17 INCORPORATING SOCIAL TRANSFORMATION INTO THE INFORMATION SYSTEMS AND SOFTWARE DEVELOPMENT LIFECYCLES

Christopher J. Hemingway
Tom G. Gough
University of Leeds
United Kingdom

Abstract

All approaches to information systems and software development assume lifecycle models, which have a significant impact upon the perspective adopted for design and development. In recognition of the limitations of conventional software engineering lifecycles, the Information Systems (IS) community has focused upon gaining recognition of organizational and human issues in systems design. The Human-Computer Interaction (HCI) community has also tried to introduce user-centered practices into software engineering, but its success has been limited because it has augmented existing lifecycle models that are document or risk-driven. Yet, user-centered design can only be fully realized through the adoption of a user-centered model of the information systems lifecycle as the basis for developing ICT-based systems. This paper explains how several of the limitations of current IS and HCI theory and practice can be overcome by modeling development lifecycles in terms of social transformation. The paper then presents an information systems development lifecycle based upon social transformation and illustrates how a user-centered software development lifecycle can be integrated into the IS development process.

Keywords: Information systems development, systems lifecycle, software development, socio-cognitive theory of information systems, user-centered design.

1. Introduction

Owing to the overriding need to manage the complexity of software systems, conventional approaches to software engineering used technology-centered methods for design (see, for example, DeMarco 1979; Yourdon 1972). These approaches embody the "waterfall" model of the development lifecycle (Royce 1970), viewing analysis as the identification of the correct structure to encode the problem domain. Avison and Fitzgerald (1995) suggest that the software development community recognize the need for a more user-centered approach, although their development methods still leave design decisions under the control of technical experts. The empowerment of users during the development lifecycle has received attention by the Information Systems community for some years, recognizing the need for user participation in systems development and the importance of managing end-user computing. In light of progress in supporting user participation in major development efforts, Mumford (1983) proposed the following typology:

1. **Consultative participation**. Analysts discuss the system's requirements with users, but the technical experts perform design.
2. **Representative participation**. Representative users work on the design team and are, thereby, involved in decision making.
3. **Consensus participation**. Users drive the design process, making all key decisions.

An obvious trend in this list is the transfer of authority and responsibility for design decisions from technical experts to users. It has rarely been noted, however, that this transfer does not empower users unless they also have the ***capacity*** to act accordingly (see section 3 for a definition of this term). Furthermore, the education of users to improve their capacities for developing and using ICT-based information systems—necessary for the effective use of end-user computing in an organization—is poorly understood. Section 2 of this paper shows that, although existing approaches to systems development have considered authority, responsibility and capacity, their use of these concepts as the basis for guiding the development process has been limited. Furthermore, analyses of capacity are not acted upon during systems development to ensure that users have the information handling skills that the systems design implies. To address these weaknesses, this paper sets out a case for changes to information systems theory and practice through the adoption of user-centered information systems and software development lifecycles. The changes to IS theory, presented in section 3, are drawn from the socio-cognitive theory of information systems (Hemingway 1999; Hemingway and Gough 1998). Lifecycle models based upon this theory are developed in sections 4, 5 and 6 to demonstrate the effects of incorporating social transformation into information systems and software development. The implications for information systems and software engineering practice are then considered and conclusions drawn.

2. Current Perspectives on the Information Systems and Software Development Lifecycles

Software engineering differs from other engineering disciplines in two key respects:

- the physical environment is viewed in terms of peripheral constraints rather than as the basic medium for development; and
- engineers are primarily concerned with systems and pay comparatively little attention to the development of general systems components.

The consequence of these differences is that software requires a fundamentally different approach to engineering. Numerous methods have been proposed, but they all (with the exception of formal methods) typically result in software systems that have numerous design flaws. Quality management is being improved through the application of quality management techniques, such as the Capability Maturity Model (Ferguson and Sheard 1998; Herbsleb et al. 1997), and standards, such as ISO9001 and ISO15504. Nevertheless, the systemic approach to design and the absence of architectural principles for developing software systems limits the quality and reliability that can be achieved and contrasts sharply with, for example, electronic engineering, where applications are constructed from a very small number of precisely engineered modules (for example, standard amplifier circuits).

