



ELSEVIER

Information and Software Technology 45 (2003) 281–287

**INFORMATION  
AND  
SOFTWARE  
TECHNOLOGY**

[www.elsevier.com/locate/infosoft](http://www.elsevier.com/locate/infosoft)

Short Communication

# A viewpoint on software engineering and information systems: integrating the disciplines

Neil McBride

*Centre for IT Service Management, De Montfort University, The Gateway, Leicester LE1 9BH, UK*

Received 11 September 2002; revised 4 December 2002; accepted 9 December 2002

The design and implementation of effective processes and technology for the delivery of computer-based information systems (IS) in organisations remains a difficult academic and professional problem. It is difficult to marry the rigorous analysis associated with an engineering-based discipline with the softer people-focused discourse required in developing ISs which essentially model social processes within organisations. Large gaps exist between the social and the technical both in the management and development of ISs [1,2]. Furthermore, the divergence of skills and cultural attitudes between software engineers and IS professionals has not helped in the delivery of robust and flexible systems for organisations [3].

The exploration of other industries and disciplines to find metaphors, models and analogies is of significant value in seeking to bridge the managerial and technical gaps faced by ISs and software engineering professionals. However, such analogies need to be evaluated for their suitability in information system development (ISD). Avison and Wilson [3] offer the construction industry as a model for ISs development 'to provide guidance which might lead to a more positive spirit of co-operation between software engineers and IS analysts.' Using the building of a custom-designed house as a model, they identify a series of separate roles within an essentially linear ISD process. These roles, based on specific skill-sets, communicate at specific points in system development as the life cycle progresses through requirements definition, to design and to build. Avison and Wilson [3] suggest that the implication of such an analogy is that there should be separate disciplinary groups, separate curricula and separate professional associations.

In this paper, I examine the limitations of the construction industry analogy and suggest that it is inadequate as a model for ISD in a dynamic organisational environment. The value of separating software engineering and ISs

development academically and professionally is questioned. I examine some lessons which can be drawn from the alternative analogies of film production and jazz improvisation. For each analogy, the structure of ISs, the process of development and the professional roles involved in development are discussed. I conclude that far from encouraging co-operation, separate curricula may strengthen existing disciplinary barriers in an unfruitful way. I suggest that ISD in evolving organisational environments requires a variety of specialists who share a common foundational knowledge of both social and technical environments.

## 1. The building analogy

Avison and Wilson [3] discuss the building of a one-off architect-designed house in which distinctive roles are assigned to the participants. The buyer discusses his/her requirements with an architect. The architect's plans are developed into a detailed design specification by a construction manager. The builder constructs the house using the detailed specification and the buyer inspects the finished product. Parallels are drawn between roles in house construction and roles in IS construction. The user is the house buyer; the architect is the systems analyst; the construction manager is the software engineer and the builder is the programmer. A number of implications are drawn from this analogy. The separate roles require separate fields of knowledge and distinctive training. The roles are primarily involved in different steps. Transitions between steps are achieved through documents communicating the work of each expert to the next expert in line. Separate curricula and professional associations are required for each role. Use of the building analogy leads Avison and Wilson [3] to suggest that different skills and traits are required for different roles and that the investigation of these roles by researchers may require

*E-mail address:* [nkm@dmu.ac.uk](mailto:nkm@dmu.ac.uk) (N. McBride).

different research approaches. They conclude that such separation of disciplines and roles should not result in competition between roles, should not encourage self-protectionism, and should require awareness of other roles besides their own. However, the use of a construction industry analogy may be misleading in terms of its use as a description of IS structure, the template it offers for the IS development process, and the roles it suggests for IS professionals. The analogy describes a sequential development process, whose underlying assumption is that the environment is predictable and stable, and whose goals include achieving efficiency and reducing uncertainty [4]. Such a model does not fit well in increasingly dynamic organisation where processes and technology are rapidly changing.

### 1.1. Structure

A construction industry analogy would suggest that ISs are stable, physical artifacts which, once built, should not change significantly. Major adaptation should not be considered and may not be possible. Once a building is constructed, minor changes become expensive and major structural changes may not even be possible. Similarly, an engineered IS is considered to be stable once built. As may be the case with buildings, expecting ISs to be stable places the burden of change on the user who must adapt to the IS's 'way of doing things'. Requirements are 'fixed in concrete' and the IS cannot adapt to the changing organisational structures and processes. An IS should be considered more of a social artifact dynamically reflecting organisational processes, workflows and values rather than a physical artifact subject to physical and engineering laws. The focus on the physical structure of the IS encouraged by a construction industry analogy may limit considerations of the social context, organisational values and political and power structures which influence the IS and drive its shape and content.

