ISSN: 2167-1435
v31 n3082

Teaching Tip: An Innovative Approach to Learning about the Software Selection Process

Stefan Tams
stefan.tams@hec.ca
Department of Information Technologies
HEC Montréal
Montréal, Québec, Canada

Abstract

When it comes to software selection, assurance of learning outcomes is a challenge for many IS faculty. Textbooks often treat the software selection process in a way that necessitates simple "learning by heart" and, accordingly, students regularly complain that the subject matter be neither thought-provoking nor interesting. For the time being, the treatment of the topic in our textbooks is what it is, begging the question of what we as IS faculty can do to improve learning outcomes related to software selection. One promising approach is to improve the appeal of related assessments and the learning process associated with them. Specifically, instead of asking students to simply regurgitate textbook knowledge in an exam, we can give them interesting assignments that provoke thoughts on software selection. Such assignments would be more interesting for the students and could improve learning outcomes at the same time. In line with this notion, the present teaching tip promotes the idea of teaching the software selection process in a more innovative manner, and it enables such teaching by providing suggestions for an innovative assignment. Initial student feedback indicates that the proposed assignment is more interesting for the students and yields better learning outcomes.

Keywords: Software Selection, Innovation, Standard Software, Requirements, Assignment.

1. INTRODUCTION

Peter, Paul, and Mary, three undergraduate students majoring in IS, understand the importance of learning how to evaluate software systems in terms of their potential costs and benefits. Nonetheless, the three IS students just cannot bring themselves to fully understand the software selection process based on their textbooks only, and they feel like they've learned little about this process after they regurgitated their textbook knowledge in the corresponding quizzes and exams. The students feel like they would have benefitted more from applying the corresponding concepts to a real-world software system and business problem.

This vignette illustrates a common problem when asking undergraduate IS students to learn about the software selection process: their disheartenment and their lack of interest in the corresponding material (Strong et al., 2006). Based on the experience of my colleagues and me, students face particular difficulty understanding the concepts of functional and non-functional requirements, and they often do not fully understand how to evaluate those requirements with respect to a specific business problem. Generally, student learning can be enhanced by administering the right assignments; meaningful assignments can greatly improve student motivation and associated learning outcomes (Lending, 2010). Yet, teaching tips are lacking that propose useful assignments in the area of software selection. Hence, *our objective with this teaching tip is to provide IS faculty with an assignment they can use to teach software selection in a more interesting, applied, and effective manner.*

Since the detailed specification and effective evaluation of functional and non-functional requirements is key to selecting software systems that are suitable to solve a given business problem in the real-world (e.g., Howcroft & Light, 2010; Rasmussen et al., 2002; Tams, 2008), our proposed assignment focuses on these requirements and on how to evaluate them with respect to a given business problem.

This teaching tip proceeds as follows. The next section provides a background on functional and non-functional requirements as well as on their interrelationship. Thereafter, the proposed assignment is introduced, followed by the assignment mechanics (i.e., grading, class and group sizes, assignment flexibility, and additional resources). Finally, the teaching tip discusses student reactions to the assignment, and it offers some concluding thoughts.

2. ASSIGNMENT FOCUS: FUNTIONAL AND NON-FUNCTIONAL REQUIREMENTS

Evaluating both functional and non-functional requirements is a critical aspect of effective software selection (Howcroft & Light, 2010; Rasmussen et al., 2002; Tams, 2008). Functional requirements are software features related to the different functions a system can carry out for a user, while nonfunctional requirements relate to the acquisition and operation of a software system, not to what the system is supposed to accomplish for the user (Gebauer et al., 2008; Rodrigues et al., 2005). Accordingly, functional requirements comprise, but are not limited to, software functionality, security, performance, and compatibility. The first functional requirement, software functionality, involves that a software system has the functions necessary to accomplish for a user what the user needs to have accomplished (Gebauer et al., 2008). Next, software security implies that the functions and data of a software system can be accessed only by authorized personnel. This requirement is functional since security these days is key; for example, an online vendor like Amazon would not be entrusted credit card information if it would not have advanced security measures. Hence, the very purpose of a software system can be compromised if the system is not sufficiently secure (Burston, 2003). Likewise, software performance, such as response time, is often linked directly to the purpose of a system. For example, consumers are unlikely to buy from an online vendor whose Web site needs to “think” for 20 seconds each time when they click on a button. Finally, system compatibility entails that a software system is integrated with an organization’s other software and hardware components.