The most widely known model of the software development lifecycle is the conventional "waterfall" model (Royce 1970). Although this model is now regarded as grossly oversimplified, its influence on more recent lifecycle models is readily discerned. Two serious and well-documented deficiencies of such lifecycle models are considered here. First, the customer (who defines the system's requirements) and the users are excluded from much of the decision process. Consequently, many decisions are based upon the developer's interpretations of what the customer requires, supplemented by his or her understanding of what constitutes a user-friendly system. This limitation has been cited as a key reason for low levels of software acceptance (see, for example, Norman and Draper 1986). Second, despite the strong technical focus, the process fails to satisfy the technical criteria applied to other branches of engineering. Several alternatives to the waterfall model have been proposed as solutions to these and other development problems, the most widely used of which are incremental development, evolutionary development using prototypes, and rapid application development (RAD).

While incremental development represents a simple extension to the waterfall model, the use of prototyping in evolutionary development is a substantial change. A significant problem for many customers and users is that, although they can state their problem, they have insufficient experience of possible solutions to state their preferences between them and their contribution to design is, therefore, limited. Prototyping helps overcome this by providing users with experience of alternative software solutions and a mechanism for improving the dialogue between analyst, customer and user. Prototyping must be carefully managed, however, to avoid slowing down development or unduly increasing costs and is only useful for studying some aspects of systems (Olle et al. 1988). A critical decision when using prototyping is whether to throw away the prototype or develop it into an operational system. Throwaway prototyping raises the prospect of long development times, as with the waterfall lifecycle, whereas developing the prototype risks poor software quality and increased maintenance costs.

Rapid application development extends prototyping tools to enable the production of workable applications for certain problem domains. RAD is particularly well-suited to transaction-based processes with well-defined inputs and relatively simple representations, such as tables, for output. From the customer's and users' perspectives, RAD has several potential drawbacks that must be carefully managed. Crucially, the focus on developing database systems that can readily be modified may encourage a short-term perspective and, thereby, shorten the time between revisions of the system. This has three potential consequences: (1) systems changes are made in response to user requests without full consideration of the strategic/organizational implications; (2) the coherence and integrity of the system may deteriorate more rapidly; and (3) savings at the initial development stage are offset by increased maintenance/redevelopment activity.

As illustrated above, software engineering has had limited success in improving the quantity and quality of customer and user participation. The HCI discipline, however, regards its primary goal as the integration of user-centered techniques into software engineering (Dix et al. 1993; Sutcliffe 1995). As illustrated by the example below, HCI methods tend to ***augment*** the software engineering process with user-centered tools and techniques, rather than ***integrate*** them at the methodological level. Consequently, development remains document or risk driven, albeit with an increased awareness of customer and user needs. The "psychological and organizational tools" developed by Clegg et al. (1996) are related to the waterfall model in Figure 1 to illustrate the limitations of augmenting software engineering approaches. With the waterfall model, the tools can provide only a list of tasks to be computerized and a method for filtering out unacceptable solutions ***after*** the development process. An evolutionary prototyping methodology would resolve some of these problems by allowing usability evaluation to feed back into the task allocation and job design processes, but this would require some revision of the tools and how they are collectively applied.

In comparison with software engineering and HCI, the information systems community has proposed more radical approaches to user involvement in systems development. Lyytinen (1987), for example, presents a taxonomy of information systems development methodologies based upon three contexts: technology, organization and language. The taxonomy is used to illustrate how the different premises of methodologies lead to different perceptions of the development process. A central point in Lyytinen's analysis is the demonstration that methodologies are partial in terms of their coverage of the three contexts. The socio-technical approaches, for example, emphasize the organization and technology contexts relative to the language context. In terms of developing a lifecycle model to underpin development methodologies, the three contexts are suitably accounted for by the notion of social transformation, as illustrated by the models presented in the following sections of this paper. The language context relates to the capacities of users to manipulate symbolic representations, which are the skills constitutive of information systems. The technology context refers to the competence for developing and manipulating technology artefacts in order to gain access to symbolic representations and to supplement language skills. The organizational context refers to the social relations that bear upon the access, control and change of information artefacts. A lifecycle model that uses social transformation as its central precept provides the potential for making methodologies contingent in their balancing of the three contexts proposed by Lyytinen. Given that systems development implies organizational change, a contingent approach is preferable to the use of a taxonomy to guide the selection of a methodology that is fixed in its treatment of social organization.