### 1.2. Process

Avison and Wilson's interpretation of the construction analogy [3] views ISD as principally a linear process in which different roles are dominant at different points in time. The joining points between roles become formal milestones checked off in formal meetings. A building analogy suggests that feedback and design change is discouraged. While Avison and Wilson [3] recognise that IS development does tend to be iterative and spiral, the analogy implies that information tends to flow in one direction.

### 1.3. Roles

The construction industry analogy suggests the separation of roles, both in terms of skills and project

involvement. The setting of such clear boundaries may result in islands of knowledge and diverging cultures. Thus communication between the roles becomes more difficult and may be more formalised within the process. Furthermore, separation of roles within an IS project may inhibit participants from obtaining a holistic view of the system, its structure and its purpose within the organisation. For example, Avison and Wilson [3] suggest that programmers, analogous to builders, concentrate on the design, testing and coding of programs. I would suggest that good testing, efficient coding and integration of programs together demand that the programmer understands the program in its system and business context, is able to interpret program logic in the context of its role in a business process and comment back to analysts where the business process does not make sense.

## 2. The film production analogy

Avison and Wilson [3] identify the film analogy as an alternative analogy. Considering how a film is made provides some useful suggestions concerning the flexibility of roles, the dynamic nature of the process, the presence of a high degree of creativity throughout the process and within each role, and the dynamic development of the content of the resulting product.

Film production first involves the development of a script. Whether the script is based on an original idea, or a literary source, it is not developed in a vacuum, but draws on other art and literary sources, for example a novel [5]. Once developed, the script forms the foundations of the film, but is still revised even during shooting and is interpreted by a number of creative artists including directors, actors and cameramen. Pre-production involves casting, location hunting, planning, budgets, music, choreography, research and scheduling. Script re-writing is also done. Design of sets, buying of props and design of costumes occurs in pre-production. During filming, discoveries about the script, problems with resources, weather and technology may alter scripting and shots. Post-production involves editing, dubbing, adding music. As the film is then distributed, marketing and press publicity becomes an important part of the process.

### 2.1. Structure

The film analogy has some advantages over the building analogy in that it suggests a more pliable product structure emerging from the production process. It also acknowledges that, usually, the film is based on some pre-existing ideas or artistic source. The film builds on a body of existing work. Information systems development also generally builds on previous work, whether it is in the form of legacy systems, pre-existing processes or pre-existing technical constraints. However, while acknowledging the way in which an IS

develops dynamically during the development process, the film analogy does not acknowledge the continuing development of the product once it has been released, nor the active role of the user in IS development. An IS is more of a dynamic, evolving artifact than a film is.

## 2.2. Process

The film analogy puts more emphasis on the dynamic development of the product. Whereas the structure of a building is fixed following design by the architect, film scripts change as the film producers interact, ideas are added by the other participants and changes may be made right up to the release of the film in cinemas. The script provides a foundation for the film, but does not dictate how the final film looks. There is still a linear process, in which different activities take place pre-production, production and post-production. The film production analogy identifies a greater sense of team work and diverse creative input than a construction analogy. It recognises both the continuous involvement of many creative staff throughout the production process and the contracting in of specialist staff for short times to carry out specific tasks. Planning is still important: locations must be prepared, actors scheduled, catering booked and shooting schedules defined. However, the detailed process is more open to day-to-day changes.

## 2.3. Roles

Film production involves a large number of different creative roles working together. Directors, actors, composers, cameramen and editors all provide their creative input to the final product. The resulting film is an expression of many interpretations, negotiated and combined through a process of consensus building which results from both negotiations within groups and dictation by the powerful. The range of roles requires a wide skill-set. Similarly, IS development requires a wide variety of roles beyond those suggested by the construction analogy. User input is of great significance. Indeed, the user might lead the development process [6]. Internet-based applications may require involvement of graphic designers, marketing staff and legal staff. A greater variety of professionals will work together on an internet-based IS. Old skills of programming, analysis and database design are still needed, but the wider integration of skills requires improved and more communicative project management [7]

## 3. Jazz improvisation

While film production more closely reflects IS development in highlighting continuous involvement of practitioners in the process and the dynamic, creative development of the product, it does not emphasise the uncertain nature of the IS development process in which

analysts, programmers, users and designers alter the shape of the resulting IS as a result of the learning and communication which occurs during the development process.

