

Using Polya to Teach System Development Methodologies: Fostering a Role Perspective in IS Students

J. Harold Pardue Michael V. Doran Herbert E. Longenecker University of South Alabama

Abstract

The role of IS is changing from a task perspective to a role perspective focused on aligning IS with business needs and goals. In this paper we present a problem solving perspective for teaching systems development methodologies as a means of fostering this shift in perspective in IS students. Polya's formal problem solving process is synthesized with the traditional systems development life cycle. This synthesis makes explicit the tacit knowledge embedded in system development methodologies and enables the transfer of domain knowledge to the broader organizational context and the alignment of IS with business needs and goals.

Keywords: Polya, Problem solving, System Development Methodologies

1.

INTRODUCTION

Beginning with the advent of computerized organizational information processing in the 1960s, the complex interplay between the rapid advancement of information technology (IT) and the business environment has led to profound changes in how organizations operate and the role of the IS professional. A study of critical skills and knowledge requirements of IS professionals by Lee, Trauth, and Farwell (1995) found that the most critical future IS activity will be aligning IS with business needs. As IT - based business systems continue to become more complex and integrative, it becomes increasingly important for IS professionals to relate business problems with IS solutions. The IS professional must shift from a "task perspective," which fosters a technical orientation, to a "role perspective" that is aligned with business goals. Rather than a task oriented focus, the IS professional must emphasize the relationship between IS and the user community. Ultimately, this shift must be reflected in IS curricula if we are to prepare IS graduates for successful IS careers.

How can IS educators begin the process of affecting this shift in focus toward a role perspective? We believe the answer lies, in part, in how IS educators present one of the essential "tasks" IS professionals perform, namely systems development.

The Systems Development Process

As organizations continue to increase their level of dependence on IT for their mission critical systems, IS

professionals are called upon to develop increasingly complex and integrated systems. To develop these systems, IS professionals employ various systems development methodologies (SDM). The three most common SDMs, Software Engineering (SE), Information Engineering (IE) and Rapid Application Development (RAD), prescribe a life cycle approach to systems development. The basic tasks of systems development are modeled as an iterative, top-down series of phases or steps. Texts on Systems Analysis and Design use a slightly different life cycle model for the systems development process, some with as many as twenty identifiable phases. We use the life cycle models listed in a prior study by Granger (1995). For RAD, we use Martin's (1991) RAD life cycle. The life cycle phases for SE, IE, and RAD are reproduced in Table

2.

CHALLENGES FOR IS EDUCATORS

IS educators face at least two challenges to fostering a "role perspective" in IS graduates. The first challenge is to enable students to see the systems development process in a broader perspective. The systems development process is derived from principles of software engineering and as such, is largely an engineering task. Historically this engineering focus has encouraged an insular, task oriented perspective and the perception of the user community that the IS community is unresponsive and narrowly focused (Keen, 1988). The challenge for the IS educator comes in maintaining an emphasis on how the systems development process relates to users. In learning detailed steps and techniques of a methodology, it is easy for students to lose

Table 1

Systems Development Life Cycles

LIFE CYCLE STAGES

Logical Design Phase Project Inception Requirements Gathering Requirements Planning Systems Analysis System Design Specification Review Business Modeling Process Modeling Enterprise Modeling Logical Design Review

Physical Design Phase Database Design User Design Programming Construction Cutover Systems Testing Production Turnover Project Production

sight of the overall business goal or need that motivated the process.

