The conceptualization is in two steps: First, the students conceptualize what they uncovered in the process debriefing; second, they do the same with what they realized in the emotional debriefing. These steps involve the students working in their teams, followed by their coming together to make a mind map [17]. The mind map is made by collaboratively writing, posting, and structuring A4-sized "Post-its" on the blackboard. Through this collaboration, they see that what they have conceptualized is similar or complementary. This discovery helps them understand that they are actually developing the underlying theory of the subject matter. To reinforce this learning experience, the instructor lectures briefly by: (1) presenting actual examples from real companies, examples that are similar to the problems experienced in the game; and (2) presenting the state of the art in the domain, introducing the trade vocabulary.

A description follows on how concretely the game is played. The goal of the game is to design and manufacture a new engine for NPI. The engine is an assembly made of Lego® bricks that NPI will hook to a Lego® flying car. To make the engine, each provider must select and work with a single supplier. There are seven possible suppliers: four part suppliers and three design suppliers (also called "original design manufacturers" or "ODM"). The suppliers provide the Lego® bricks necessary to make the design. They also give application notes explaining how to put the parts together. Buying from an ODM saves the design time, but has a cost. The suppliers and NPI are played by the instructor. In this paper, the author illustrates in details the first week of the game. The remaining two weeks are more succinctly presented. To make the explanation clearer, the author describes the first week of the game from one provider's viewpoint. The provider's name is "Best Engine". First, each team, and in particular Best Engine, gets the following material:

- one *board* representing the provider (i.e., Best Engine) with its development and manufacturing processes, the selected supplier, NPI, the banks, and the shareholders;
- 70 *coins* (one coin representing 10'000.- monetary unit; a monetary unit is equivalent to a dollar or a euro), each financial transaction is realized by moving coins on the board;
- one *design and parts list* that describes the technical characteristics, the availability and financial conditions to get design (from design's suppliers) or parts (from part's suppliers);
- one *journal* (printed spreadsheet) used to log all events and transactions, to audit the company, to record the company situation from week to week, and to make the financial report;
- two *yearly financial reports* (printed spreadsheet), one per financial year, used to compute the profit & loss, the balance sheet, and the cash flow statements in each fiscal year;
- a small *manual* explaining the game.

The game is now described month by month (in company time). Table 2 summarizes the first three weeks of the course.

month	Week1 : Engineering	month	Week 2: Manufacturing and Finance
1	Best Engine creation	13	NPI places an order for production units
	NPI gives to Best Engine an RFQ		Best Engine orders for the parts
2	Best Engine prepares the RFQ answer		Best Engine closes its financial year
3	Best Engine prepares the RFQ answer	14	Best Engine waits for the parts
4	Best Engine gives the RFQ answer to NPI	15	Best Engine receives the parts
5	NPI orders 10 protos		Best Engine manufactures and delivers the units
6	Best Engine orders the parts for the protos	16	
7	Best Engine waits for the parts	17	Best Engine gets paid
9	Best Engine manufactures & delivers the protos		
	NPI evaluates the protos	month	**Week 3: Quality**
10	Best Engine has to rework the protos	18	NPI places a stop shipment
	NPI agrees to pay the protos		Best Engine is asked to make a product recall
11			Best Engine proceeds to the failure analysis
12	NPI pays Best Engine		Best Engine replies on the necessity of the product recall

