https://doi.org/10.48009/2_iis_2011_9-12

Issues in Information Systems

Volume XII, No. 2, pp. 9-12, 2011

ADDRESSING THREE SERIOUS GAPS IN SYSTEMS ANALYSIS AND DESIGN TEXTBOOKS

Yousif Mustafa, Simplex Systems, Inc., Detroit, MI 48202, drymustafa@gmail.com

ABSTRACT

It is a fact that the Systems Analysis and Design course is the pinnacle of any IS/CIS/MIS curriculum. Accordingly, one would expect the quality of the textbooks used for this course to match that level of importance. However, after two decades of teaching Systems Analysis and Design and using a wide array of textbooks, we have identified a good number of serious defects and shortcomings in almost all the popular textbooks in use. These defects range from poor introduction of the subject to students, bad presentation of the material, inaccuracy, and illogical order of presentation, to incomplete information. The purpose of this research paper is to outline a few major defects and provide solutions. Additionally, we also hope to appeal to our colleagues in order to build a consensus. It is critical that we pay attention to these problems and voice our concerns to the authors and publishers to remedy the situation in future editions of their textbooks. This paper raises three major issues namely finding the root-cause of a problem, the need for a domain-expert and a process-expert.

Keywords: root-cause, domain-expert, process-expert, systems analyst, SDLC

INTRODUCTION

A Systems Analysis and Design course is the pivotal component in every IS/MIS/CIS/BIS curriculum in our colleges and universities. It is where senior students learn to combine and implement a wide variety of technical, managerial, personal, and, most importantly, problem-solving skills. They also learn how to communicate, organize, present and appeal to others. In short, the systems analysis and design course, supposedly, is developed to prepare our students to be problem solvers. Accordingly, that mission-critical course must be meticulously designed to utilize, among other things, high quality textbook(s). Clarity, logical flow of information, accuracy, organized presentation of the materials, and depth of knowledge are prerequisites for such high level textbooks. Good, meaningful, and relevant examples must also be used to convey an idea. A simplified approach is definitely different from a primitive one. Unfortunately, we found more than a handful of textbooks which have chosen to use a simplistic primitive approach. Fictional personal or telephone dialogues between imaginary entities are presented to students in order to explain a concept. A frequently used example would be that Kathy (HR Director) asks John (IT staff) to produce a weekly report showing the overtime hours worked last week. In turn, John asks Kathy for a couple of days to figure it out. The above example depicts a low-level primitive mode of writing, which is totally unacceptable and it is nothing but a mere insult to the intelligence of our students. Our senior MIS/CIS students, as future leaders, are much more intelligent and capable of understanding a concept without using these silly fictional stories. This approach may have worked forty years ago, however, it is entirely inappropriate for the Twitter and Facebook driven generation. Nowadays, a 16-year old Facebook savvy, Egyptian girl mobilized hundreds of thousands of demonstrators in downtown Cairo and created an influx of world-class revolutions. We truly believe that the previously mentioned approach tarnishes the quality of this course. We urgently and I call upon our colleagues, authors of these books, to depart from this mode of writing.

This paper, however, will not address most of these issues, Rather, it will focus only on three major defects which we have identified based upon our two decades of teaching Systems Analysis and Design. The paper is definitely not a criticism of any specific author, textbook, or publisher, rather a presentation of a constructive critique. Our goal is to build a consensus among faculty in order to appeal to authors to change directions before it is late. Exploring issues like these might raise a few eye-brows, however, we have a firm belief that our students' education and their interest are above any other consideration. We will highlight the defects in an anonymous approach as much as possible. Therefore, the author will avoid, for obvious reasons, direct citations within the paper body. A partial list of the most commonly used textbooks is provided at the end of this article.
The three major defects which will be discussed in this paper are:

1. The complete absence of any methodology to define a problem and find its root cause.
2. The complete absence of the concept of Domain Expert.
3. The absence of the concept of process and process management.

Each of these defects will be detailed in the next section along with our proposed solutions for each. A conclusion will be drawn at the end.

THREE MAJOR DEFECTS

We have carefully examined the most popular Systems Analysis and Design textbooks authored and published within the past decade and have identified the following three major defects:

1. Absence of a Methodology to Define a Problem and Identifying its Root Cause:

All the textbooks emphasize the concept of a problem definition and present it as the corner stone in solving any problem. Intuitively if you couldn't understand and describe a problem you wouldn't be able to solve it. None, however, provide a formal or informal method (or exercise) to show students how to define and conceptualize a problem. Moreover, none but one [5] mentions how to go one step further to find the root cause of the problem. That single book briefly addresses the Fishbone and Pareto charts as two tools for finding the root cause of a problem. Almost, all textbooks always mention the root cause of the problem, but none provide either a methodology or a tool to do so. Subsequently, a detailed, real-life example to show students the logic and tools to diagnose the root cause is totally nonexistent.

Proposed Solution: Dedicating an entire chapter to explain how to define a problem correctly and accurately. The chapter should present a few methodologies to formulate and conceptualize the problem along with some examples and exercises. The chapter may also provide the basic knowledge and techniques, such as the 5Y's to identify the root cause of a problem, as well as, methodologies such as Root Cause Analysis (RCA). A list of available tools for identifying the root cause such as the Run Chart, Fishbone diagram, Tree diagram, and Pareto charts, along with the advantages and disadvantages of each, should also be discussed. A detailed real-life example in which one of the tools is implemented in order to find the root cause should also be a part of this chapter. The chapter should be taught within the early steps of the Analysis phase of the SDLC.