A second challenge is to prepare IS graduates to train and counsel the user community in the use of systems development methodologies, a major activity in the new role of the IS professional. It is insufficient for IS graduates just to be skilled and competent system developers. With the rapid growth and diffusion of personal computers, off-the-shelf software and visually-oriented development environments, and the shift from mainframe computing to the network-based, client-server computing, the user community is increasingly assuming the task of systems and application development (Farwell, Kuramoto, Lee, Trauth, & Winslow, 1992). It then becomes the responsibility of the IS professional, as the systems development expert, to educate the user community in the proper use of system development methodologies and the disciplined adoption of software engineering principles. In this new environment, the role of the IS professional shifts to that of an internal consultant (Sullivan- Trainor, 1988, Livingston, 1989). A challenge for the IS educator lies in both teaching *how* to do systems development and how to *teach* how to do systems

SE

X X

X X X

X

X

X X X

m

X

X X X X

X

X

X X X

RAD

X

X

X X

development. How then can IS educators prepare IS graduates to a) translate a technically oriented, engineering process into something that the user community can readily learn and apply and b) clearly communicate how this process relates to business goals and needs?

3 .

MEETING THE CHALLENGES

In considering ways in which we could prepare our own students to assume this new role and effectively align IS with business needs we looked for something fundamental or common to both the systems development process and business needs that we could build on. We concluded that the underlying phenomenon common to both the systems development process and business needs is problem solving.

Regardless of the functional area or proposed system, the fundamental activity undertaken is problem solving. This suggestion has added relevance given the fact that the development of problem solving skills has been recognized in recent IS curriculum research as a critical element in IS education (Couger, Davis, Dologite, Feinstein, Gorgone, Jenkins, Kasper, Little, Longenecker, Valacich, 1995).

68

Journal of Information Systems Education

IS'95 calls for IS graduates that are "creative, innovative, entrepreneurial problem solvers." IS'97 repeats this call in the "Exit Characteristics" section on problem solving (IS'97, 1997). IS graduates should be able to use "problem solving models, life cycle stages, and creativity techniques" to solve business problems. The current trend toward information technology integrated operations requires IS professionals play a liaison role within an organization (Livingston, 1989) where they must pull together integrative IS solutions that span multiple functional areas within an organization. We suggest that one way IS educators can meet the challenge of fostering a new "role oriented" perspective in IS graduates is to present the traditional systems development process in the context of a formal problem solving process.

Synthesis of Polya and the Systems Development Process

Polya first published his book entitled "How to Solve it" in 1945 (Polya, 1957). Since that time, the simple yet powerful heuristics he outlined in his four-step method have been used to help students in many disciplines learn the basic process of problem solving and the discovery and invention of solutions. In all, Polya's book contains over 60 heuristics. These heuristics are generic, simple and natural, and express what is fundamental to problem solving in almost any problem domain. Polya's four-step method is expressed in the form of incisive questions such as: "What is the unknown? What are the data? What is the condition?" These questions are grouped into four steps or phases: Understand the Problem, Devise a Plan, Carry Out the Plan, and Look Back and Evaluate. Our synthesis of Polya's four-step method and the systems development process is depicted in Table 2. The following paragraphs briefly describe each step in Polya's methodology and how they relate to the systems development life cycle.

Table 2

Synthesis of Polya and Systems Development Methodologies

Polya

SE

IE

Understand Problem

Devise a Plan

the

Project Inception Requirements Gathering

Systems Analysis

System Design Specification Review Database Design

Project Inception Business Modeling Enterprise Modeling

Process Modeling

Carry Out the Plan

Look Back and Evaluate

Programming

Systems Testing Production Turnover Project Production

Programming

Systems Testing Production Turnover Project Production

RAD

Requirements Planning

User Design

Construction (Development) Cutover

The first step is to *understand the problem*. This expression emphasizes two points. First, students are dealing with a *problem*. This point is perhaps obvious yet easy for students to lose sight of. When students first learn the mechanics of a methodology, they tend to focus on developing proficiency in the methodology, leading to a tendency to view the methodology and the task it embodies as an end unto itself rather than a means to arriving at a

plausible solution to the problem that motivated the task. For example, in our experience it is not initially obvious to students that the symbols in a data flow diagram represent an abstraction of the problem. Students have to be explicitly taught that data flow diagramming represents more than just a process of mapping data flows in an information system, that it is a process of abstraction designed to provide an understanding of the problem. Thus, the second point is that students must *understand* the

Journal of Information Systems Education

problem. It is not enough simply to gather facts and generate charts, diagrams, and dialogues. Students must understand how these diagrams and charts embody a problem that requires a solution and what the nature of the solution should be. Some of the questions Polya suggests for developing an understanding of the problem are: "What is the unknown? What are the data? What is the condition?"