Non-functional requirements serve to complement the functional ones. They include, but are not limited to, the maturity of a software, the financial stability of a software vendor, the vendor support offered in terms of user training, the technical support offered by a vendor, and the price of a software system. These requirements are not linked directly to the purpose of a software system, but they are necessary to put the system to an effective use (Gebauer et al., 2008; Rodrigues et al., 2005). Software maturity entails that a software system should be mature enough to no longer suffer from any initial programming-related bugs. A good proxy for software maturity is the version number of a system; a high version number generally points to a mature product with few remaining bugs (Burston, 2003). The financial stability of a software vendor is a crucial element in ensuring enduring support from the vendor; a financially unstable vendor might go bankrupt at any time after a customer has purchased a software product and, as a result, could become unable to keep delivering important updates and other forms of support for the product (Rasmussen et al., 2002). Further, it is important that a software vendor can offer sufficient support in terms of user training. Such training can help ensure that the users have both the intention and confidence necessary to adopt a system and that they are able to use the system effectively. A related, relevant criterion is technical support (Gebauer et al., 2008). Software vendors must, generally, assist a client with various aspects surrounding the implementation as well as the continuous operation and maintenance of a software system. Last but not least, the price must be in an appropriate range for a potential client. It is key that a software system does not exceed a client organization’s available budget (Rodrigues et al., 2005).

Figure 1 puts the functional and non-functional requirements into context; the non-functional requirements are at the periphery, surrounding the functional ones that are core (Rasmussen et al., 2002). This representation of the different requirements does not suggest that non-functional requirements are not important, only that they are not directly linked to the purpose of a software – in contrast to functional requirements. The figure also demonstrates that functional and non-functional requirements have a complementary relationship; often, meeting functional requirements is of limited value when the non-functional requirements are not met. For instance, even a software system that can perfectly accomplish for an organization what the organization needs to have accomplished is of limited value for the organization if it exceeds the available budget so that the organization cannot afford its purchase. The same holds true if the system suffers from substantial problems, such as bugs, related to software immaturity.

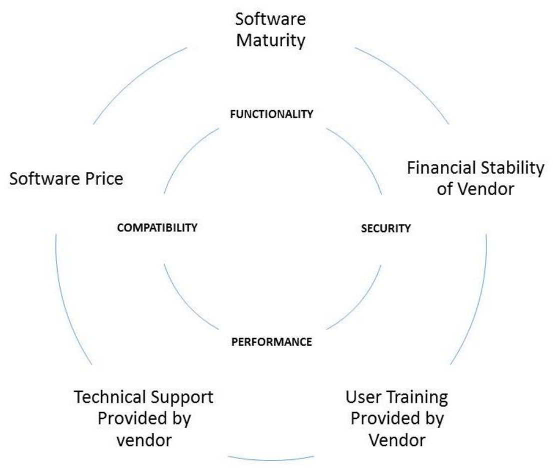
Software Maturity
FUNCTIONALITY
Financial Stability of Vendor
Software Price
COMPATIBILITY
SECURITY
PERFORMANCE
Technical Support Provided by vendor
User Training Provided by Vendor

Figure 1. Functional requirements (inside) and non-functional requirements (outside)

In summary, functional requirements are necessary software features that relate to the different functions a software system can perform for a user (i.e., they relate to the question of whether the system can do for a user what the user needs to have done) (Gebauer et al., 2008), while non-functional attributes relate to the acquisition and operation of a software system.