Table 2: Description of the course's first three weeks

In month one, the team sets the board, by putting, respectively, 3M and 2M on the bank area and on the stakeholders' area of the board. Now, Best Engine needs to be financially created. To do so, the students move 3M from the bank and 2M from the stakeholder into Best Engine boundary. These moves correspond to a loan from the bank and a capital investment from the stakeholders. This money is used by Best Engine to buy land (1M) and, equipment (2M); the rest (2M) remains as cash for operations. As soon as Best Engine is created, operating costs (such as salaries) are paid every month. All engine providers have the same initial investment. At the end of month one, Best Engine gets a request for quotation (RFQ) from NPI. NPI is interested in buying engines for its new product - a flying car. The RFQ, identical for all engine providers, is a request for a detailed product specification, a product price, and a production lead-time. The RFQ details the expected requirements (form factor, performance, and production quantity) and the number of samples needed for evaluating the product. Best Engine has three months to reply to the RFQ. To reply, Best Engine needs to select a design among all possible designs (as shown in the design and parts' list), then to evaluate the terms and conditions for the sale. These engines have different costs, performances, development costs, development times, and sourcing lead-times. These characteristics were carefully selected, based on the author's experience, to reflect the typical situations found in "real" companies. When Best Engine has selected its supplier, it has to inform the instructor who, in turn, has to ensure that no other engine providers use the same supplier. This action simulates the limited capability of the suppliers and increases the competition. At the end of month four, Best Engine (as well as the other providers) gives its quotation to NPI. In month five, NPI orders 10 samples. Best Engine designs its engines at a given development cost and a given lead-time (typically one month). If a complete design is bought from a design supplier, there is no design lead-time. Best Engine can anticipate the supplier lead-time and can order the parts for the prototype even before the design is complete. In this example, Best Engine buys in month five from a parts supplier with two months lead-time. In month seven, the parts are received (i.e., the Lego® bricks and a stack of coins representing the cost of the parts). The manufacturing lead-time is one month. The manufacturing costs are added to the cost of the parts. In month eight, the engine is delivered to NPI. Depending on the selected supplier and on the provider's initiative in getting the parts in an anticipated manner, the prototypes can be delivered between month seven and month eleven. When Best Engine delivers the prototypes, it provides a Lego® engine corresponding to the selected design. NPI puts the engine on a Lego® aircraft. It then becomes evident to Best Engine that the requirements provided by NPI and the specifications provided by the suppliers were incomplete. As a result the providers need to rework all engines. Depending on the selected design, the time to rework can be between one month and seven months. In the case of Best Engine, they need to rework their engine for one month. Their new prototypes are delivered in month nine. NPI agrees to pay for the prototypes (and the development cost if Best Engine included the payment of the development cost as part of its answer to the RFQ). The payment lead-time is three months (unless it has been negotiated differently by Best Engine). Finally, in month twelve, Best Engine receives the payment from NPI. Ideally, the revenue should cover the development costs, manufacturing costs, parts costs, and operating costs. Quite often the revenue is not adequate to cover the costs: thus the students realize the complexity of evaluating, under stress, the product cost and price.

During the process debriefing the students describe the practical problem they had: incomplete RFQs, designs that did not satisfy the specifications or that did not work, inadequate pricing, etc. During the emotional debriefing, they describe the stress they felt, the difficulties of choosing without all the necessary information, etc. In the second course of the week, the material identified during the debriefings is conceptualized. After this experience, the students discover the concept of development process and the importance of the different tasks in the process (such as contract review or design verification / validation). What they discover in the emotional debriefing led them to the theory of project management - a means to reduce stress and to improve the predictability of the project's results. At the end of this first week, two of the goals set for the business level module are already achieved. The following weeks will actually reinforce this learning.

Before week two, the instructor decides which quantity NPI will purchase from each provider. The quantity depends largely on the provider's time to market and, to a lesser extent, on the selling price. For example, the provider who is able to deliver first receives an order for 50% of the quantity to be produced. This selection illustrates the importance of time to market (to command a higher market share and margin) and of manufacturing planning (to get parts on-time). In week two, the game resumes. During this second week, the financial year ends for all teams; and the students have to update the yearly financial report using the journal and the coins on the board. The game and the financial documents are designed in such a way that one can easily understand the necessary concepts by looking at the board and at simple equations shown on the documents. In the process debriefing, the students explain the logic behind the profit & loss, the cash-flow statements, and the balance sheet (based on what they have observed in terms of parts/products and monetary flows). In the emotional debriefing the students analyze their feelings after this second week. Usually, there is not much difference from the first week. This analysis triggers a discussion on how experience gained in one context can be reused in another context. It leads to the discovery of the concept of "meta-model" (a model in which knowledge is captured independently of a given context).

In week three, the game continues. At that time, the companies get a "stop shipment" notice. Each company experiences different kinds of problems (e.g., leaks, over-consumption, explosion, etc.). These problems might require a product recall. The companies that were ahead (in terms of volume shipped) are in a more critical situation because they would have to recall more engines. NPI requires the providers to explain why the engines are faulty and what quality process the providers have in place. If the provider is able to explain why only one engine would have a fault, the recall can be avoided. Of course, usually the providers did not anticipate the need of a quality system. In the debriefing, the students define what the minimal and necessary quality processes are. Here again, the course ends with the presentation of concrete examples from real companies. A brief emotional debriefing is also made.

