Do Engineers Have Enough Computing Knowledge and Skills after Obtaining the Undergraduate Degree?

Fani Zlatarova
Elizabethtown College
Elizabethtown, PA

Pavel Azalov
Penn State Hazleton
Hazleton, PA

Do Engineers Have Enough Computing Knowledge and Skills after Obtaining the Undergraduate Degree?

Fani Zlatarova
Elizabethtown College
Elizabethtown, PA

Pavel Azalov
Penn State Hazleton
Hazleton, PA

Abstract

One of the main student learning outcomes for engineering students could be formulated as: "*Collect, analyze, and interpret data*". Obviously, this statement implies possessing computing knowledge and skills. Current engineering students are future operational employees, supervisors and team leaders, middle managers and knowledge workers, and also top managers. Obtaining an appropriate background in Computing during the years of undergraduate studies is important for their successful career. The authors of this paper try to answer the question if the currently offered undergraduate computing courses for engineering students provide the needed preparation for taking advantage of Information Technology when developing a variety of projects in the everyday professional activity. After analyzing academic programs for engineering majors at different academic institutions, a recommendation is proposed to include the Systems Analysis and Design course belonging to the Software Engineering computing area. This course should be a required course or at least an elective course not only for students majoring in Computer Engineering but also in all Engineering curricula. The content of the course includes basic topics from the Computing theory and practice and provides students with a rich variety of Information Technology tools needed for the planning, analysis, design, implementation, operation, and support of engineering activities.

1. Introduction

Along with the traditional knowledge and skills, the engineering practice requires a great degree of creativity when developing miscellaneous projects. Finding an appropriate solution for the problems related to a project starts with the analysis and design of the respective problem and after that choosing the approach to solve it (Fig. 1). Presently, information technology (IT) is widely applied in all areas of the human life and especially in the engineering area because it offers significant advantages.

Software Engineering (SE) as one of the major computing sciences emerged from the traditional engineering practices by introducing IT to them. Today, it includes highly advanced methods that allow the development of software systems applied in different engineering-related cases. Not only the SE theoretical aspects have been researched, but SE-related standards have been established by the American National Standard Institute (ANSI) in collaboration with the International Standard Organization (ISO). In their everyday practice, engineers use a variety of software products as users and also as developers of specialized software systems. This is the reason SE represents an indivisible part of the engineering projects [5, 9].

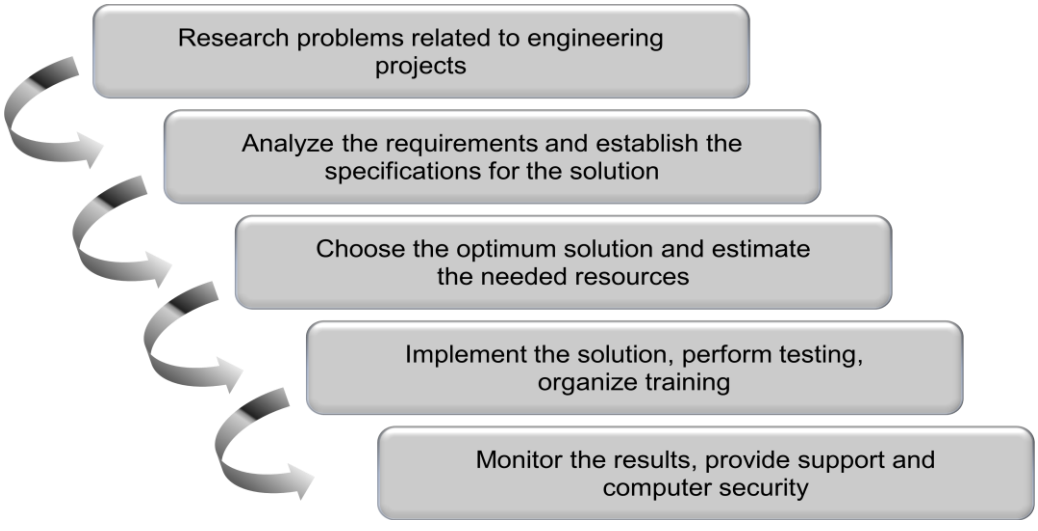

Figure 1. Steps in developing engineering projects

Researching Engineering curricula, which are different than Computer Engineering (CE) curricula, it would be interesting to understand if the computing courses included in them provide students with the needed computing knowledge and skills [6, 14].