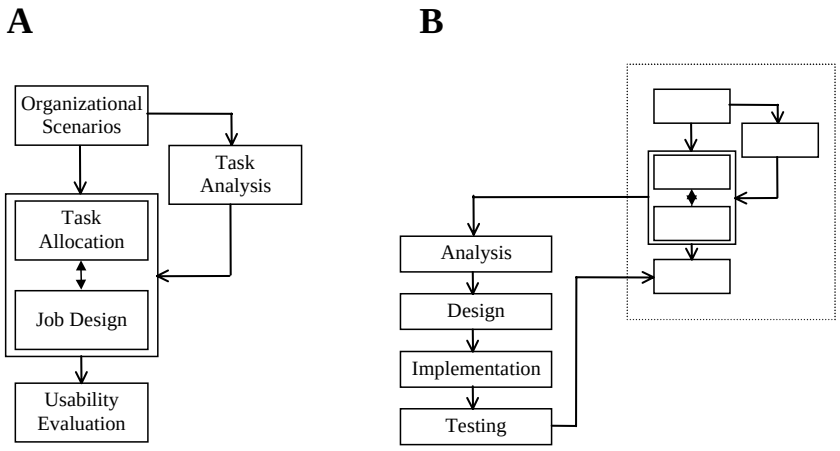

Figure 1. (A) Psychological and Organizational Tools (Reproduced from Clegg et al. [1996] courtesy of the Institute of Work Psychology, University of Sheffield, United Kingdom) and (B) Their Fit with the Waterfall Lifecycle

Iivari (1990a,1990b) develops a lifecycle model from a similar perspective to that adopted by Lyytinen. Extending Boehm's (1988) spiral model of the lifecycle and drawing upon the PIOCO development methodology, Iivari presents a three-phase evolutionary lifecycle. The three phases—organizational, technical and conceptual—are essentially synonmous with the three contexts proposed by Lyytinen. By incorporating these factors into a lifecycle model, Iivari addresses several of the problems identified by Lyytinen's taxonomy, resulting in a feasible, although somewhat complex, lifecycle model that is sensitive to social change. As illustrated above, Lyytinen's contexts and, hence, Iivari's three-phase approach focus on developing software rather than user skills. Thus, although Iivari recognizes that the evolutionary development involves a considerable amount of learning on the part of the developers, no explicit mechanism is proposed for ensuring that users participating in the development process have the requisite technical and conceptual skills. Furthermore, the lifecycle retains a focus on software and does not provide any specific process to support the development of users' information handling abilities.

Iivari, Hirschheim and Klein (1998) identify and compare five approaches to IS development that are significantly different than conventional software engineering. In a similar vein to Lyytinen's taxonomy, the analysis clarifies and contrasts the basic premises underlying the approaches. While this framework provides some useful insights, its analysis of ontological and epistemological positions has a number of weaknesses, the most significant being the failure to make a useful distinction between ontological and epistemological anti-realism (see Hacking 1983) and the narrow range of ontological commitments considered. The predominance of philosophical concerns in the analysis suggests that it is of academic, rather than practical, interest. Indeed,

Iivari, Hirschheim and Klein conclude with a number of comments about IS as an academic discipline. If extended to address the above weaknesses, however, the framework may prove a useful starting point for analyzing existing methodologies and considering how they might address issues of social transformation more thoroughly. Prerequisites for such methodological development are systems and software development lifecycles that regard social transformation as central to effective information systems development. The remainder of this paper describes such lifecycle models based upon the socio-cognitive theory of information systems.

3. Key Concepts from the Socio-cognitive Theory of Information Systems

The lifecycle models presented in the following sections draw upon the socio-cognitive theory of information systems (Hemingway 1999; Hemingway and Gough 1998). As the theory is only recently developed and not widely published, the concepts central to developing the models are outlined here. It is important to note that terms such as system and organization are used in the following sections according to their definitions in the socio-cognitive theory. The theory's definitions of agency, action, motivation and values are also used in developing the models, although their definitions are not provided in this paper.

3.1 Information Artefact

All communication takes place via some medium. Furthermore, some media can also store representations (i.e., arranged signs and symbols). Media capable of storage are classed as information artefacts and are characterized in terms of three modalities:

- **Modes of organization**: The use of, for example, spatio-temporal analogues or symbolic abstractions to encode a message into a medium.
- **Modes of selection**: The filtering of information prior to its representation.
- **Modes of navigation**: Mechanisms by which the user can interact with modes of organization and selection.