The second step is to *devise a plan*. The focus of this step is the formalization of how the data are connected and how the unknown is linked to the data. The unknown is the solution. The plan lays out a tentative means of moving from the data (the known) to the solution (the unknown). A sample of the questions Polya suggests for discovering how the data are connected and how the unknown is linked to the data are: "Have you seen it before? Or have you seen the same problem in a slightly different form? Do you know a related problem? Could you restate the problem?" Polya emphasizes that the plan is not the solution. The plan is a means (a task) for arriving at the solution. Formulating a plan enables the student to conceive the idea of the solution. In the RAD life cycle, the prototype (User Design) that emerges from the JAD sessions with users acts as the vehicle for conceiving the idea of the solution.

The third step is to *carry out the plan*. Polya makes a subtle distinction in the expression of the third step; students are carrying out the *plan* not the *solution*. The plan outlines a series of tasks for arriving at the solution. The idea is that the output of this step is not an artifact per se, but an artifact designed to achieve some desired goal, namely a solution. In the construction phase of the RAD life cycle, the prototype is validated, thus validating the solution it represents. Further, to arrive at a reasonable solution, Polya emphasizes that each intervening step must be correct.

The final step involves *looking back*. This step involves verifying that the solution is correct. The student must ask whether the solution actually solves the problem. There is always the risk of developing a good solution to the wrong problem. Again, the point is not to produce an artifact, but to produce a solution to a problem. This step is intended to emphasize that a solution is never complete and can always be improved. At the very least, the student's understanding can be improved. The next section discusses how the synthesis of Polya's four-step process and the systems development process can help meet the challenges of fostering a role perspective in IS graduates.

4. DISCUSSION

In this paper we argue that by synthesizing Polya's process and the systems development process, the tacit knowledge embedded in systems development methodologies is made explicit. In this way, students are explicitly taught the underlying abstractions of the systems development process and are therefore better prepared to transfer their SDM

knowledge to broader organizational contexts. Soloway (1986) makes a similar argument for teaching introductory programming. He argues that if students are to transfer the knowledge of programming to other problem-solving domains, they must be explicitly taught the underlying abstractions of programming, not just the methods and syntax. Research in computer science suggests that it is not the syntax and constructs of a methodology or language that represents the major impediment to learning to solve problems and program effectively (Spohrer & Soloway, 1986, Spohrer & Soloway, 1986). Rather, students stumble when attempting to "put the pieces together" (Soloway, 1986) into an integrative, creative solution.

Research has shown (see e.g., Chase & Simon, 1973, Curtis, 1985, Shneiderman, 1980), and everyday experience illustrates that expert problem solvers rely extensively on unconscious strategies and heuristics. This tacit knowledge, sometimes expressed as "intuition" or "a gut feeling," is essential for seeing patterns and trends in an otherwise unfamiliar situation. Unlike novices, experts have the ability to see a current problem as an instance of an old problem that has already been solved. Experts build up libraries of stereotypically solutions that allow them to transfer knowledge from one problem domain to another (Soloway, 1986). It is this transfer of domain knowledge that sparks the conception of a creative and innovative solution. By explicitly teaching students the knowledge implicit in the systems development process, IS educators provide students with a mental language or vocabulary that transcends the systems development process. Students can begin to see how knowledge of the systems development process relates to other knowledge domains and begin building their own mental library of stereotypical solutions. Polya's four-step process shifts the focus from mastery of a "task" to mastery of a "role." That is, students learn to see themselves not as systems developers, but as problem solvers with a particular skill set and specific domain knowledge.