2. Computing Courses in Engineering Curricula

Most academic institutions require only one undergraduate computing course for engineers and usually, this is an introductory course in programming, based on a programming language such as Java, C++, FORTRAN, Python, and others [1]. The programming course is very useful, because students would have the opportunity to learn introductory programming paradigms and develop routine in writing programs in a specific programming language. This is useful because engineers use existing software packages containing program libraries especially developed for solving engineering problems. They also participate in miscellaneous real projects at different levels. Basically, they need to provide the right input to existing programs and interpret the output obtained after the programs execution. There are colleges and universities offering few SE topics as a part of other courses. For example, it is possible to find similar courses having names such as Project Management, Basic Design, Introduction to Engineering Design, Engineering Design with CAD Systems, Engineering Design and Graphics, and others. However, thinking about the possible overall activities relevant to engineering projects, essential computing knowledge and skills are not part of the engineering curricula. With small exceptions, even Engineering curricula do not include the Systems Analysis and Design (SAD) course that represents the basic course from the SE area. This should be considered as absurd. The SAD course definitely should be one of the keystone courses offered to CE students. Mainly, the discussion from this paper focuses on including the SAD course in the curricula for engineering majors different than CE.

Students, who realize the importance of having a solid background in Computing take additional computing free elective courses or register for a Computer Science (CS) or Information Systems (ISs) minor or even major. This track is very challenging for students because engineering majors require a very high number of credits. The authors' opinion is that adding a SE course, e.g. SAD, as a required course or at least as an elective course for engineering students would be extremely useful. Students would be able to benefit from the topics discussed in this course, to develop experimental practice-oriented projects, to understand the core of the project management, and to practice with existing computer-aided software engineering (CASE) tools including computer-aided project management tools.

Having this course in the engineering curricula would better satisfy the recommendations established by leading organizations such as ASEE, IEEE, ABET, and others about engineering curricula and respective accreditations.

The phases of the project development from the engineering point of view and the related computing knowledge and skills accompanied by corresponding software products are described in the next section. This is intended to show why possessing strong background in SE would be important for engineering students.

3. The Role of the Systems Analysis and Design in Engineering Projects

Engineering activities rely heavily on implementing a variety of software systems which represent specialized ISs involving a high degree of integration, i.e. the output from an IS is accepted as the input by another IS. ISs combine IT, people, and data to support application requirements. In particular, Engineering represents a major application area. An IS could be graphically drawn as it is shown in Fig. 2.

Figure 2. Components of an information system

This simplified model suggests to a high extent why ISs are so important for engineers. In most of the cases, they use ISs to collect and store data, to use them for executing programs from specialized software packages, and to analyze, visualize, and interpret the obtained results. Sometimes, they also participate as developers of special-purpose ISs implemented when building up civil or industrial constructions, creating intelligent machines, designing power networks, researching biodigital problems, and many others [2, 3, 13]. Using the SSP (Subroutine Scientific Package), which is an IBM software product written in FORTRAN, could be considered as one of the first steps in the evolution of incorporating IT methods in the engineering research and practice. Currently, it is still used; however, the IT advances include much more. The innovative spirit that is characteristic for engineers puts them in a position to be among the first people adopting the most recent IT methods and approaches and implementing a variety of ISs, such as:

- enterprise computing systems,
- transaction processing systems,
- business support systems,
- knowledge management systems, and
- user productivity systems.

These systems are used by engineering professionals working at different levels of a company or organization (Fig. 3).

Figure 3. Typical levels of a company or organization

Developing ISs or using them requires engineering students to be able to know how to work with miscellaneous systems development computing tools such as:

- tools for project management,
- tools for systems modeling and design,
- tools for creating experimental models by applying agile methods, and
- computer-aided software engineering (CASE) tools.

Engineering students should also be familiar with the 4G (4th Generation) environment which includes data modeling languages (UML as a world standard), database languages (SQL as a world standard), powerful CASE tools, application generators, report generators, and others. At the moment of their graduation, engineering students should have experience from using different approaches for systems development, such as the traditional structured analysis, the object-oriented analysis, and the widely implemented contemporary agile methods [10].

4. The Variety of Basic Computing Knowledge and Skills for Engineering Students

As it was mentioned above, Engineering undergraduate curricula include, if any, introductory programming courses, such as Programming for Engineers with C++, Programming for Engineers with Java, or Programming for Engineers with FORTRAN [4, 8]. Actually, in most of the cases, students are required to select only one of them. This is far away from the understanding of a strong computing background for engineering students which is needed for the development of practice-oriented experimental engineering projects and for their future professional activity. These students should possess SE-oriented knowledge and skills relevant to the following phases characteristic for the development of an engineering project:

- project planning,
- project analysis,
- project design,
- project implementation, and
- project operation and support.

