Systems Architect and Systems Analyst: Are These Comparable Roles?

Jack Downey
University of Limerick
Castletroy
Co. Limerick, Ireland
+353 61 213072
jack.downey@ul.ie

ABSTRACT
The aim of this paper is to define the role of the 'systems architect' in the Irish telecommunications software sector and to compare it with Misic's definition of a 'systems analyst' in the information systems arena. The architect definition is based on in-depth interviews with practicing architects. The interview instrument is informed by social cognitive theory and the interviews were analyzed using the method of triples. The conclusion is that there are noticeable similarities between the roles, suggesting that the Irish telecommunications software sector can benefit from the computer personnel research carried out in the information systems field.

Categories and Subject Descriptors
K.7.1. [**Occupations**]:

General Terms: Human Factors, Theory.

Keywords: Irish Telecommunications Software, Systems Architect, Systems Analyst, Skills, Social Cognitive Theory, Method of Triples.

1. INTRODUCTION
This paper has two goals: The first is to define the 'systems architect' role as it pertains to the Irish telecommunications software sector and the second is to compare that with Mark Misic's definition of a systems analyst [1].

1.1 What is a 'Systems Analyst'?
For the purposes of this paper, a systems analyst will be described using the definition developed and validated by Misic [1:p 35]:

> "A systems analyst is a problem-solving specialist who works with users and management to gather and analyze information on current and/or future computer-based systems. With this information, the systems analyst, working with other MIS personnel, defines the requirements which are used to modify an existing system, or to develop a new system. The systems analyst identifies and evaluates alternative solutions, makes formal presentations, and assists in directing the coding, testing, training, conversion, and maintenance of the proposed system."

1.2 What is a 'Systems Architect'?
It would be reasonable to assume that someone called a 'systems architect' is responsible for the overall architecture (both hardware and software) of a computer system. In any given sector, an in-depth knowledge of that problem domain would also be expected, reflecting the view that systems architecture is "a result of technical, business and social influences" [2].

The same authors give the activities involved in systems architecture as: creating a business case, understanding the requirements and creating the architecture. That architecture must be communicated to the stakeholders and evaluated before a system is implemented. During construction of the system, the systems architect must ensure that the implementation conforms to the architecture.

The Guide to the Software Engineering Body of Knowledge [3] does not have a knowledge area devoted to architecture. Instead, it places "architectural structures and viewpoints, architectural styles, design patterns, and, finally, families of programs and frameworks" into the Software Design knowledge area. This suggests that Software Design should be the system architects' focus. However, based on the activities listed by Bass et al [2] above, we would also expect them to be concerned with Software Requirements and to have some part in directing the Software Construction effort.

From these descriptions, the role of the systems architect seems similar to that of a building architect. Indeed, both the Massachusetts Institute of Technology and Carnegie Mellon University have taught design using the studio techniques of building architects [4]. If this model is correct, systems architects should spend a great deal of time with customers, trying to understand their concerns. As it was put in [5]: "The vast majority of the artifacts we design are created for particular groups of users. Designers must understand something of the nature of these users and their needs." Having identified the requirements, the architect should construct the overall architecture of the product, addressing all the non-functional requirements, such as performance, maintainability, scalability, security and reliability.

Companies in the telecommunications sector provide their programming staff with both managerial and technical career ladders. Senior software developers can choose between advancing via a management career (the first step being a team leader role), or the more technical focus of the systems architect. Note that, in some North American companies, the titles: 'member of technical staff' or 'staff engineer' are used instead of systems architect.

The next section will present the methods used to gather and analyze information pertaining to systems architects. This is followed by a definition of the systems architect role, based on the interview data. Once this role is defined, it is compared with the Misic definition of the systems analyst. The paper ends with the implications for research that this comparison suggests.

2. Methodology

A total of thirty-one software practitioners from the Irish telecommunications domain have been interviewed as part of a larger field study. Five of the interviewees were systems architects and these interviews provide the basis for the analysis presented here.

2.1 The Interview Instrument

The interview instrument draws on Albert Bandura's social cognitive theory. This states that a person's behavior, personality and environmental influence each other reciprocally. He describes this dynamic as "triadic reciprocal determinism" [6]. We must ensure that the interview instrument explores the interviewees' personalities, behaviors and their work environment.