3.2 The Individual

When representing individual actors in an activity system, three factors are of the most interest: the individual's memories of their experiences as actors; the individual's interpretations of messages that he or she extracts from information artefacts; and the ability of the individual to act. The ability to act is dependent upon the physical relations between the individual's body and the environment and the individual's largely tacit knowledge of how to act in different situations. The individual's present knowledge of how to act defines his or her capacity for future action. The relationship between cognitive abilities, capacity and power are explained in detail in Hemingway (1999) and Hemingway and Gough (1999).

3.3 Social Organization

A distinguishing characteristic of the theory is its attempt to provide an integrated perspective on cognition, the body and communication. Thus, its analysis of social interactions begins with the consideration of physical co-presence, which is regarded as the basis for early forms of learning. Co-presence is important because it permits two or more individuals to refer to the same phenomena, even though they are all drawing upon their own experiences. Over time, the actions of co-present individuals can become regularized, thereby providing inter-subjective knowledge.

When considering social interaction within organizational forms, institutionalized practices and documented procedures and standards for action can be discerned (see, for example, Giddens 1984). It is important when intervening in social activity systems to have some understanding of the power relations that facilitate the maintenance of these regularized actions and standardized working practices. When considering information artefacts, the power issues can be considered in terms of (1) the actors' abilities to access information artefacts and interpret their message contents; (2) actors' abilities to control the message content of information artefacts; (3) authority over the access and control of information artefacts and their message contents; and (4) responsibility for the quality of information artefacts and their message contents.

4. Incorporating Social Transformation into the Information Systems Development Lifecycle

Social transformation is an intentional change to a social organization. In the context of information systems, such transformations must address six types of issues, which form the basis for any intentional information systems development:

1. the development of actors' capacities to represent, communicate and interpret messages and use the resulting knowledge to guide action;
2. the development of information artefacts to supplement the capacities identified in point 1;
3. the development of users' capacities to exploit ICT artefacts;
4. the provision of sufficient access to and control over information artefacts for the effective performance of tasks;
5. the alignment of power over with responsibility for information sources; and
6. the identification and resolution of deficiencies in the information available within the organizational form.

Information systems development begins when at least one stakeholder participating in an activity system perceives a deficiency in the system and subsequent analysis reveals that the deficiency can be resolved, at least in part, through changes to communication processes and information usage. Thus, as illustrated in Figure 2, the first two stages of the lifecycle are stakeholders' perceived dissatisfaction initiating the lifecycle, followed by the analysis of the activity system to determine whether the deficiency can suitably be addressed in information systems terms.

Where the contribution of IS developments is clearly identified, the next stage of analysis is to gain a detailed understanding of the stakeholders' perceived deficiencies and to classify these according to the six types listed above. The results of this analysis

Figure 2. An Information Systems Lifecycle Based Upon the Socio-cognitive Theory of Information Systems

of the activity system is the identification of activities and information artefacts that need to be modified. Having identified ***what*** needs to be changed, the analysis goes on to consider ***who*** will be affected by the changes. It should be noted at this stage that past and present power relations and information needs will have led to the emergence of

stable information flows and standardized information artefacts within the activity system. Many such flows will be components of regularized working practices. Thus, unless widespread dissatisfaction is apparent, it can generally be assumed that the *status quo* serves the participants in the activity system reasonably well or, at least, satisfies those participants with most power. Consequently, a key issue to be addressed when identifying stakeholders affected by the proposed change is the assessment of stakeholders' power over the affected information flows and information artefacts and their satisfaction with the existing organization. The balance of satisfaction and power is a key factor determining the feasibility of any changes to the activity system. During this analysis, it may be useful to distinguish between the following stakeholder categories:

- **Current Information Users**: Those participants in the information system who are presently able to access or control an information artefact and/or its content.
- **Potential Information Users**: Those participants who have legitimate access to information artefacts, but presently do not have the capacity to manipulate the artefacts and/or interpret the messages that they contain.
- **Secondary Information Users**: Those participants in the activity system who have power to direct the activities of any current information users with respect to the use of the information flows and information artefacts in question.
- **Activity Relevant Stakeholders**: Those persons in the activity system who will be directly or indirectly affected by changes to the information flows and information artefacts being considered.