3. THE ASSIGNMENT

Assignment Preparation

To better understand what functional and non-functional requirements are, how they can be evaluated, and how the evaluation can be presented to organizational decision makers, the students evaluate a software system as part of an assignment. For this purpose, the instructor divides the class into groups and, subsequently, assigns different software systems to the various different groups (e.g., Group 1 is assigned the system Maconomy, Group 2 is assigned the system Sage, and Group 3 is assigned the system Salesforce). Next, the groups assume the roles of independent consulting firms, such as Pricewaterhouse Coopers or KPMG, and the instructor assumes the role of a decision maker from a client organization asking them to evaluate a certain software solution on his/her behalf. More specifically, the instructor provides the students with the following, brief description of the client organization:

Greatcorp Inc. is a medium-sized manufacturing organization headquartered in North America with plants located in the United States, French Canada, Germany, Spain, France, Greece, Italy, China, India, and Turkey. Greatcorp is in need of a material requirements planning (MRP) system with an added dashboard that features advanced security measures, fast response times, high scalability, and compatibility with the existing Apple infrastructure. The company has a very small IT team and IT helpdesk that are, both, unexperienced with MRP systems and dashboards. Similarly, the company's employees have no prior experience with such systems. The budget for the purchase has been set to $ 250,000.

In a first step, all groups of students gather the information necessary to describe the systems assigned to them in terms of their functional and non-functional requirements (for example, they describe what types of functions Salesforce offers, what its compatibility characteristics are, and how much it costs). The groups can rely on various sources, such as the software vendor's website and online forums, to fully describe their respective systems in terms of these attributes. Once the descriptions are completed, the groups can derive recommendations from their descriptions as to whether the systems are useful for Greatcorp's purposes. For example, given that Greatcorp is a multinational organization with plants located around the world, vendor support and user training should be offered in multiple languages (i.e., English, French, German, Spanish, Greek, Italian, Chinese, Hindi, and Turkish). Furthermore, since the company has very limited IT expertise and its employees have no prior experience with MRP systems, the support offered by the vendor must be both extensive and multifaceted.

Overall, this approach asking the students (1) to fully describe a system in terms of its attributes and, then, (2) to compare that description to the depiction of Greatcorp's needs in order to (3) derive conclusions about the software's usefulness for Greatcorp necessitates the students to fully understand the concepts of functional and non-functional requirements, and it asks them to think through the consequences of a system's characteristics for a specific business need.

Assignment Presentation
Once the groups' evaluations of their respective software systems are complete (i.e., the evaluations include the description of the functional and non-functional requirements as well as the recommendation as to whether a given system is useful for Greatcorp and why), the groups present their evaluations in two forms: a Software Selection poster and a video. As regards the former, the groups prepare a 36" x 60" poster to present their evaluations during a poster session (much like poster sessions at academic conferences). The students will ask why they should prepare a Software Selection poster and not a presumably more modern PowerPoint presentation. The answer that instructors can give to this question is simple: because posters and PowerPoint slides serve different purposes in organizations. PowerPoint slides are used for presentations that follow a Push principle, implying that the presenters push the information they deem relevant onto the audience. As such, PowerPoint presentations follow a "talk and listen" structure, often with questions being asked only at the end of the presentation. This format is useful for larger audiences, where interaction must be limited so that the presenter can cover all the relevant content. By contrast, poster presentations follow a Pull principle, implying that individual members of the audience drive the presentation with their questions about the content being shown on the poster. Hence, in contrast to PowerPoint presentations, poster presentations are more suitable for smaller audiences, and they have the added benefit of fostering interaction, as the interaction between an independent consultant from Pricewaterhouse and a decision maker from a client firm like Greatcorp.

During the Software Selection poster session, each student group generally receives useful feedback from the instructor as well as from the other student groups. To make sure that the students learn from this feedback and do not disregard it, they will be asked to incorporate it in the second part of the assignment: the Software Selection video. The video should be about five minutes long, and it can be submitted to the instructor in the form of a commented PowerPoint presentation (the students can also use Microsoft Photo Story 3; in any event, they should not be asked to film each other with a camera). Preparing the video by incorporating the feedback from the poster session allows the students to deepen their understanding of the software selection process further.

Overall, preparing the Software Selection poster and the video will allow the students to learn about the software selection process in a relatively innovative way, especially when compared to more traditional means such as exams, written reports, or the PowerPoint presentation format used in most of their other classes. On this basis, the students have an interesting, thought-provoking, and rich learning experience about software selection.