Making explicit the underlying abstractions of the systems development process, Polya's strategies and heuristics provide a natural, simple, and generic lexicon or vocabulary for translating the technical process of systems development into various business domains and at varying levels of user sophistication. This vocabulary provides a basis for articulating an explanation for why and how an IS solution is aligned with business goals. Rather than highlighting the differences between IS and the user community, this vocabulary focuses on what they share in common, namely problem solving.

5. RECOMMENDATIONS

The synthesis outlined in this paper proposes a shift in focus in how educators teach the systems development process. We operationalize this shift by integrating the Learning Units in the 15'97 (15'97,1997) course 15'97.7 - Analysis and Design with Polya's four-step method.

Journal of Information Systems Education

Table 3

Integration of Polya and IS '97.7 -Analysis and Logical Design Learning Units

Learning Unit Number

72

74

77 78 79

Learning Unit Objectives

LO-OIO8

LO-OIO6, LO-OIII

LO-OIO9 LO-OO61 LO-OO52

Competency Levels and Body of Knowledge Elements

32.10.10, 33.8.1

32.10.1,32.10.2,33.4.2

23.5.1,33.9.1 32.2.6,22.10.10 32.3.5,23.9.4

Polya

Understand the Problem

Devise a Plan

81	LO-0118	3 1.6.2, 3 1.6.5, 3 3.9.2, 2 3.9.5	Carry Out the Plan
-----------	----------------	---	---------------------------

82 83 84

LO-OIO2 LO-O206 LO-OII5

23.5.4,33.7.13 23.7.13

33.7.13,23.9.3

Our recommendation for integrating specific Learning Units is summarized in Table 3 and discussed below.

Learning Units 72 and 74 emphasize problem detection, opportunity finding, information gathering, and creativity. The tacit knowledge or strategy underlying the techniques of this step (e.g., JAD and Interviewing) is the goal of understanding the problem, the business need. And Polya's questions, e.g., "what are the data," and "what is the unknown" provide a non-jargon laden vocabulary for communicating effectively with users and stakeholders.

Learning Units 77, 78, and 79 emphasize the development of a shared vision of the proposed solution. Here Polya's questions for moving from the data to the solution, e.g., "Have you seen it before? Or have you seen the same problem in a slightly different form? Can you restate the problem?" bring into relief the tacit strategy underlying design techniques. Namely, the point of these techniques is to formalize a shared vision or plan for moving from the collected data to a solution.

Learning Unit 81 focuses on design and implementation. But when teaching data models (e.g., relational, hierarchical, or object) or programming techniques, educators should reiterate that the point is not to build an application *per se*, but to arrive at a solution to a business

Look Back and Evaluate

need. In other words, the solution follows from the plan formalized in the previous step.

Learning Units 82, 83, and 84 emphasize the development and use of metrics for assessing and assuring quality. Educators teach students how to measure and benchmark complexity, usability, and performance, but underlying the techniques of system development quality assurance is the real need to *look back and evaluate*. Polya's method brings students back to the reason the system was built in the first place, to solve a problem.

In summary, through this synthesis, students learn problem solving strategies and heuristics that transcend the systems development process by making explicit the tacit knowledge embedded in systems development methodologies. An understanding of these underlying abstractions should better prepare IS graduates to transfer their domain specific knowledge to larger organizational contexts and effectively align IS with business needs and goals. The student's focus is shifted from mastering and performing specific tasks, to preparing for fulfilling an organizational role.

6. REFERENCES

Couger, J. D., Davis G. B., Dologite, D. G., Feinstein, D. L., Gorgone, J. T., Jenkins, A. M., Kasper, G. M., Little J. C., Longenecker, H. E., Valacich, J. S., (1995). IS'95: Guideline for undergraduate IS curriculum. MIS Quarterly 19.. (3),341-359.