Information systems development involves a high degree of improvisation due to the particularly innovative nature of the product. Rigid engineering approaches may not yield sufficiently adaptive IS. Recent organisational studies have recognised the parallels between jazz improvisation and organisational and product innovation [4,8].

Jazz improvisation involves the use of minimalist musical structures including harmonies, melodies and rhythm. These basic structures are then varied by the musicians. Jazz musicians within a small team then elaborate simple structures in complex ways. Variation and experimentation are supported because the musical structures are present and the musicians operate within a set of social norms where roles change and there is intense interaction within the group.

Kamoche and Cunha [4] suggest the concept of minimal structures, a set of consensual guidelines and agreements within which improvisation can develop. These structures ‘constrain the turbulence of the jazz process by specifying particular ways of inventing and co-ordinating musical ideas’. Kamoche and Cunha [4] differentiate between social minimal structures and technical minimal structures. In jazz improvisation, social minimal structures include behavioural norms, communication codes, leadership and soloing and cultural attitudes supportive of risk-taking. Technical structures within jazz improvisation include key definition, chord progressions, repertoire and song templates, instrumental knowledge and skills in refashioning performances in response to colleagues and audiences. Using concepts from jazz improvisation, a model for new product development can be devised in which minimal social and technical structures can be identified. Table 1 uses Kamoche and Cunha’s improvisational model as a basis for a minimal structure model of ISD. For each social and technical dimension of jazz improvisation identified by Kamoche and Cunha, the IS development equivalents are defined.

A jazz model may provide a basis for understanding extreme programming [9] in which values such as communication, simplicity, feedback and courage in decision-making, and principles such as rapid feedback and incremental change form the minimal social structures; and practices such as the use of coding standards, patterns and metaphors constitute the minimal technical structure.

### 3.1. Structure

A jazz improvisation analogy highlights the dynamic structure of ISs which change rapidly in response to organisational changes and environmental stimuli. The IS will be based on minimal structure that rapidly adapts to developing ideas and user and process changes. New

Table 1  
An improvisational model of information systems development

Social structure		Technical structure	
Jazz improvisation	ISD	Jazz improvisation	ISD
Behavioural norms	Job divisions. Devolved responsibilities. Expected role of customers. Meeting structures and work patterns	Keys, chord progressions, repertoire	Quality and process standards. Definition of accepted development routines and methods
Communicative codes	Socialisation patterns and work expectations. Email usage patterns. Cross-project communication. Social networking and team communication. Shared languages and shared assumptions about business processes and culture	Templates of songs, choruses, etc. on which to improvise	Program structure templates. Component libraries. Set of minimal system structures. Product vision
Partnering in autonomous ensemble	Information sharing. Apprenticeships. Exchange of expertise	Wide stock of talent and performative competence	Pool of varied programming, analysis and business expertise. Cross-training an technical and analytical skills
Trust within wide zones of manoeuvre and constructive controversy	Open discussion and questioning environment. Discussion and debating forums	Knowledge of music technology and instrumentation	Knowledge of available IT technology and how to integrate various technology to achieve the required information system
Alternate soloing and comping. For leadership and personal development	Flat management. Small teams. Revolving leadership of information systems teams	Experimenting with new instruments, styles and textures of sound	Researching IS tools and methods. Bricolage. Constant review of business strategy and ideas to identify new IS needs and technology opportunities
Risk-taking and continuous learning	Support for new ideas and new approaches to IS development. Support for continuous learning both in-house and external. Supportive culture of risk-taking	Refashioning performance in response to colleagues and audience	Structures of information flows from users. User involvement in design, development and testing. Iterative design and re-modelling of IS in response to users

programs may be developed, or existing components reassembled to meet changing requirements. Furthermore, software patterns may be seen as the equivalent of minimal structures. A jazz improvisation analogy suggests that while core structures may be defined and kept relatively stable, considerable variation and creative adaptation is built on these minimal structures.

### 3.2. Process

The use of a jazz improvisation analogy emphasises the dynamic nature of IS project development. While planning is important, and minimal management structures and procedures should be adhered to, considerable flexibility is possible in the development process. Iterative development is the norm. Product is continuously delivered to the customer or audience. The process permits a high level of individual risk-taking and promotes continuous learning. At various points in the development project different themes may be developed, focusing on different business or technological issues, analogous to solos in jazz. Re-working material is a central element of the software development process. This dynamic process is supported by minimal structures defining processes and values. Communication occurs throughout all activities in the project and is not limited to defined meetings.