The corresponding project development computing tools relevant to each phase are discussed.

4.1. Project Planning

Usually, this phase includes the formal request for the development of an engineering project by describing corresponding problems which should be solved. The preliminary investigation performed during this step consists of the analysis of the project's justification and feasibility. CASE tools such as Visible Analyst, Systems Architect, and others could be used for the preparation of the overall documentation, which is supposed to fulfill rigid standards imposed by ANSI and ISO. Using CASE tools affects the project management in a positive way and creates a better support for achieving the final goals of the project. The individual and group performance improves. The integrated meaningful information allows making right decisions. There exist stronger controls that lead to a higher quality of the project and reduce the cost of the project tasks.

The most widely used software products for automating the management of projects are Microsoft Office Project and Open Workbench. They allow the good understanding of project details and are very appropriate for the visualization of different project aspects. Both systems are applied for planning, scheduling, monitoring, and reporting activities performed during the planning phase. The project's success is determined by specific quality standards and depends on the decisions made about the project scope, budget limits, and time constraints. All of these factors could be modeled by using corresponding SE methods and tools to create a corresponding set of diagrams such as Gantt charts, PERT/CMP charts, network diagrams, and numerous spreadsheets.

Knowing how to create diagrams, which are specific for the planning phase, is important. For example, fishbone diagrams are considered to be an analysis tool that graphically highlights existing causes of problems. Pareto charts represent one of the most used tools for quality assessment. Using spreadsheet processing systems and even the graphical features of the recent text-processing systems empowers analysis and visualization aspects related to the planning process. Spreadsheet systems are also very appropriate for purposes of the risk management of the developed projects.

Having a clear vision and correct understanding about the organizational structure of the company or organization, which manages the project, is critical for the smooth flow of the entire project. Organizational charts should be developed and Microsoft Visio would be a helpful project development tool to draw them.

Using report generators belonging to the 4G environment would be useful for reporting the progress in the project development at the end of each project phase.

All above mentioned tools allow the correct analysis of the project usability and help to estimate the needed costs, time schedules, and possible benefits.

4.2. Project Analysis

The analysis of engineering projects is similar to the systems analysis that represents the second step in the project/system development. This step includes the preliminary modeling and team-based development strategies, the fact-finding techniques, and preparing the corresponding documentation. Students should be taught to acquire skills that allow them to identify a problem, evaluate its main elements, and find an optimum solution. Conveying interpersonal communication skills to students is also important for the successful accomplishment of a project. SE provides information to students about the most used group-based techniques for developing team projects such as JAD (Joint Application Development), RAD (Rapid Application Development), and available modern agile methods, which reflect current trends for productive and creative collaboration between users and project developers.

There exists a rich set of modeling tools and techniques related to the analysis phase. The most important of them are mentioned below.

The functional decomposition diagrams (FDDs) are used to visualize function and/or processes. They implement the top-down approach in representing the main tasks of a project and the respective functionality. Another graphical analytical tool related to the structured analysis is provided by drawing data flow diagrams (DFDs). They show a closer look to the specific data exchange operations, represented through data flows, and describe corresponding process details at different levels. When documenting project aspects related to the object-oriented approach, UML, which is accepted to be a world standard, is very helpful for the development of a variety of useful diagrams. UML is not related to a specific programming language, and this offers flexibility in drawing diagrams such as:

- class diagrams,
- object diagrams,
- use case diagrams,
- state diagrams,
- sequence diagrams,
- activity diagrams,
- collaboration diagrams,
- component diagrams,
- deployment diagrams, and
- other features accompanying them (packages, notes, and stereotypes).

Engineers work with people representing different groups participating in the project development. They should possess the skills needed to conduct successful interviews, to write questionnaires and surveys, and to analyze and interpret obtained results. Working with documentation, which is electronically stored, is also part of the analysis phase. Being able to use contemporary communication tools and systems for distributed collaboration is highly needed.

The analysis phase also includes the data and processing modeling by using appropriate data models. This could be performed by drawing Entity-Relationship (ER) diagrams, containing the data and the relationships among them, or by drawing DFDs by choosing one of the existing sets: the Gane Set and the Sarson and Yourdon Set. The good understanding of the concepts related to DFDs, such as balancing, leveling, data flows, and processes would lead to the production of correct and meaningful diagrams that reflect clearly the project essentials.