Given that stakeholders may themselves lack the capacities to make the requisite changes to the system, the following points should also be considered at this stage:

- Do stakeholders with the capacity and authority (i.e., legitimate power) to change information artefacts perceive a need for change?
- Do stakeholders with the authority to make the required changes perceive the need for changes but lack the capacity to make them?
- Do stakeholders with the capacity to make the changes lack the authority to enact them?

The analysis of stakeholders in the above terms will provide a good indication of which system changes are likely to be successfully initiated. A further consideration at this stage is the extent to which the changes affect the goals of stakeholders. The analysis of goals is one of the most difficult aspects of information systems analysis because the goals of individuals, groups and organizations must be considered, yet, as illustrated by the ontology of the socio-cognitive theory of information systems, all such goals ultimately reflect the motivations of, and interactions between, individual stakeholders.

Having determined what changes are to be made and who is likely to effect, affect and be affected by the changes, the next phase of development is to plan the change process. Owing to the diversity in goals and the complex network of power relations, the planning process takes the form of a negotiation cycle, illustrated by the bold lines in Figure 2. Accounting for the findings of the two analysis stages, an initial change plan is formulated. The direct and indirect impacts of the plan are then identified, perhaps including some consideration of the likely extent of "ripple effects" resulting from the changes made to the activity system. The anticipated impacts are then compared with the findings of the analysis to evaluate the feasibility of the planned changes. While an entire spectrum of findings may result from the evaluation, four broad types of outcome may result:

- the proposed changes are not implemented because the likelihood of success is low, the benefits are low and/or the associated risks are high;
- a second analysis phase is conducted because the proposed changes are found to be of low feasibility but are perceived to be feasible in the context of a change process that is essentially different in scope (i.e., it is realized at some stage that the perceived problem that triggered the lifecycle is not actually the central problem that needs to be addressed);
- the change plan is modified as a result of improvements identified during the evaluation of the initial plan; or
- the change plan is executed.

As illustrated by the negotiated implementation cycle (shown as a dashed line in Figure 2), it is not possible in practice to distinguish between the negotiation of the change plan and the implementation of change. The reason for this is that the precise nature of the change and its consequences cannot be completely predicted and, consequently, any actual change to an activity system must itself take place through a negotiation process. The negotiation of implementation is shown to include further analysis precisely because the impacts of the changes need to be understood and responded to if change is to be successful (analysis at this stage is not, of course, as detailed as during the initial stages of IS development). During the process of changing the activity system, new information artefacts may need to be introduced or existing artefacts redesigned. A guide for considering social transformation when developing information artefacts is presented in section 5. Where the information artefacts are ICT-based, a software development lifecycle, as described in section 6, will constitute part of the implementation cycle.

Following the negotiated changes to information artefacts and information usage, the activity system will eventually stabilize and negotiations will significantly reduce, or even cease. Provided that no other significant changes to the system occur, modified processes will eventually become regularized and accepted as social norms. It should be noted that the lifecycle described above has various iterative phases. Furthermore, numerous change processes can co-occur and interact, with the initial change process sometimes stimulating other changes. Such knock-on effects can even occur at the planning stage because the discussion of systems developments can affect the satisfaction of stakeholders with other aspects of the activity system. Although the change plan has effectively been completed at this stage, it is essential that monitoring of the changes takes place while the system settles into a new routine. If it becomes apparent that unexpected side-effects of the change are becoming problematic, a new information systems development lifecycle may be initiated.

5. Social Transformation and the Development of Information Artefacts

A key component of many information systems developments is the creation or modification of information artefacts. To provide a basis for the user-centered design of such artefacts, which includes software systems, Table 1 relates the key concepts of social transformation to three modalities that can be used to describe information artefacts.

Table 1. Socio-cognitive Concepts Used in the Design of Information Artefacts

Information Artefact	Individual	Social
Organization	Memory	Regularization and standardization
Selection	Interpretation	Communication
Navigation	Capacity	Power (Access, control, authority and responsibility)

Each row of Table 1 permits the characterization of information artefacts in their present and planned future states in terms of the social transformations they will support. Considering the first row, an information artefact can be regarded as located on a continuum. At one extreme, the artefacts are standardized to reflect well-established practices within the activity system (this is common in, for example, accounting systems). At the other extreme are flexible artefacts that can encode highly individualized messages. Typical examples include "notes" fields in structured databases and an employee's personal records about a client. Messages encoded in unstructured artefacts are subject to broader interpretation than standardized messages and are not always well-understood by individuals outside their immediate context of use.