Each of the six factors shown in figure 1 is described in the following sections. For a more detailed description of the interview instrument, see Downey [7].

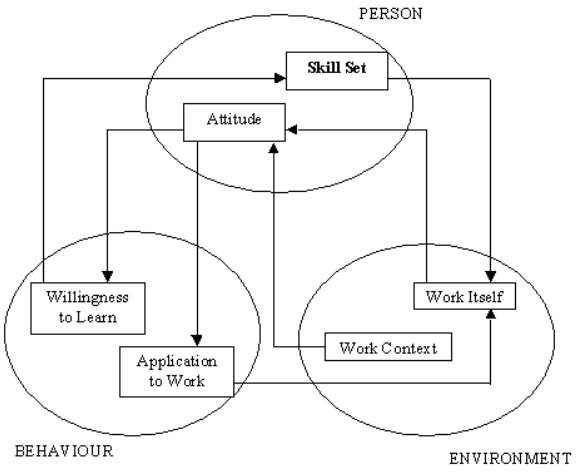

Figure 1 Social Cognitive Framework [7]

2.1.1 Environmental Factors – Work Itself

It is important that the interview determines what the person actually does. This is shown in figure 1 as the 'Work Itself'. To elicit this information, the interviewee is asked the general question: What do you do? To structure the answer, a checklist is used that itemizes the main phases of the software development lifecycle. This draws on various IEEE standards and recommendations, specifically those for purchasing [8], documentation [9], project management [10] and the Software Engineering Body of Knowledge [3].

2.1.2 Environmental Factors – Work Context

There seems to be general agreement in the literature that work itself is the key motivator for people [11-14]. However, the work context provides potential for de-motivation. Aspects like pay, conditions and supervision are what Herzberg calls 'hygiene factors'. They do not motivate in themselves, but cause grave dissatisfaction if they are not adequate.

Although not explicitly elicited, the description of the job reveals details of the work environment – the Work Context. In particular, the interviewer learns about the software development process and the interaction between different roles. To a certain extent, insight also can be gained into the interviewee's motivation.

2.1.3 Personal Factors – Skill Set

The principal personal factor considered in this study is the person's skill set. This is the product of education, training and both work and life experience. Skill set information is obtained by asking: What skills do you use in your role? Drawing on the skills surveys carried out in the information technology sector [15-19], a checklist of skills was prepared covering the areas of technical, business and interpersonal skills.

2.1.4 Personal Factors – Attitude

Attitude is the other personal attribute considered. This is a difficult area to query explicitly because, as Hogg and Vaughan [20] explain: "you cannot see, touch or physically examine an attitude; it is a hypothetical construct". However, two points raised by Allport [21] offer an approach:

- "An attitude always has an object of reference. One has an attitude *toward* partnerships, communism, or artic exploration."
- "Attitudes are usually pro or con, favourable or unfavourable, well disposed or ill disposed; they lead one to approach or withdraw from their object."

Therefore it may be possible to learn of a person's attitude by examining their behaviors.

2.1.5 Behavioral Factors – Willingness to Learn

Since we are interested in skill sets, we look at learning behaviors – the interviewees' Willingness to Learn. This factor is assessed by the biographical part of the instrument where they are asked: How did you acquire these skills? They are invited to talk about their early career motivation (from their school days), their formal third-level education and their experiences of mentoring. The interview also examines their lifelong learning attitudes by probing their views on professional training courses, night classes and trade literature.

2.1.6 Behavioral Factors – Application to Work

The other behavioral attribute – Application to Work – is not explicitly canvassed. However, it can be deduced from the level of enthusiasm observed during the interview, particularly in the job description section. Indeed, a problem in several interviews was the need to interrupt people who were prepared to talk indefinitely about their work!

2.2 Analyzing the Data

Miles and Huberman [22] describe the first stage of qualitative analysis as 'data reduction', where the large volume of data is simplified and focused. Thus each interview was studied in detail and analyzed sentence by sentence. Short, descriptive terms called codes were created and related sections of the interviews were filed under these codes. This basic level of analysis is called 'open coding' [23]. The interview data were entered into the QSR Nvivo software package. This kept track of the assigned codes and allowed all interview data associated with a given code to be viewed.