In week four, the game is over. The students brainstorm on what should be the functions of an IT system necessary to support "their" company. Then a running demonstration [19] of an Enterprise Resource Planning (ERP) system is shown, illustrating again, the business processes in companies and how the IT system supports these processes.

III.B Operation Level Module

The instructor's specific goals for the operation level module are to teach the students: (1) to understand what is specific to the requirement analysis of an IT system, (2) to understand how to identify and select practical solutions based on what already exists as IT infrastructure in the company (which dictates what is feasible or not), and (3) to realize how to reuse experience from context to context.

In the operation level, the teams of students role-play the IT group of the same company as before. In this module, each team receives the request to improve the existing IT system. The project is triggered by the frustration of the customer representative who cannot obtain information about the spare parts from the IT system. When a customer, such as NPI, needs spare parts, the customer representative must call the warehouse clerk. Since the warehouse is located in another continent, the communication is tedious because of the time zone differences and language barriers. The team must reply to this request with a precise IT system specification and an implementation plan. They will have to implement the IT system in the third module of the course.

The operation level module lasts three weeks. In the first week, the team makes the IT specifications more precise. The team interviews the customer representative, the warehouse clerk (by phone and with language difficulties to communicate with the person), and an engineer from NPI in charge of the maintenance of the engine. These roles are held by teaching assistants. The team then first categorizes the problems using a cause and effect (or fishbone) diagram [20] that shows the problem to be addressed and the causes of the problem. Based on this analysis, they specify the IT system's expected features and priorities (e.g., enable the customer's representative to get on-line data on the spare parts). During the debriefing, the students raise the point that they do not get consistent answers from the different interviewees (the different roles have been defined to have slight inconsistencies, similar to what is frequently observed in companies). At this point they decide to actually go see how these people work. Thus, the students realize the importance of observing the real business processes, instead of basing their work on what people recall about their responsibilities. This method is called contextual inquiry [21].

In the second week of the operation level module, each team goes "on-site". To make this visit possible, a mock-up of the IT system was developed, so that students can actually watch the teaching assistants role-playing the different jobs. For example, in one location, the students can observe the customer representative on the phone, using his or her application (i.e., a mockup of the IT system that has been developed and that represents the function of the ERP system). In another location, the students can observe the NPI's maintenance engineer who desperately needs spare parts and who has difficulties explaining to his representative which parts he wants. The warehouse clerk cannot be visited because he works oversea. After having made their observations, the students make a second analysis of the problems, and they discover the initial request (i.e., on-line consultation for the customer's representative of the warehouse inventory) is not the best solution to the problem. They notice that an on-line selection and ordering of spare parts can be done directly by the customer because the customer has all the necessary information to place the order if the IT system is accessible at all times and in all locations. In principle, the realization that the initial request for development was not adequate should not be a surprise (the students lived a similar situation in the business level module). See Section IV for a discussion on this point.

In the third week, each team writes the requirements for the IT system. The requirements are defined using a set of Unified Modeling Language (UML) [22] diagrams (i.e. use-case and deployment diagrams) and some textual descriptions. The students write their implementation plan by analyzing what already exists as IT infrastructure in the company (model “as-is”). The students can role-play the interview of a senior IT specialist of the company. The team then defines possible scenarios representing the different solutions that can be provided (models “to-be”) and selects among these solutions. They decide if they will implement their solution by purchasing a commercial product or by building their application themselves (“buy / build decision”). If they decide to build, they evaluate whether they should do it internally or should outsource it. The selection is based on an analysis of return on investment, on a risk analysis, and on the chance that the IT system will really be used [23].

To conclude this second module, a guest speaker is invited to make a presentation and reply to the students’ questions regarding the relations between people issues and IT development.

III.C IT Level Module

The instructor’s specific goals for the IT level module are to teach the students: (1) to understand how to use commercially available platforms to develop IT applications and, in particular, how applications can be developed by assembling and wrapping legacy systems; and (2) to understand how fundamental mechanisms taught in the other courses are packaged in commercial products.