### 3.3. Role

A variety of roles are employed within the development team. A jazz improvisation analogy highlights the rotation of roles and the development of mentor - expert partnerships. Every role engages with the project throughout its lifecycle. While team members may have specific expertise within a variety of technologies, there is no separation of roles. All participants are totally committed for the whole project; they listen to each other and contribute to the development and improvisation. All participants fully understand the social and technical context and are sensitive to the client.

## 4. Implications

Table 2 identifies some differences between Avison and Wilson's construction analogy and the film production and jazz improvisation analogies discussed in this paper. If we view ISD as a dynamic process, involving a variety of experts working together in an integrated team, that shares a common holistic view of the project's technology and business context, then there are a number of implications for the practice and teaching of software engineering and IS which are discussed below using Avison and Wilson's headings.

### 4.1. Disciplinary groups

The separation of software engineering and IS into different departments within different faculties is not conducive to shared understanding and culture. All aspects of the construction and delivery of ISs should be in one faculty. The faculty should draw on a wide variety of disciplines, from sciences such as mathematics through to social sciences such as psychology. The interdisciplinary nature of computing should be emphasised, as well as the common goals. While specialist groups covering, for example, information management, artificial intelligence and programming will exist within the faculty, communication of shared understanding and values between these groups should be paramount. This may be achieved through joint seminars, cross-teaching, and shared accommodation. Members of a computing faculty, regardless of expertise should develop a shared understanding of the social and technical context of computing.

### 4.2. Curricula

Core curricula should be developed that bridges the managerial, social and technical and is an essential part of all computing-based degrees whose primary focus is on the skills and concepts involved in the building and delivery of ISs to organisations.

This curricula might include:

- Systems modelling and analysis
- Information service management
- Internet technology and infrastructure
- Communications technology and management
- Quality standards control and management
- Programming
- Database design and development
- Business context and strategies
- Software development management
- Computational intelligence
- Systems integration
- Procurement and contracting
- Human behaviour, and human-computer interfaces

The core curricula should lead to a series of specialisms. Here the focus is on gaining expertise in specific areas to which core skills may be applied. Examples include:

- Real time and embedded systems
- Internet technologies and e-commerce system development
- Evolutionary computation and adaptive systems
- Information systems management and ethics
- Retail systems technology, design and implementation.
- Financial systems and applications.
- Knowledge management and management IS

Table 2  
Comparison between construction analogy and film production/jazz improvisation

Construction analogy	Film production/jazz improvisation analogy
Sequential progression	Iterative dynamic progression
Product structure defined and fixed at start	Product structure evolves in response to new ideas and environmental change
Roles fixed in skill base and temporal involvement with project	Roles rotate and project involvement is continuous throughout the project's lifetime
Changes difficult to achieve	Changing structure and new structure is the norm
User input limited temporally and functionally	User input critical throughout project
Technology fixed at start of project	Technology adaptation occurs as new tools and methods are explored
Focus on technology and rigorous adherence to pre-defined design	Focus on creativity and development of new ideas within a minimal design framework

- Service industry computing.
- Public sector computing.

Through such curricula, all students are given a holistic understanding of the technology and the business and social context. Theory and practice are always integrated and informing each other. The impact of the technology is not treated as separate to the design and development of the technology: the strong links between the two are recognised.

#### 4.3. Professional associations

Separate professional groups for systems engineers and systems analysts [3] may only serve to increase communication barriers in what should be an integrated population of experts working for an organisation to derive benefits from computing. Professional associations such as the British Computer Society should seek to widen their understanding of the area of organisational practice. For example, in the United States, the Association of Computing Machinery covers both the managerial and technical aspects of professional computing. Software engineering and ISs are part of an integrated discipline concerned with the delivery of organisational benefit using computer technology. Professional associations should be the guardians of standards of technology and management, defined by minimal structures.

#### 4.4. Academic associations

The emergence of separate academic groupings for IS may reflect academic insecurity and attempts to construct academic disciplinary boundaries around groups of interests that are inherently interdisciplinary. Academic associations should encourage technical and managerial exchange of knowledge.

#### 4.5. Teamwork

Teamwork should involve working together on tasks within the project. Cultural distinctions and task demarcations may be damaging to team building. Job rotation,

changing roles and leadership and mentoring may contribute to developing a team that shares the values and vision of the project. Information systems development team members need to adopt the same minimal social structures within which process improvisation may occur.