As all thirty-one interviews were analyzed, new codes were created as new topics were encountered. Eventually saturation was reached and no further codes were needed. In total, sixty-four codes were required. For this study, the interviews with system architects were extracted from those of the product managers, project managers, programmers, testers, technical writers and customer support personnel. Then each code was studied across the five relevant interviews. All five architects agreed in only six of the sixty-four codes as follows:

1. [Degree] All have bachelor's degrees in engineering.
2. [Budgeting] None of them manages a budget.
3. [Interviewing] All have interviewing experience, which can be attributed to their having had some type of leadership experience.
4. [Presentation Skills] All have to give presentations and have received training in this.
5. [Requirements Review] All review the preliminary, or marketing, requirements documents from the product management / marketing function.
6. [Courses] All have reservations about the benefits of short professional training courses.

In contrast, the interviewees who are involved in design employ different design methodologies. These architects use the Unified Modeling Language (UML), object-oriented design, structured design and prototyping tools.

From these codes, we get a very limited picture of the systems architect. They are all experienced people, whose careers are based on an engineering foundation. They all take part in the requirements definition phase. Since they do not manage budgets, it is unlikely that they carry out a formal management function.

In order to build up a better picture of what a systems architect is, a further level of coding was needed. It was decided to examine the remaining codes using the method of triples. This works by analyzing a set of elements (for instance, contributions by the interviewees filed under the same code), three at a time, and identifying the odd one out. Having identified the odd one out, we need to know, not only why this is unique, but why the others are similar. This form of analysis is also used in the repertory grid technique [24].

Each of the sixty-four open codes was examined, taking the contributions of three interviewees at a time. Based on their thoughts, thirty-five factors were identified: thirteen relating to the person's personal background and learning behaviors; twenty-two relating to the company, group or project situation.

These findings are presented in the next section.

3. Defining the System Architect Role

The tables presented in this section illustrate the findings of the method of triples. Each contains four columns. The first is a reference number; the second is the open code; while the remaining columns contain the contrasting views that the interviewed systems architects expressed under that code. In some cases, a single juxtaposition is presented; while in others, one situation is contrasted with a number of opposing viewpoints.

3.1 Personal Background and Learning Behaviors

Table 1 lists the factors that relate to the interviewees' personal background and their learning behaviors. We can see that the interviewees were good at school, favoring mathematical subjects. Two now hold master's degrees; with one having completed the coursework, but not the project. Three have studied for degrees at night. An engineering background is cited in some cases as the basis for problem-solving skills.

Those interviewees who enjoyed good mentoring experiences, such as graduate recruitment programs, endorse the practice. However, the one person who had a poor experience signals the need for carefully selected mentors.

> "The mentor I was assigned … was a troubled individual. I used go into the lab. I would be sitting down while he was testing stuff – he didn't want me there at all. I was terrified!"

Although some interviewees no longer keep up with the trade journals, they remain in tune with industry developments – citing the Internet as a better source of information. Interestingly, those who feel comfortable with technical writing duties were the ones who received writing training in college.

Their work seems to be the main facet of their lives. Although most play some sort of informal team sports, only one has a significant outside interest.

For most of the interviewees, team leading is a role they do not want to take on – citing a preference for technical work and a reluctance to get involved in personnel issues. The others either have taken technical leads or project management roles; they also were the only ones interested in personal development courses.

Table 1. Personal and Behavioral Factors

	Open Code	One Position	Contrasting Position(s)
1	Career Motivation	Good at math and science	Inspired by school visit from real engineer
			Engineering had highest entry requirements
2	Books and Journals	No longer keeps up with trade journals	Keeps up with trade journals
3	Degree	Completed Master's degree	Completed course-work only
			Bachelor's degree
4		Degree course(s) pursued full-time	Worked on Master's degree part-time
			Enrolled in Open University
			Obtained a degree by night
5	Mentoring	Was in a graduate recruitment program	Received training but no mentoring
			Poor, or no, mentoring
6		Good mentoring experiences	Poor mentoring experience
			No mentoring
			Poor, or no, mentoring
7	Problem Analysis	Cited engineering background for problem solving skills	Explained problem solving methodology
			Problem solving skills gained by experience
8	Team Sports	Informal team sports	No team sports
9	Technical Writing	Technical writing ability	Writes slowly
10		No training in technical writing	Some training in college
11	Management	Former team lead (opted for technical role)	Provides technical leadership
			Manages projects
			Takes on leadership roles
12	Personal Development	Not interested in personal development	Pursues personal development courses
13	Outside Interests	No significant outside interests	First aid work with Civil Defense