Considering the second row of Table 1, information artefacts can be located on a continuum representing the encoding and extraction of messages. The possible means for encoding and extracting information are determined by the possible modes of organization. An artefact that encodes the details of numerous customers, for example, may be sorted in a particular order (or permit certain methods of sorting). The resulting organization affects the ways in which parts of the encoded message can be encoded and extracted, and the relative efficiency of encoding and extracting messages. Of most significance, however, is whether users interact *through* the information artefact or interact *with* it. The distinction between organization and selection can be subtle because, particularly with static artefacts, such as paper forms, the modes are closely coupled. Organization refers to what can be encoded by an information artefact and how. Selection refers to the means available to a user for manipulating a message during its interpretation.

The third row of Table 1 does not represent a continuum, but a number of factors that need to be considered to ensure the stability and effectiveness of the information system. Access to information artefacts relevant to a task is often critical to the effective performance of the task. Control refers to the ability of the user to modify the message content of an information artefact in order to communicate information to other users. Authority refers to the legitimate means by which a stakeholder may limit access to, and control over, information artefacts and their message contents. Responsibility for information artefacts and their message contents identifies the stakeholders who will be rewarded or sanctioned to reflect the quality of the information artefact and its message contents. Much of the analysis relevant to balancing these issues in a systems context are addressed by the IS development lifecycle. In terms of developing artefacts, the main concern is providing modes of navigation by which information access and control can

be managed and, particularly in terms of software systems, authority and responsibility implemented.

6. Incorporating Social Transformation into the Software Development Lifecycle

As illustrated in section 4, information systems development addresses six interrelated concerns. The information systems community has typically focused on issues three to six, with comparatively little attention paid to skills issues during the development process. Although the authors stress that most benefits will arise from IS developments that address all six issues and their interrelationships, this section extends the IS development lifecycle to propose an alternative to the software development lifecycles considered in section 2.

The software development lifecycle is an abstraction of specific software concerns from the more general IS development lifecycle. This abstraction can be achieved with the above lifecycle by identifying the ICT artefact requirements that can be effectively supported by a software solution. The decision to develop computer-based information artefacts must take into account the implications particularly in terms of required ICT skills, but also in terms of "information ownership" and other power-related issues that are affected by the properties of information artefacts. The analysis presented in section 5 provides a framework for appraising the likely impacts of information artefacts on power relations in an activity system, which can be used as a guide for software design. The use of Table 1 as a framework is quite readily achieved because the description of information artefacts in terms of organization, selection and navigation relates quite directly to concepts used in conventional data modeling and software design. Modes of organization, for example, correspond with the logical data model in terms of data storage and human-computer interface design. Modes of selection correspond with both user and system operations for encoding, sorting and filtering. Modes of navigation refer to the dialogue between human and computer and, at the systems level, to the logical arrangement of data files and the interconnections between them that are achievable with the operating system and applications available. Further research needs to be conducted before a detailed software lifecycle and development methodology can be produced. An indication of future work by the authors is presented in section 8.

7. Conclusions

Only limited success has followed the attempts by both the software engineering, IS and HCI communities to change information systems and software development practices to make them more user-centered and more sensitive to the organizational and human context in which development takes place. The reason for such limited success is that user-centered design can only be fully realized through the adoption of user-centered lifecycle models on which to base development processes.

User-centered systems and software development need to be integral to information systems theory and practice. This end can be achieved by incorporating social transformation into these lifecycles, which is made possible by the socio-cognitive theory of information systems.

8. Future Work

Future research issues relevant to the further development of the socio-cognitive theory of information systems and the lifecycles presented in this paper, which the authors aim to address, are:

- the study of information skills and how they can be developed in users;
- the relationships between users' information skills, task characteristics, the properties of information artefacts and task performance;
- ways of improving user capacities for developing information artefacts and information systems to support their tasks;
- the negotiation of development capacities between IS and software engineering professionals and users;
- the development of detailed IS and software development methodologies based upon the lifecycle models presented in this paper; and
- the evaluation of alternative software solutions in terms of their information and IS development skills requirements.

References

Avison, D. E., and Fitzgerald, G. *Information Systems Development: Methodologies, Techniques and Tools*, 2nd ed. Maidenhead, England: McGraw-Hill Publishing Company, 1995.