#### 4.6. Skills and traits

Avison and Wilson [3] separate software engineers and systems analysts on the basis of traits. Software engineers are portrayed as mechanists, interested in the technology and systems analysts as romantics, interested in people. While personality traits will differ, all roles should develop technical, service and people skills. Mechanists can be trained in social skills. Romantics can gain logical and technical understanding to temper the decisions they make. It is not acceptable for any IS practitioner to remain ignorant of the technology or the client. Skills in critical thinking and social interaction can be taught. IS analysts should be good at problem solving and software engineers should develop communication skills.

#### 4.7. Research approaches

Both qualitative and quantitative research approaches are appropriate within an integrated computing discipline. Quantitative measurement is equally valuable in measuring, for example, the management aspects of IS implementation and the technical aspects of software quality. Qualitative research is of value in longitudinal studies of ISD and in understanding human-computer interfaces. Computing demands integrated research approaches to answer complex research questions with both social and technical dimensions.

#### 4.8. The position of programming

Programming involves the selection and combining of minimal technical structures. Increasingly programming is a process of component assembly and involves the interpretation of existing systems and solutions. Such roles require improvisation on themes and structures derived from

business processes. Training in programming languages is necessary but insufficient for effective systems development. The programmer who behaves as a coder, isolated from the business and from the technical and managerial improvisation may not be an effective member of an ISD team.

## 5. Conclusions

This paper has examined the shortcomings of using a construction analogy for learning about IS development. Such an analogy can give a misleading impression about the stability of the IS and the interaction and separation of roles in IS development. I suggest that a film production analogy is of more value in emphasising the variety of creative roles, the involvement of these roles throughout the project, and the dynamic way in which the product develops. I have suggested that ISD is more of an improvisational process and that the application of jazz improvisation in organisational studies of new product development may be relevant to IS development. However, all these analogies are inadequate in taking product-oriented view of IS. Both construction and film production focus on the product. The client is remote and passive. Furthermore, the emphasis is on the product as an end in itself. I would suggest that IS are not ends in themselves, but tools that form part of the delivery of a service. The development of the IS should not be seen as the sole purpose of IT professionals. Rather the IS is a tool for delivering benefits to the organisation in terms of, for example, better client interaction and more efficient work-flow.

An analogy is required which puts ISs in the context of the delivery of an information service within the organisation. The building of a technical artifact does not constitute the service. For many organisations, the focus of the IT department is on the delivery of a service using procured systems or assemblies of components. Analogies from

hospitals, higher education or the leisure industry may be more appropriate for highlighting the service-oriented nature of ISD.

No analogy will perfectly match IS development and therefore provide all the lessons that may be required. Indeed, there is a danger with any analogy that it may emphasise certain characteristics of the development process at the detriment of others. A danger with the construction analogy is that it emphasises product-orientation and inflexibility. I would suggest that service-orientation and flexibility are key elements required in the process of developing ISs in an environment where systems must be fluid, product lead times are short and business change is the norm.

## References

- [1] J. Peppard, J. Ward, Mind the gap: diagnosing the relationship between the IT organisation and the rest of the business, *Journal of Strategic Information Systems* 8 (1999) 29–60.
- [2] D. Champion, F.A. Stowell, PEARL: a systems approach to demonstrating authenticity in information systems design, *Journal of Information Technology* 16 (2001) 3–12.
- [3] D. Avison, D. Wilson, A viewpoint on software engineering and information systems: what we can learn from the construction industry, *Information and Software Technology* 43 (2001) 795–799.
- [4] K. Kamoche, M. Pina e Cunha, Minimal structures: from jazz improvisation to product innovation, *Organisational Studies* 22 (5) (2001) 733–764.
- [5] S. Birtwistle, S. Conklin, *The Making of Pride and Prejudice*, Penguin, London, 1995.
- [6] F.A. Stowell, D. West, *Client-led Design. A Systemic Approach to Information System Definition*, McGraw-Hill, Maidenhead, 1994.
- [7] N.K. McBride, Public domain information systems, Proceedings of the Ninth Annual Business Information Technology Conference, Manchester Metropolitan University, November 6–7, 1999.
- [8] K.E. Weick, Improvisation as a mindset for organisational analysis, *Organisation Science* 9 (1998) 543–555.
- [9] K. Beck, *Extreme Programming Explained*, Addison Wesley, Reading, MA, 1999.