Based on the personal and behavioral factors in table 1 a picture emerges of an intelligent person who is very dedicated to the profession. This is someone who enjoys technical work above all and is not interested in politics and people interaction. As pointed out in [25], the latter are not natural skills for people with technical backgrounds. One of the interviewees notes that management tasks take up time better spent with technical work:

> "Coordinating and planning is [part of] a project management role. They are roles, or tasks, I tend to shy away from, partly because they're hard to do right and also because, if you do that, you won't be able to do any architecting. Your technical role is very much diluted once you get caught up in that whole area."

Having explored these factors, a profile of the type of person who would be attracted to the role of software architect emerges. However, they do not help us define the role. For this we need to investigate the situational factors – those relating to the company, group and project.

3.2 Situational Factors

The remaining twenty-two factors that relate to the architects' roles are given in table 2. They have been ordered to match the development life-cycle. From these we can see that the systems architect is involved from the moment the product management function comes up with requirements as follows:

Table 2. Situational Factors

	Open Code	One Position	Contrasting Position(s)
1	Customer Contact	Deals with customers through product management	Direct customer meetings
	Product Management	Little involvement with product management	Extensive involvement with product management
	Coping with Ambiguity	Reduces ambiguity by seeking answers to questions	Review process addresses ambiguity
			Whole role is defined as reducing ambiguity
			Draws on experience and asks for advice
2	Feasibility	Determines feasibility	Provides estimates for feasibility studies
			Carried out by another group
		Estimates based on work breakdown structure and experience	Estimates supported by a methodology and historical data
		Develops prototypes	No prototyping
	Costing Costing	in terms of time estimates only	Awareness of development costs versus unit costs
3	Motivation	Motivates by enthusiasm	Not keen on directing or motivating
	Risk Management	Risks identified but not managed	Risks identified and mitigation plans put in place
			Risks not identified
			Risks managed
	Corporate Culture	Problems with groups in other parts of the company	No such problems expressed
4 Func	tional Specification	Writes functional specifications	Does not create functional specifications
5 Sy	stem Architecture	Hardware issues form part of the work	Purely software systems
6 C	oding	Develops code	Reads code extensively
		Codes in C	Codes in Java
7 T	esting	Involved with testing	No testing
8	Customer Support	Provides customer assistance after delivery	None besides bug-fixes
			No involvement after delivery
9 P	urchasing	Deals with suppliers	No supplier interaction
		Deals with suppliers (at a technical level only)	Deals with suppliers (at a commercial level)
10	Presentation Skills	No involvement with academia	Extensive involvement
11	Teamwork	Floating teams	Part of fixed team
12 T	raining and Documentation	Gives training	No training courses given
13	Promotion	Promotion earned on technical merit	Promotion associated with visibility and successful projects

1. Several of the interviewees express the view that the systems architect role is defined as the interface between product management and development. Their role is to ensure that the requirements are possible to satisfy, detailed enough to ensure correct implementation and also that conflicting requirements do not pull the product in opposing directions.

   "Normally the product managers are on the different sites and they have lists of

> requirements, which are normally pretty rough – they mightn't be in great English and you have to decipher what they require and send back a list of questions to tease out what they want … Once the detailed set of requirements are drawn up, then we go through it with the customers directly."

2. The amount of effort that goes into feasibility studies is very company-specific. In most of the companies studied, the architects simply provide schedule estimates. These are based on personal experience for the most part but methodologies, such as function-point or wide-band Delphi analysis, are used; supported by historical data. One interviewee mentions the unit cost of the finished product.

> "On this product, for example, we'd make a lot of cross-discipline decisions. So, for example, some silicon feature that might make software cheaper would be traded off against the extra software cost versus maybe the product cost that the silicon would cost. So we make a lot of cross-domain decisions."

   The general consensus is that personal experience is adequate for developments similar to those done previously. However, the best way to generate estimates for unprecedented work is to develop a prototype.