Using data dictionaries, also called catalogs, containing the description of all the data and everything related to them allows a high data independence at logical and physical levels of the data representation and the data processing. Knowing how to describe existing processing relevant to a project is also needed. Flow-charts can graphically show the programming constructs inside the modular design of a project. Other commonly used tools for process description are:

- Structured English, similar to a Pseudocode used to explain algorithms for solving problems by using a computer;
- Decision Tables, showing the logical structure and displaying the possible combinations between outcomes and existing conditions;
- Decision Trees, which graphically represent all the elements contained in a decision table.

The object-oriented modeling of data and related properties and operations (methods) is possible if knowing the corresponding object-oriented concepts, tools, techniques, and object-oriented programming languages (for example Java and C++) which are considered in most of the introductory programming courses offered to engineering students. Understanding terms and notions relevant to the object-oriented analysis such as object, class of objects, inheritance, encapsulation, polymorphism, associativity, and others could be used when writing program code which solves specific project-related algorithmic problems. As it was mentioned above, UML is the main modeling tool in the object-oriented system analysis. Engineers should be able to use the most recent strategies for project development. Possessing knowledge about the traditional development strategies is a requirement. However, the Web-based strategies start to overtake because of the globalization of the world's economies. SE will provide engineering students with knowledge and skills to perform cost-benefit analysis and to choose the right software packages and hardware products which could empower tremendously the development of their projects. Students should be familiar with the dynamic aspects of IT and should have the ability to learn new computing technologies, such as the cloud computing, the upcoming Web 3.0, and everything which makes the virtual reality become an everyday practice [7, 12].

Performing financial analysis related to projects by using appropriate software products is required when analyzing the cost and benefits of a project. Orientation about existing specialized software and needed corresponding hardware is helpful. All the guidelines related to the project development and provided by SE will assure a solid basis for the successful termination of the project.

4.3. Project Design

After accomplishing the planning and analysis of a project, the next phase, the project design, could start. The output of the results from the project should be determined at this moment. To a high extent, the success of the project is based on the user-friendly human-computer interaction, i.e. on the language tools and interface environment used in an IS which is implemented in an engineering project. SE provides specific principles which should be known and kept. When engineers use specialized software packages, an appropriate output should be produced to help them obtain valuable multimedia results. Knowing how to create miscellaneous reports, such as detail reports, exception reports, or summary reports would be extremely appreciated for the decision making process. Learning how to work with pivot tables in spreadsheet processing systems or in database management systems (DBMSs) would be also helpful. SE also teaches how to create well-structured and well-edited documents and Web story boards which are technically sound.

One of the most frequently asked questions by employers during internship and job interviews is if the applicant can use the relational language SQL used in the majority of DBMSs today because ISs are used everywhere in the human life. They are based on databases and DBMSs. Respective basic knowledge and skills about them should be acquired by engineering students. They should understand topics from the area of the data design, data models, relational and object-oriented DBMSs, normalization, query processing, Internet DBMSs, and others.

All projects involve collection of data and their storage in appropriate media. Organizing them in a database is already a tradition. Developing database diagrams in different DBMS environments would clarify the data modeling and the establishment of correct relationship corresponding to a given data model. Applying integrity rules, such as entity integrity and referential integrity, is directly related to the development of specific databases. Students should know the basic maintenance operations on data along with the retrieval operation. This knowledge is required for always having well-organized and consistent data in a database. Methods of the Artificial Intelligence are used when creating knowledge databases

used for the development of data warehouses by applying data mining software which looks for meaningful patterns and desirable relationships among data.

One of the most important requirements today when processing data in different computer and networking environments is related to the data, computer, and network security issues. Possessing basic knowledge about network systems software and hardware is critical for the safety of the electronic communication used in the majority of the developed projects in a global environment.

4.4. Project Implementation

After the first three phases, the project finally should be implemented, tested, and the people, who will use the project outcomes, should be trained. SE describes the implementation tasks which are accompanied by corresponding quantitative and qualitative metrics. During this phase, structure charts are developed to visualize the modularity of the project tasks and to describe their functionality by determining the corresponding primitive functions.

Engineering students would benefit to understand the possible conversion steps in the project implementation such as:

- direct implementation,
- parallel operation,
- pilot operation, and
- phased operation.

They also should be able to perform post-implementation evaluation by using specific CASE tools and appropriate application software systems.