4. THE ASSIGNMENT MECHANICS

Grading
As indicated by Harris and Rea (2009), it can be difficult to grade group assignments in class because it can be challenging to determine the contributions of the individual group members. Hence, consistent with Lending (2010), in this assignment the emphasis is placed on the creation of well thought-through Software Selection posters and videos rather than on evaluating the quality and quantity of the work of individual students. In other words, the emphasis is on the final products of the assignment. Nonetheless, all students fill out worksheets showing what they contributed to the final products and when so that their individual contributions can be assessed if desired. Although primitive, this method is very effective to determine student grades since students seldom over-report their work (Lending, 2010). To ensure that students view this assignment as an opportunity to learn about software selection rather than merely as a

means to obtain a good course grade, the points given for the Software Selection poster and video together should not exceed 15 percent of students' course grades.

As regards the Software Selection poster, the student groups are asked to evaluate each other's posters during the poster session to deepen their understanding of the software selection process. In other words, while each student group will have already learned about the initial steps of the software selection process by preparing its respective Software Selection poster, it will deepen its understanding of this process further during the poster session by examining and evaluating the posters from all of the other groups. Evaluating the posters from all of the other groups will help a group of students engage with the software selection process beyond the efforts that went into the development of their own poster. In accordance with this learning objective of the poster session, the grades of all student groups will be determined by (1) the instructor's evaluation of their posters, (2) the average of the evaluations a group has received from all of the other groups, and (3) the fact that a group has submitted its evaluation sheet (which evaluates all the other teams) to the instructor. The instructor's evaluation should have a weight of approx. 43%, the average evaluation a group has received from all of the other groups should have a weight of approx. 43% also, and the submission of the evaluation sheet to the instructor should have a lower weight of approx. 14%.

Concerning the Software Selection video, the students are asked to incorporate what they learned from the poster session; hence, the grading of the video should be more rigid than that of the Software Selection poster. We recommend the following grading structure:

- 25% for the description of the functional attributes
- 25% for the description of the non-functional attributes
- 25% for the conclusions regarding the software's usefulness for Greatcorp
- 25% for the professionalism of the video.

Points can be deducted if the video exceeds the recommend duration of five minutes.

Group and Class Sizes
The assignment has been tried with as few as six groups and with as many as sixteen across seven to nine sections of the same class over multiple semesters. If there are fewer than six groups, the assignment is less interesting for the students and their learning experience is less rich. If there are too many groups, some aspects of the assignment become overly repetitive, resulting in similar problems. Hence, we recommend the assignment for class sizes large enough to allow for the creation of between six and sixteen groups. Concerning the number of students per group, we recommend between three and five students. If there are less than three students in a group, the assignment quickly becomes too complex, and the students lose interest. If there are more than five students, the coordination effort increases so that the students lose interest also.

Assignment Flexibility
The assignment can easily be modified to address specific teaching needs in greater detail. For example, the aspect of system security is treated rather generically in the example above. This aspect can be specified in greater detail in the description of the client organization's needs. For example, one could ask the students to think through the security aspects of open source software (AlMarzouq et al., 2005) or meso-level applications (Beebe & Rao, 2010). The same holds true for the other functional and non-functional attributes. The assignment is very flexible in that respect and can be used for various teaching needs.

Additional Resources
A book chapter detailing the roles of functional and non-functional attributes in the software selection process can be obtained from the authors. Similarly, a Microsoft Excel file can be obtained from the authors that includes grading rubrics. Finally, the authors can provide a slide on the costs and benefits of poster presentations and PowerPoint presentations that can be included in a regular lecture to prepare the students for the assignment.

5. STUDENT REACTIONS

The assignment was administered in between seven and nine sections of a class over multiple semesters from January, 2012 to Mai, 2014. Student responses to the assignment were generally positive and included:

- "I learned a lot,"
- "The assignment was very helpful to understand how to select software,"

- "I learned a lot about effective communication from presenting the poster,"
- "I loved that we could use what we learned from the poster when making the video, this really helped me understand the material," and
- "This assignment was different and exciting, I feel like I learned a lot from it."

Overall, students have generally been quite positive about their learning experiences with this assignment. We received very few negative comments, often related to the inconvenience imposed by the creation of the poster and video. Creating those was sometimes perceived as inconvenient because the students are so used to giving PowerPoint presentations that they had difficulty seeing the value of such seemingly "old" forms of presentations as posters. However, pointing out the aspects discussed in Section 3 of this paper alleviated the students' concerns.