3. Architects do carry out the project management of developments and, as such, will have to complete risk management studies. Otherwise they merely identify potential risks and feed them into the assigned project manager.

4. As a general rule, the architects write functional specifications if the marketing people are happy that the project is feasible.

> "It's a document which would analyze the requirements and come up with an attempt at a system-architecture. In other words, what are all the parts – separate computing entities – that are required to achieve this feature or service and also to come up with an estimate for how much effort is involved in implementing each of those parts."

   They also prefer to remain involved with the project beyond the feasibility phase.

5. As we saw earlier, the architects are involved in design, using different methodologies. All the architects work for companies where new features are being developed for mature products. This probably explains why architecture is not a significant part of their work. In one case, the only architectural decisions made are related to the assignment of software processes to computer platforms in a multi-host system.

> "If you have new services and functionality to introduce, you put them on different boxes. So the architecture wouldn't really change. Architecture is how the boxes are organized – what's running on the different boxes. How you manage whether they're highly available, or whether they're fault tolerant or whatever. … For software architecture, you'd identify where different processes would sit."

6. Many architects are expected to produce deliverable code. Assembly language, C, C++ and Java are all used, with PERL scripting being employed in some prototypes.

7. Similarly, these architects take part in the testing process. Although Myers [26] warns that testing is not well understood in the software development world, the interviewees all display an appreciation of the theory behind testing.

> "I took courses on it many years ago … This was an area I was interested in … Unit testing is something we're getting stronger at and also integration and system testing. So we developed our own testing tools and I had a big input into the testing tool development."

8. Architects are also involved with the product after delivery. For instance, in one company, existing databases have to be reformatted to work with the new features and in another, the architect provides consultancy services to customers who wish to interface with third-party products or improve their performance. Training courses have been given by the architects.

> "Sometimes it's to help them tune their applications – we went through a phase of that on [the last product] where customers were trying to get certain performance out of the product and their software ... In other cases, customers might have been using products from third parties ... and might have found it difficult to get it working on ours. We would have helped them out with that too."

9. The architects have limited involvement with suppliers. Rarely do they have to consider commercial aspects, being involved instead with the specification of third-party software.

> "It would be more a case of reviewing statements of work and deciding what they need to do in the first place. The actual money negotiation – there are professionals who do that … Also, reviewing their test plans and reviewing the results of their test plans."

Besides enjoying the hands-on technical aspects of the job, another reason for the architects' interest in working through the development cycle could be the comfort of being part of a team. The architects with least involvement in the construction phase found themselves floating between teams that were formed to carry out specific tasks (such as determining the feasibility of a feature).

Summarizing these findings into a role definition gives:

The systems architect is a technical expert with experience of the company's products and the industry in general. S/he works with product management to refine customer requests into a set of

engineering requirements that are possible to implement. S/he will identify possible implementations that satisfy the requirements and assess their feasibility in terms of time schedules and headcount. Architects sometimes remain with a project through construction, writing the functional specification and contributing to the design, code and test activities as a technical lead, an advisor or in a hands-on development capacity. After product delivery, architects may provide additional consultancy as well as training to end-users.

4. Comparing System Architects with System Analysts

While a picture has emerged of the systems architect role, it differs significantly from the expectations presented in section 1.2. The principal concerns of the architect seem to centre on Software Requirements with a major involvement in Software Construction. However, this contribution is often a hands-on as well as a guiding role.

Indeed, the role looks very similar to Misic's [1] definition given in section 1.1. Factoring out the responsibilities offered in that definition, we can see several parallels between Misic's systems analyst and the architects of this study:

1. "Is a problem-solving specialist". Problem solving or problem analysis is accepted by the interviewees as a key skill and several provide examples of the methodology they use. Part of their problem solving difficulties relate to having to make decisions based on incomplete information. They have all learned how to cope with such ambiguity – one of the interviewees going as far as stating that the role of the architect is about reducing ambiguity.
2. "Works with users and management". Surprisingly, few of the architects meet the customers directly. However, they do work with them via the product management function. Being aware of the customers' perspectives influences the approaches to problems. For instance, one architect endorses the use of function-point analysis because the estimates can be presented to the customer in terms of functionality rather than in terms of life-cycle phases. Now the customer knows how much each feature is expected to cost in terms of time.
3. "Gathers and analyses information about computer-based systems". For one of the architects, performance modeling is an important part of his feasibility work. For the others, knowledge of the latest telecommunications protocols is essential in order to inter-work with other node s in the telecommunications network. Getting the product to inter-work with third-party software is another example.
4. "Works with other MIS personnel". To understand the customer requirements, the architect must work with the product management function; to determine the initial estimates, s/he must interact with the programming teams. Obviously, they contribute to the project management effort by identifying risks, generating estimates and, in some cases, work breakdown structures. Finally, the y work with the customer support group to help deal with upgrading, performance and inter-working issues.
5. "Defines requirements". Requirements coming from the product management function often cannot be immediately implemented. The architect must define a more detailed set of requirements from the se, a set that is testable and measurable.
6. "Identifies and evaluates alternative solutions". The interviewees discuss alternative solutions and the importance of recording these is stressed. This is useful if the product is elaborated in the future.
7. "Makes formal presentations". This is also part of the systems architect role. All the interviewees have received training in giving presentations, although some describe themselves as nervous presenters.
8. "Assists in directing the coding, testing, training, conversion, and maintenance". As we've seen with our architects, this might be better expressed as: 'assists and directs'. Most of the architects stay with the project to delivery and beyond. Some are more involved in the hands-on technical work, but others provide a technical lead (direction) by taking part in the reviewing and inspection processes.

5. Conclusion

This paper has shown that the role of a systems architect in the Irish telecommunications software sector is not dissimilar to that of a systems analyst in the information systems (IS) arena. It should be noted that the architects studied here work in Irish subsidiaries of multi-national companies. As pointed out by McGovern [27], Irish subsidiaries do not offer the full spectrum of research and development work. This suggests that the systems architects working in head office may not conform to this definition.

The systems analyst role is often a position on the way to management. As noted by Lee [28], "a programmer/analyst progresses through his/her career to systems analyst and to IT manager". In contrast, the systems architect role is designed to offer progression without management responsibility, reflecting its place on the technical career ladder.

The parallels between the roles means that the Irish telecommunications sector can benefit from the extensive research carried out in the computer personnel research discourse. For instance, the lessons learned in what makes a good systems analyst [29-31] could be applied to the assessment of systems architects. Also insights into training needs [32], personality [33] and, to a lesser extent, career paths [28] can benefit the understanding of architects.

One of the questions posed in the literature is: who is the IT workforce? Kaarst-Brown and Guzman [34] note that one of the problems with answering that question relates to "outdated or exclusionary definitions" of IT workers. This paper makes a small contribution to addressing this problem, by providing a definition for a systems architect. Similar analysis on the other interviews carried out in the overall study should yield definitions for product managers, project managers, programmers, testers, technical writers and customer support personnel. The approach taken may also be applicable to "IT professionals, computer scientists, software developers, and business professionals trained in MIS, as well as various occupational sub-categories in

organizations including programmer, analyst, network specialist, and project manager, to name only a few" [34].

6. ACKNOWLEDGMENTS

This research has been supported by the Science Foundation Ireland Investigator Programme, B4-STEP (Building a Bi-directional Bridge Between Software ThEory & Practice).

7. REFERENCES

[1] M. M. Misic, *Journal of Systems Management*, **47**, 34-40 (1996).

[2] L. Bass, P. Clements and R. Kazman, "Software Architecture in Practice," Addison-Wesley, ed. 2nd, 2003.

[3] IEEE Computer Society, "Guide to the Software Engineering Body Of Knowledge," IEEE Computer Society, Los Alamitos, California, Los Alamitos, California, ed. 2004 Version, 2004.

[4] S. Kuhn, *IEEE Software*, **15**, 65-71 (1998).

[5] B. Lawson, "How Designers Think," The Architectural Press, London, 1980.

[6] A. Bandura, "Social Foundations of Thought & Action: A Social Cognitive Theory," Prentice Hall, Englewood Cliffs, New Jersey, 1986.

[7] J. Downey, A Framework to Elicit the Skills Needed for Software Development, J. E. Moore and S. E. Yager, Eds., 2005 ACM SIGMIS CPR Conference, Atlanta, Georgia, ACM Press, 2005, p. 122-127.