2. Absolute Absence of the Domain Expert:

A systems analyst is a technical person first and foremost; however, he/she is expected to have acquired some basic knowledge of a few business domains. Practically and theoretically speaking, he/she could not be an expert in the auto, banking, health care, retail, or service industries simultaneously. Thus, the intuitive immediate questions would be: "How could a systems analyst efficiently and correctly design a solution (to solve a problem on hand) and design the fine-grained processes for a domain which he/she has no prior exposure to?", "How could a systems analyst who spent most of his/her career in the auto industry, for example, be an effective problem solver in the healthcare domain?". A systems analyst would certainly not become an expert in health care by simply going through the company's documents or interviewing a few people in the company. A Domain expert would have had years of accumulated knowledge and experience, which goes far beyond company documents and databases about the domain.

All the textbooks strongly emphasize engaging and involving the users in the entire SDLC. However, involving and engaging users have always meant approving, disapproving, or suggesting a few ideas by the users. It never meant that users actually identify and design the processes for the new system. The design of the processes of the new system has always been the burden of the systems analyst. As such, there is a compelling need for a person on the team with a deep and thorough knowledge of that domain, a "Domain Expert". A life insurance "Domain Expert", for example, would know exactly the required steps to issuing a policy, the sequence of those steps, their input/output, and constraints.

Proposed Solution: Recognizing and emphasizing the role of a "DOMAIN EXPERT", an expert in the subject domain, as an essential member of the Systems Analysis and Design team. The Domain Expert might be the

systems analyst himself/herself, or someone else chosen among users and by users. The role of the Domain Expert should be emphasized and never hidden or lumped in the users group. His/her role is to identify the correct processes, their right sequence, their I/O, and conformance to the business rules of the subject domain. The Domain-Expert should be clearly highlighted when listing the composition of the Systems Analysis and Design team.

3. Complete Absence of the Concept of Process Expertise:

After identifying the processes, their correct sequence, and validating them against the business rules, those processes should be examined for efficiency. Process Management or Process Engineering goes beyond drawing the DFD's and should be part of the process of Systems development. Completing and validating the DFD's don't necessarily produce efficient software. Yet, most of the existing systems analysis and design textbooks neither examine this subject nor even raise the issue. The concept of process management or process engineering is totally non-existent.

Proposed Solution: Introducing the concept of a process, process management, or process engineering. A basic foundation, with examples, defining the process, its I/O, efficiency, and types of processes should all be presented to students. Immediately after finalizing the DFD's, students should be taught how each process is subjected to an efficiency test. The following is only a sample matrix that should be applied to each process on the final DFD's:

1. Number of inputs: A good process should not require a large number of inputs in order to function. A large number of inputs will result in slowing the process and making it difficult to debug and fix. Such a process may need to be split into several smaller processes in order to distribute the inputs among them.
2. Complexity: A process must be simplified to include a reasonable number of activities. A process with a complex algorithm would be harder to implement, takes a longer time to execute, and would be more difficult to be improved, debugged, or repaired. The process should be broken into smaller sequential processes.
3. Time Duration: Lengthy processes are also hard to debug and fix and they should be broken down into several processes, which would take shorter times to execute.
4. Resources: A resource-hungry process should be reviewed and simplified instead of tying up many resources for one process.

These issues should be introduced to our students in a reasonable depth and clarity during both the Analysis and design phases.

CONCLUSION

Our proposed solutions are not intended to provide full-blown detailed chapters, which would be ready to be plugged into a textbook in time for the next edition. Nevertheless, they can be used as a basis for the missing topics. We are simply pointing to the fact that there are a few defects in the existing generation of Systems Analysis and Design textbooks. This paper is a "wake-up" call" to draw the attention of faculty, authors, and publishers that our students are not provided with the knowledge and skills set required for problem solving. Without exceptionally mastering problem solving, our students will never be able to compete against someone with the same skill set who would be happy making $10 a day. We can generate C++ code, for example, using a CASE Tool, but solving a problem is a human intellectual skill that cannot be substituted.

We believe that addressing the above discussed issues would be of great benefit to our students and the respective industries they would work for after graduation.

There are so many other defects that this paper did not address due to space and time limitations.

Following is a brief list of topics that we excluded for future investigation.

1. None gives more than 4-5 lines, at best, to distinguishing between Systems Analysis and Design and Software Engineering.
2. None of the existing textbooks demonstrates a systematic method, or a real-life example, showing our students how to calculate the cost or time for developing a software system.
3. Most of the existing books rehash those silly primitive conversations between fictional entities repeatedly in every edition. This only result in reducing our students' intellect as well as lowering their level of thinking and

imagination. Rather than challenging our students to aim higher and higher, some textbooks have chosen to use those stupid conversations between (Sue) and (Jim) to convey a message.

REFERENCES

1. Dennis, A., Wixom, B., and Roth, R. (2009). Modern Systems Analysis and Design, 4th Edition, John Wiley & Sons.
2. Hoffer, J., George, J., and Valacich, J. (2011). Modern Systems Analysis and Design, 6th Edition, Prentice Hall.
3. Hoffer, J., George, J., and Valacich, J. (2005). Modern Systems Analysis and Design, 4th Edition, Prentice Hall.
4. Kendall, K, and Kendall, J. (2011). Systems Analysis and Design, 8th Edition, Prentice Hall.
5. Shelly, G, and Rosenblatt, H. (2010). Systems Analysis and Design, 8th Edition, Course Technology.
6. Valacich, J., George, J., and Hoffer, J. (2009). Essentials of Systems Analysis and Design, 4th Edition, Prentice Hall.
7. Whitten, J., and Bentley, L. (2007). Systems Analysis and Design Methods, McGraw-Hill.