The IT level module lasts seven weeks. In this module, the students develop, by assembling available technologies and wrapping legacy systems, a web application that provides the inventory status and the capability to order spare parts. This web application can be used by the customer, the sale representative, and the warehouse clerk. The third module uses the Microsoft platform extensively, and in particular, COM+/ MTS/ MSMQ/ ADO (now integrated in .Net [26]). The Microsoft platform was chosen because it was not known by the students. Another alternative could have been the OMG platform’s J2EE [27]. The module simulates an iterative approach in which an increasingly sophisticated application is developed. In the first iteration, the goal is to make a web interface to access information on the spare parts. The following technologies are used: n-tier architecture, application server, database connector, and web interface. Then, a second iteration is made to add the on-line ordering of the spare parts. The following additional technologies are used: object transaction monitor and XML-based message queuing.

An important aspect of this module consists in searching information in the Internet vendor’s documentation. This trains the students on how to access web resources available to developers.

IV. Learning Experiences with the course

Since the course deals with business topics, IT requirements and information technology, the students do not think that such breadth could have any meaningful depth. At the end of the course, the breadth/depth combination is what the students appreciate the most. It shows them that, even as engineers, they may, in a relatively detailed manner, be involved in areas in which they are not used to working. It teaches them not to fear such involvement (and this relates well with the author's experience as an engineer and manager in a rapidly growing company).

One of the first challenges was to attract and keep students. By its style, the course appears very unusual to students; it was the only course in the curriculum based on experiential learning. In particular, students doubt whether they will learn anything in a game and, more importantly, whether they will be judged fairly at the end of the course (the course has an oral exam; each student must answer three questions - one per module). With the help of the pedagogic group [28] the message, given the first time the instructor met with the students, was improved to address effectively their fears.

A second difficulty lay in the practical application of the experiential learning method. Initially, not enough time was allowed for the students' conceptualization. As a result, the students had the impression that they were not learning much in the game and in the role-playing. Following the suggestion of the pedagogic group, the time, given to the students to conceptualize, was increased considerably, thus solving the problem. This point is essential because it is through the discussion with their peers that the students actually discover the theory underlying their experience. This change made the course successful.

One of the most original aspects of the course is the work with emotions and feelings (proposed by Nadine Stainier, from the pedagogic group [28]). This aspect helps engineers understand how to benefit from their emotions and to challenge their work practices based on what they feel. As a result, they can become more effective professionals.

One important difficulty for the students is the reuse of their experience from context to context. For example, at the adequate level of abstraction, the first section of the first module (i.e., the reply to the RFQ for aircraft's engine) is similar to the first section of the second module (i.e., the request for an IT system development). Experience shows that almost none of the teams thought of reusing knowledge from one module to the next. The context is so different between the engine development (first module) and the IT system development (second module) that the students do not recognize that they are in a similar situation of product development. A possible improvement of the course would be to assign one student per team as a "process and knowledge person" to help them figure out how to reuse experience.

The instructor's presentation of concrete examples and the guest speaker's presentation were very important in helping the students believe in what they learn. In summary, the diversity of the teaching methods used (competitive game, role-playing, testimony, examples, actual IT implementation) contributed to the originality and relevance of the course.

To have the course based on an engine manufacturing company (i.e. hard goods) is also interesting. The business processes used in the hard goods industry (and shown in the business level module) provide a powerful metaphor that can be used by the instructor to illustrate, in the last two modules, how to design and manufacture IT solutions. Specific design methods are based on this principle [24, 25]. In these methods, an IT system is considered as a company that acquires parts (e.g. financial information), manufactures products from these parts (e.g. generate financial reports from the financial information) and then distributes the products to the customers (e.g. distribute financial reports to the company management via some data warehouses).

After three years of practice and improvements, the course reached a high level of maturity. This was reflected by the evaluations from the students. The course eventually attracted between 15 and 20 students, a good level of participation for an elective.

When the responsibility for the course was transferred to another instructor, it became apparent that such a course could not be taught by someone who did not share a similar practical experience as the author. The solution found was for two instructors to give the course - one being responsible for the business level and operation level modules, and the other for the IT level module.