[8] IEEE Computer Society, "IEEE Recommended Practice for Software Acquisition," IEEE Inc, New York, 1998.

[9] IEEE Computer Society, "IEEE Standard for Software User Documentation," IEEE Inc, New York, 2001.

[10] IEEE Computer Society, "IEEE Guide Adoption of PMI Standard: A Guide to the Project Management Body of Knowledge," IEEE Inc, New York, 2003.

[11] F. Herzberg, B. Mausner and B. B. Snyderman, "The Motivation To Work," John Wiley & Sons Inc., ed. 2nd, 1959.

[12] J. R. Hackman and G. R. Oldham, "Work Redesign," Adison-Wesley, Reading Massachusetts, 1980.

[13] G. M. Weinberg, "The Psychology of Computer Programming, Silver Anniversary Edition," Dorset House, New York, ed. 2nd, 1998.

[14] J. B. Thatcher, Y. Liu and L. P. Stepina, The Role of the Work Itself: An Empirical Examination of Intrinsic Motivation's Influence on IT Workers' Attitudes and Intentions, SIGCPR '02, Kristiansand, Norway, ACM, 2002, p. 25-33.

[15] E. M. Trauth, D. W. Farwell and D. Lee, *MIS Quarterly*, **17**, 293-307 (1993).

[16] D. M. S. Lee, E. M. Trauth and D. Farwell, *MIS Quarterly*, **17**, 313-340 (1995).

[17] S. Sawyer, K. R. Eschenfelder, A. Diekema and C. R. McClure, *ACM SIGCPR Computer Personnel*, **19**, 27-41 (1998).

[18] N. Shi and D. Bennett, *ACM SIGCPR Computer Personnel*, **19**, 3-19 (1998).

[19] C. L. Noll and M. Wilkins, *Journal of Information Technology Education*, **1**, 143-154 (2002).

[20] M. A. Hogg and G. M. Vaughan, "Social Psychology," Prentice-Hall, Harlow, England, ed. 2nd, 1998.

[21] G. W. Allport, "Pattern and Growth in Personality," Holt, Rinehart and Winston, Cambridge, Massachusetts, 1937.

[22] M. B. Miles and A. M. Huberman, "Qualitative Data Analysis," Sage Publications, Thousand Oaks, California, ed. 2nd, 1994.

[23] A. Strauss and J. Corbin, "Basics of Qualitative Research," Sage Publications, Inc., Thousand Oaks, California, ed. 2nd, 1998.

[24] V. Stewart, "Business Application of the Repertory Grid Index," Enquire Within Developments Ltd, 1997.

[25] T. DeMarco, "Slack - Getting Past Burnout, Busywork and the Myth of Total Efficiency," Dorset House, 2001.

[26] G. J. Myers, "The Art of Software Testing," John Wiley & Sons, Inc., Hoboken, New Jersey, ed. 2nd, 2004.

[27] P. McGovern, "HRM, Technical Workers and the Multinational Corporation," Routledge, 1998.

[28] C. K. Lee, Transferability of Skills over the IT Career Path, J. E. Moore and S. E. Yager, Eds., 2005 ACM SIGMIS CPR Conference, Atlanta, Georgia, ACM Press, 2005, p. 85-93.

[29] N. P. Vitalari and G. W. Dickson, *Communications of the ACM*, **26**, 948-956 (1983).

[30] K. D. Schenk, N. P. Vitalari and K. S. Davis, *Journal of Management Information Systems*, **15**, 9-50 (1998).

[31] J. L. Wynekoop and D. B. Walz, *Information Technology and People*, **13**, 186-195 (2000).

[32] M. M. Misic and N. L. Russo, *The Journal of Systems and Software*, **50**, 65-73 (2000).

[33] J. E. Moore (1991) Personality Characteristics of Information Systems Professionals. Athens, Georgia, ACM Press, pp. 140-155.

[34] M. L. Kaarst-Brown and I. R. Guzman, Who is "the IT Workforce"?: Challenges Facing Policy Makers, Educators, Management, and Research, J. E. Moore and S. E. Yager, Eds., 2005 ACM SIGMIS CPR Conference, Atlanta, Georgia, ACM Press, 2005, p. 1-8.