4.5. Operating and Support Activities

The support activities for managing the results obtained from a project are important for the smooth operation. They include different types of maintenance tasks such as

- corrective maintenance,
- adaptive maintenance,
- perfective maintenance, and
- preventive maintenance.

Teaching SE to engineering students creates the opportunity for them to adopt the principles of ethical conduct and moral behavior as users and developers of software products [11]. Students should be able to show integrity in their individual decisions as engineering professionals. These decisions could affect in a positive way their social and working environment.

5. Conclusions

By describing the major SE topics and in particular the SAD topics, it would be possible to understand how important they are for students majoring in different engineering fields. Some of these topics are considered briefly in introductory programming courses which are a part of engineering curricula. However in the majority of the cases, they are not included at all. A well-designed SE course would be a good addition to every engineering curriculum. The content of this course would enable the development of practice-oriented individual and team projects [15] which provide students with a routine in using a rich

variety of appropriate software products and corresponding computing environments and integrated platforms.

The topics included in a SAD course which should be offered to engineering students during their undergraduate studies should represent an indispensable part of the engineering curriculum. Possessing a strong background in Computing would allow them a flying start in their professional career along with solid knowledge and skills in pursuing graduate degrees.

Foremost, engineers are people of action. Engineering students deserve to be prepared for the challenges of their profession. They should be able to implement their creativity and make their dreams come true by also relying on the computing power.

References

[1] Bäcker, A. *Computational Physics Education with Python.* IEEE Computer Society, Computing in Science and Engineering, May 2007, pp. 30-33.

[2] Glotzer, S. C., B. Panoff & S. Lathrop. *Challenges and Opportunities in Preparing Students for Petascale Computational Science and Engineering.* IEEE Computer Society, Computing in Science and Engineering, September 2009, pp. 22-27.

[3] Hasna, A.M. *E-competence in chemical engineering learning and teaching*. 2nd International Conference on Education Technology and Computer (ICETC), Volume 4, 2010, pp. V4-330 - V4-336.

[4] Jermann, W. H., *Using computers in an engineering curriculum. Where are we? Where are we going?* Frontiers in Education, FIE 26th Annual Conference, Proceedings volume, 1996, pp. 1402-1404.

[5] Kelly, D, S. Smith & N. Meng. *Software Engineering for Scientists.* IEEE Computer Society, Computing in Science and Engineering, September 2011, pp. 7-11.

[6] Kral, J. & M. Zemlicka. *Engineering Education - A Great Challenge to Software Engineering.* ICIS 08. Seventh IEEE/ACIS International Conference on Computer and Information Science, 2008, pp. 488-495.

[7] Lutz, R. *Software Engineering for Space Exploration.* IEEE, Computer, September 2011, pp. 98-101.

[8] MacBride, G., E. L. Hayward, G. Hayward, E. Spencer, E. Ekevall, J. Magill, A. C. Bryce & B. Stimpson. *Engineering the Future: Embedding Engineering Permanently Across the School–University Interface.* IEEE Transactions on Education, Volume 53, Issue 1, 2010, pp. 120 – 127.

[9] Meyer, B. *Software Engineering in the Academy.* IEEE, Computer, May 2001, pp. 28-35.

[10] Qumer, A. & B. Henderson. *An evaluation of the degree of agility in six agile methods and its applicability for method engineering.* Information and Software Technology, Volume 50, Issue 4, March 2008, pp. 280-295.

[11] Rashid, A. J. Weckert & R. Lucas. *Software Engineering Ethics in a Digital World.* IEEE, Computer, May 2009, pp. 34-41.

[12] Rhodes, D. H. *Systems engineering: an essential engineering discipline for the 21st Century.* ICSE, Proceedings of the 24rd International Conference on Software Engineering, 2002.

[13] Sendlinger, S. C., D. J. DeCoste, T. H. Dunning, D. A. Dummitt, E. Jakobsson, D. R. Mattson & E. N. Wiziecki. *Transforming Chemistry Education through Computational Science.* IEEE Computer Society, Computing in Science and Engineering, September 2008, pp. 34-39.

[14] Welch, H.L. *Teaching a service course in software engineering.* IEEE, 37th Annual Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports, 2007, pp. F4B-6 - F4B-11.

[15] Zlatarova, F. *Diversity in the Development of Computing Projects.* ASEE, Spring 2007 Middle Atlantic Conference Proceedings, 2007, http://www.asee.org/papers-and-publications/papers/section-proceedings/middle-atlantic/spring-2007.