V. Conclusions

Software engineering is increasingly exposed to business issues. Presented here is a problem-based course that exposes future software engineers to business and IT issues. This course satisfies the goals identified by the author: (1) to show how the different specialties taught in universities work together (second and third modules); (2) to raise the engineers' awareness of their role in enterprises' competitiveness, and (3) to give engineers the opportunity to experience the challenge of working in multiple contexts (first and second modules). In addition, through the course, the students understand the importance of the "soft issues" and how their awareness of these issues can improve their professionalism.

With such a course, the students will not only integrate more quickly in their professional life, but they will also contribute in very different and original ways. They will stimulate their companies to become more pro-active in considering their business / IT integration as a competitive advantage. For this reason, this course could also be considered for on-job training.

Acknowledgments

I wish to acknowledge the very significant contributions of the following people: Gil Regev (IT level module), Guy Genilloud (operation level module), Nadine Stainier (pedagogic contents), and Frédéric Bouchet (technical contents).

References

[1] Yeh, R., "Educating Future Software Engineers," *IEEE Transactions on Education*, Vol. 45, No. 1, Feb. 2002.

[2] Rainey, V., "Beyond Technology - Renaissance Engineers," *IEEE Transactions on Education*, Vol. 45, No. 1, Feb. 2002.

[3] ACM and IEEE Task Force on Computing Curricula, *Computing Curricula 2001 Computer Science*, IEEE Press, 2001. Available: www.acm.org/sigcse/cc2001

[4] ACM and AIS [Online 20030710], *MSIS 2000 Model Curriculum and Guidelines for Graduate Degree Programs in Information Systems*, 2000. Available: www.aisnet.org/Curriculum/#curriculum

[5] ACM and IEEE Task Force on Computing Curricula [Online 20030710], *Computing Curricula on Software Engineering*, SEEK Project. Available: sites.computer.org/ccse/

[6] IEEE Software Engineering Coordinating Committee [Online 20030710], *Software Engineering Body of Knowledge.* Available: www.swebok.org

[7] Beach, D., and Srinivasan, V., *Integrated Design for Marketability and Manufacturing (IDMM)*, Stanford University, 1997.
[8] EPFL [Online 20030710], *Computer Science Section*, http://sin.epfl.ch.
[9] HP [Online 20030710], *IT Strategy and Architecture (ITSA),* Available: http://www.hp.com/hps/strategy/sa_method.html.
[10] Zachman, J.A., "A Framework for Information Systems Architecture," *IBM System Journal*, vol. 26, no. 3, 1987.
[11] Wegmann A., "On Systemic Enterprise Architecture Methodology (SEAM)", *International Conference on Enterprise Information Systems, 2002*.
[12] Kolb, D., *Experiential Learning*, Prentice Hall, 1984.
[13] Audi, R., *The Cambridge Dictionary of Philosophy*, Cambridge Press, 1999.
[14] Becker, R., "Taking the misery out of experiential training," *Training*, Feb 1998.
[15] Moller International Inc. [Online 20030710], www.moller.com/skycar
[16] Waldman, P., "Great Idea… if it Flies: Persistent Inventor Readies Newest Edition of 'Skycar'," *Wall Street Journal*, June 28, 1999.
[17] Buzan, T., and Buzan, B., *The Mind Map Book,* Plume, 1996.
[18] ISO, *ISO 9000:2000 Quality Management Systems*, 2001.
[19] SAP, Visual. Discover R/3 3.0, CD-ROM
[20] Ishikawa, K., *What is Total Quality Control?*, Prentice-Hall, 1985.
[21] Beyer, H., and Holtzblatt, K., *Contextual Design*, Morgan Kaufmann, 1998.
[22] OMG [Online 20030710]. *Unified Modeling Language*. Available: www.omg.org/uml.
[23] Markus, M. L., and Keil, M., "If we Build It, They Will Come: Designing Information System That People Want to Use," *Sloan Management Review.* Summer 94.
[24] Evans, M., *The Software Factory*, Wiley, 1989
[25] Apple Computer Inc. *Vital Fundamentals,* Apple Press, 1993.
[26] Microsoft [Online 20030710], *.NET*, www.microsoft.com/net.
[27] Sun [Online 20030710], java.sun.com/j2ee/.
[28] EPFL [Online 20030710], *Centre d'appui à l'enseignement* (CAP-GE), www.epfl.ch/CPD.