Chase, W.C., & Simon, H. (1973). Perception in chess. Cognitive Psychology 4, 55-81.

Curtis. B. (1985). Tutorial: Human Factors in Software Development. IEEE Computer Society.

Farwell, D., Kuramoto, L., Lee, D. M. S., Trauth, E., & Winslow, C., (1992). A new paradigm for MIS: Implications for IS professionals. Information Systems Management 9(2), 7-14.

Granger, M., (1995). Which software development method for an information systems curriculum?: Using a cataloging framework with the analytic hierarchy process. International Business Schools Computer Quarterly 6 (3), 32-39.

IS'97 (1997) Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems [WWW document]. URL <http://www.is2000.orc/is2k/rev/Review1.asp>

Keen, P. G. W., (1988). Roles and skill base for the IS Organization. In J. J. Elam, M.J. Ginzberg, P.G. W., Keen, and R. W. Zmud (Eds.), Transforming the IS Organization. ICIT Press, Washington, D.C.

Lee, D. M., Trauth, E. M., & Farwell, D., (1995). Critical skills and knowledge requirements of IS professionals: A joint academic/industry investigation. MIS Quarterly 19 (3), 313-340.

Livingston, D. (1989). The MIS link: The systems integration. Systems Integration 22(4),8364-9342.

Martin, I. (1991). Rapid Application Development, Macmillan Publishing Company, New York.

Polya, G. (1957). How to solve it (2nd ed.). Princeton University Press, Princeton, NJ,

Shneiderman, B. (1980). Software Psychology: Human Factors in Computer and Information Systems. Winthrop Publishers, Cambridge, Mass.

Soloway, E., (1986). Learning to Program = Learning to Construct mechanisms and Explanations. Communications of the ACM 29 (9), 850-858.

Spoherer, J., & Soloway, E. (1986). Novice mistakes: Are the folk wisdoms correct? Communications of the ACM 29 (7), 624-632.

Spoherer, J., & Soloway, E. (1986). Analyzing the high- frequency bugs in novice. In E. Soloway & S. Iyengar (Eds.), Empirical Studies of Programmers. Ablex, New York.

Sullivan- Trainor, M., (1988). Not just another end-user liaison. Computerworld 22 (12),95-97.

AUTHOR BIOGRAPHIES

Dr. Harold Pardue is an associate professor of Computer and Information Sciences in the School of Computer and Information Sciences at the University of South Alabama. He received his Ph.D. in MIS from the Florida State University in 1996. His current research interests include Industry-level IT commercialization, Electronic Commerce, CIS curriculum development. His articles have been published in the Journal of Computer Information Systems, the Journal of Psychological Type, System Dynamics Review, Engineering Economist, and Arrowhead Journal of Business.

Dr. Michael V. Doran is a professor of Computer and Information Sciences in the School of Computer and Information Sciences at the University of South Alabama. He serves as the coordinator of the Computer Science specialization. He received his Ph.D. in Computer Science from Tulane University in 1989. His current research interests include CIS curriculum development, software engineering, real-time systems, AI and robotics. His articles have been published in the Journal of Computer Information Systems, the Journal of Psychological Type, and the Journal of Information Systems Education.

Dr. Herbert E. Longenecker, Jr. is a professor of Computer and Information Sciences in the School of Computer and Information Sciences at the University of South Alabama. He received his BS at Tulane University, and PhD from Rockefeller University. His research interests have been centered on issues relating to information systems curriculum development. He has served as co-chair for the development of IS'90, IS95 and IS'97 Model Curricula for Undergraduate Programs in Information Systems. He is co-chair of IS'93 Model Curriculum for Two Year Programs of Information Systems. He has presented research results at AIS, ACM, DSI, IACIS, ICIS, IRMA, ISECON, and SAIS meetings. His research has been published in MISQ, Database, nsE, and in society proceedings.