In addition to these student reactions, it deserves mentioning that problem-based learning as advocated in this paper generally has several benefits for students, including deeper understanding, higher student satisfaction, higher self-efficacy, and better lifelong learning skills (Weimer, 2009).

6. CONCLUSIONS

Software selection in general, and functional as well as non-functional requirements in particular, are important elements of the IS curriculum (Topi et al., 2010). To teach these aspects effectively, we as IS faculty must find innovative means of teaching that our students perceive as interesting and that allow them to deepen their understanding of the subject. With the right assignments, we can help our IS students move from unconvinced participants, who perform an assignment because they have to, to active and excited participants who learn to appreciate the class concepts and to apply them effectively (Lending, 2010). Hence, we have taken a first step toward helping IS students like Peter, Paul, and Mary (our opening vignette) appreciate and understand the software selection process, and we hope that future work continues to develop material in this important area.

7. ACKNOWLEDGEMENTS

We would like to thank Jacques Robert, Gilbert Babin, and Ana Ortiz de Guinea for the development of the assignment. Furthermore, we would like to thank the thousands of students who have participated in this activity over the years. Their enthusiastic responses to the assignment have encouraged us to continue using it and improving upon it.

8. REFERENCES

AlMarzouq, M., Zheng, L., Rong, G., & Grover, V. (2005). Open Source: Concepts, Benefits, and Challenges. *Communications of the Association for Information Systems*, Vol. 16, Article 37.

Beebe, N., & Rao, V. S. (2010) .Improving Organizational Information Security Strategy via Meso-Level Application of Situational Crime Prevention to the Risk Management Process. *Communications of the Association for Information Systems*, Vol. 26, Article 17.

Burston, J. (2003). Software Selection: A Primer on Sources and Evaluation. *CALICO Journal*, 21(1), 29-40.

Gebauer, J., Tang, Y., & Baimai, C. (2008). User requirements of mobile technology: Results from a content analysis of user reviews. *Information Systems and e-Business Management*, 6(4), 361-384.

Harris, A., & Rea, A. (2009). Web 2.0 and Virtual World Technologies: A Growing Impact on IS Education. *Journal of Information Systems Education*, 20(2), 137-144.

Howcroft, D., & Light, B. (2010). The Social Shaping of Packaged Software Selection. *Journal of the Association for Information Systems*, 11(3), 122-148.

Lending, D. (2010). Using a wiki to collaborate on a study guide. *Journal of Information Systems Education*, 21(1), 5-13.

Rasmussen, N. H., Goldy, P. S., & Solli, P. O. (2002). *Financial Business Intelligence: Trends, Technology, Software Selection, and Implementation*, Wiley: New York.

Rodrigues, G., Rosenblum, D., & Uchitel, S. (2005). *Using scenarios to predict the reliability of concurrent component-based software systems*. In M. Cerioli (Ed.), Fundamental Approaches to Software Engineering, pp. 111-126.

Strong, D., Fedorowicz, J., Sager, J., Stewart, G., & Watson, E. E. (2006). Teaching with Enterprise Systems. *Communications of the Association for Information Systems*, Vol. 17, Article 33.

Tams, S. (2008). Business intelligence applications in personnel controlling [*Personalcontrolling mit Business Intelligence*], VDM: Saarbruecken.

Topi, H., Valacich, J. S., Wright, R. T., Kaiser, K., Nunamaker, Jr., J. F.; Sipior, J. C., & de Vreede, G. J. (2010). IS 2010: Curriculum Guidelines for Undergraduate Degree Programs in Information Systems. *Communications of the Association for Information Systems*, Vol. 26, Article 18.

Weimer, M. (2009). *Problem-Based Learning: Benefits and Risks*. In: Faculty Focus: Higher ED Teaching Strategies from Magna Publications, http://www.facultyfocus.com/articles/effective-teaching-strategies/problem-based-learning-benefits-and-risks/, accessed October 2nd 2014.

ISSN: 2167-1435
v31 n3082

This page could not be removed from the template. I assume it will be removed during the copy-editing process. Thank you.