Boehm, B. W. "A Spiral Model of Software Development and Enhancement," *Computer*, 1988, pp. 61-72.

Clegg, C.; Coleman, P.; Hornby, P.; Maclaren, R.; Robson, J.; Carey, N.; and Symon, G. "Tools to Incorporate Some Psychological and Organizational Issues During the Development of Computer-based Systems," *Ergonomics* (39:3), 1996, pp. 482-511.

DeMarco, T. *Structured Analysis and Systems Specification*. London: Prentice-Hall, 1979.

Dix, A.; Findlay, J.; Abowd, G.; and Beale, R. *Human-computer Interaction*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

Ferguson, J., and Sheard, S. "Leveraging Your CMM Efforts for IEEE/EIA 12207," *IEEE Software*, September/October 1998, pp. 23-28.

Giddens, A. *The Constitution of Society*. Cambridge, MA: Polity Press, 1984.

Hacking, I. *Representing and Intervening: Introductory Topics in the Philosophy of Natural Science*. Cambridge, England: Cambridge University Press, 1983.

Hemingway, C. J. "Toward a Socio-cognitive Theory of Information Systems: An Analysis of Key Philosophical and Conceptual Issues," in *Information Systems: Current Issues and Future Changes*, T. J. Larsen, L. Levine, and J. I. DeGross (eds.). Laxenburg, Austria: International Federation for Information Processing, 1999, pp. 275-286.

Hemingway, C. J., and Gough, T. G. "A Socio-cognitive Theory of Information Systems," Technical Report 98.25, School of Computer Studies, University of Leeds, United Kingdom, December 1998.

Herbsleb, J.; Zubrow, D.; Goldenson, D.; Hayes, W.; and Paulk, M. "Software Quality and the Capability Maturity Model," *Communications of the ACM* (40:6), 1997, pp. 30-40.

Iivari, J. "Hierarchical Spiral Model for Information System and Software Development Part 1: Theoretical Background," *Information and Software Technology* (32:6), 1990a, pp. 386-399.

Iivari, J. "Hierarchical Spiral Model for Information System and Software Development Part 2: Design Process," *Information and Software Technology* (32:7), 1990b, pp. 450-458.

Iivari, J.; Hirschheim, R.; and Klein, H. K. "A Paradigmatic Analysis Contrasting Information Systems Development Approaches and Methodologies," *Information Systems Research* (9:2), 1998, pp. 164-193.

Lyytinen, K. "A Taxonomic Perspective of Information Systems Development: Theoretical Constructs and Recommendations," Chapter 1 in *Critical Issues in Information Systems Research*, R. J. Boland and R. A. Hirschheim (eds.). Chichester, England: John Wiley and Sons, 1987.

Mumford, E. *Designing Participatively: Participative Approach to Computer System Design*. Manchester, England: Manchester Business School, 1983.

Norman, D. A., and Draper, S. W. *User Centered Systems Design: New Perspectives on Human-computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates, 1986.

Olle, T. W.; Hagelstein, J.; Macdonald, I. G.; Rolland, C.; Sol, H. G.; van Assche, F. J. M.; and Verrijn-Stuart, A. A. *Information Systems Methodologies: A Framework for Understanding*. Wokingham, England: Addison-Wesley Publishing Company, 1988.

Royce, W. W. "Managing the Development of Large Software Systems," in *Proceedings of WESTCON*, San Francisco, CA, 1970.

Sutcliffe, A. *Human-computer Interface Design* 2nd ed. Basingstoke, England: MacMillan, 1995.

Yourdon, E. *Design of On-line Computer Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1972.

About the Authors

Christopher Hemingway is with the Information Systems Research Group, School of Computer Studies, at the University of Leeds, United Kingdom. His main research interests are user-centered systems and software development; systems and software evaluation; and computer support for the modeling of information by experts and the communication of information between experts and non-experts. His research provides an integrated treatment of these issues through the development of the socio-cognitive theory of information systems. He can be reached by e-mail at cjh@ieee.org.

Tom Gough is also with the Information Systems Research Group, School of Computer Studies, at the University of Leeds, United Kingdom. His research interests relate to the "why" and "how" of information systems and include user-centered development and co-developing the socio-cognitive theory. Tom also has a general interest in medical informatics, particularly addressing health and safety issues during in IS development. He can be reached by e-mail at tgg@scs.leeds.ac.uk.