3 INFORMATION SYSTEMS DEVELOPMENT AS FLOWING WHOLENESS

Paulo Rupino da Cunha
António Dias de Figueiredo
Universidade de Coimbra
Departamento de Engenharia Informática/
Centro de Informática e Sistemas da Universidade de Coimbra
Pólo II, Pinhal de Marrocos
3030 Coimbra
Portugal

Abstract

This paper asserts that information systems development (ISD) should be understood as a continuous and holistic process. To support this view, ISD is analyzed in light of the current challenges of permanent business model innovation and of the ensuing pressures on the fast, but controlled, adjustment of the supporting information systems. In a context of growing complexity, increased interaction between people, departments, and enterprises, wide availability of heterogeneous enterprise software applications that call for integration, and concerns about the preservation of legacy, the need of completely new approaches to ISD becomes absolute. This paper describes an approach we have developed to this end, and shows how it can be used to fulfill the aim of continuous and holistic ISD.

1. INTRODUCTION

More than 20 years ago, the English physicist David Bohm proposed a theory that explained the universe as a flowing wholeness. Inspired by the ideas of Heraclitus, the old Greek philosopher who pointed out that "you cannot step twice into the same river, for the waters are continually flowing on," Bohm described the world as unfolding and enfolding from moment to moment as a kind of pulsating wholeness (Morgan 1997). This metaphor of stability and wholeness within permanent flow inspires our view of the current state of information systems development.

As the basis of competition and wealth creation in the digital age becomes, more and more, a process of business model innovation (Tapscott 1999) and the relationship between business model and application architecture grows closer, information systems development becomes a *continuous* process. Developing an information system ceases to be an end; it becomes one of the most strategic permanent activities of a business, inseparable from business strategy itself.

This causes application management decisions to rise to the top of the management agenda (Kalakota and Robinson 1999). Originally relegated to the information technology (IT) department, information systems development is now becoming a shared venture that needs to be deeply understood, not just by information systems experts, but also, and increasingly so, by senior managers.

On the other hand, the growing complexity of applications, the severe pressures of time-to-market, the need to preserve some legacy, the availability of modular applications of growing sophistication, such as enterprise resource planning (ERP) and customer relationship management (CRM) solutions, the extension of the enterprise up and downstream, and the advent of enterprise communities are insistently calling for completely different approaches to information systems development.

The big challenge of information systems development is not, anymore, that of building up a complex *homogeneous* system, whose "blueprint" is a highly detailed technical document, specifying logic and data structure that programmers use to code the solution. The big challenge is that of building an eminently *heterogeneous* system from very distinct parts (some of them complete packages) that come from varied sources, at different times. It is also the challenge of weaving all those distinct parts (of potentially different ages) into a whole without loosing sight of that whole. It is, finally, the challenge of doing it *continuously*, in a permanent flow, so that the continual changes to the business model can be accommodated by the information system in an evolutionary way.

Continuous development, *wholeness*, *integration* of existing and new complex and disparate parts, and friendliness towards the various actors involved, namely senior managers, are the key features of our proposal for information system development.

As (Baskerville 1999) points out, studying new or changed methodologies implicitly involves the introduction of changes, making it impossible, from a social-organizational viewpoint, to get by without injecting the new technique into the practitioner's environment. Not many research approaches maintain their validity in such a context. Thus, we used action research, as one of the few approaches that can be legitimately employed to study the effects of specific alterations in systems development methodologies in human organizations (Baskerville and Wood-Harper 1996). The cyclic nature of action research is one its main supports of rigor and validity (Baskerville and Wood-Harper 1996; Dick 1997; Dick and Swepson 1994; Lau 1999). Practice and theory inform each other synergistically—in a convergent process across iterations—enabling the continuous *evaluation* and adjustment of the emergent framework—in this case, the proposed methodology (Avison et al. 1999). The authors develop their practice through a university/enterprise interface institution that is constantly contracted to carry out projects for the public and private sectors. This is what made possible an intervention where action-research can be used to solve complex real-world problems.

We chose four such projects, selected on the basis of the characteristics that could stress the methodology. The information systems for three of the projects are still in the design phase, while the fourth went on-line in the third quarter of 2000. Users and managers were much pleased with the outcome, and are already requesting a new contract in which the information system will be extended to cover new areas of the organization.

It is interesting to note that, in the various projects, our premises hold. Senior managers have been the driving force for the evolution of the information systems and users are deeply involved (which reinforces the need for the ISD approach to be intelligible); a significant amount of legacy systems that still meet organizational needs are being integrated and leveraged to new uses (not discarded and replaced); several needs are effectively being satisfied by purchasing available software packages; custom development is restricted to integration purposes and issues very specific to the organization. The visible facet of the resulting information systems is often a user friendly intranet that hides complexity while integrating and leveraging into a coherent whole the dispersed legacy and the new solutions, regardless of their sourcing: acquisition, rental, custom development, or outsourcing.

In the following section, we start by looking into the implications of the increasingly deep relationship between business models and the process of information systems development. This leads to the recognition of the need of new ISD paradigms, which is discussed in the next section. A brief introduction is then made to a methodology previously proposed by the authors, which is summarized so as to provide a common ground for the explanation, in the following section, of how continuous development can be handled. We conclude with a brief summary and some avenues for future work.

2. BUSINESS MODELS

With most organizations now developing their information systems solutions for the "new economy," business modeling is becoming the most central part of project development, with model-based technologies being actively sought to develop fast and in a controlled manner (Kruchten 2000). In this new approach, CEOs and marketing directors are deeply involved in developing the models, rather than just giving broad instructions to "'business domain experts' that might have known how the business is run but were not empowered to make decisions about changing it" (Kruchten 2000). Indeed, business development is becoming more and more a reflection on the nature of the business and the way it is run, involving "people from the various parts of the organization, from executives with the power to make decisions, to 'grass roots' and end users who feel the consequences of those decisions" (Kruchten 2000).

3. THE NEED FOR NEW ISD PARADIGMS

Information systems, once fairly *homogeneous* software solutions strongly based on custom coding, are now mutating into *heterogeneous portfolios of applications* where ready-made software, such as enterprise resource planning (ERP) packages or shrink-wrapped software from various vendors, plays an increasingly important role in supporting continuously evolving business needs. The code that remains to be written tends, at present, to concentrate mainly on the customization of packaged software, on the development of fragments of middleware that perform the required integration between packages, and on small developments that support distinctive business facets (Asbrand 1999; Stephens 1998;Ward and Peppard 1996).

When accepting this reality, we must also recognize that the *portfolios of applications* increasingly need to combine systems of various ages and in different stages of technological evolution. Indeed, the concept of legacy—a traditional hindrance factor when developing "the" information system with most ISD methodologies—no longer makes sense: the innovation of today is often the legacy of tomorrow, and we must be able to take in as much of that legacy as possible when evolving an information system if we wish to avoid drastic ruptures in our budget and in the sense of belonging of our users.

In addition to all this, the accommodation of intranets and extranets, the integration of portal services, the advent of the extended enterprise, and the perspective of inter-enterprise communities to which we must belong contribute to change the whole concept of modern information systems development.

Figure 1 represents a typical information system of the present. We can see how custom developed monolithic information systems of the past have given place to a set of specialized autonomous solutions that are made to work together: SAP for finance and production planning; an AS/400 mainframe solution for plant automation; an Oracle data warehouse gathering data from SAP and feeding it to Clementine's analysis software, and so on.

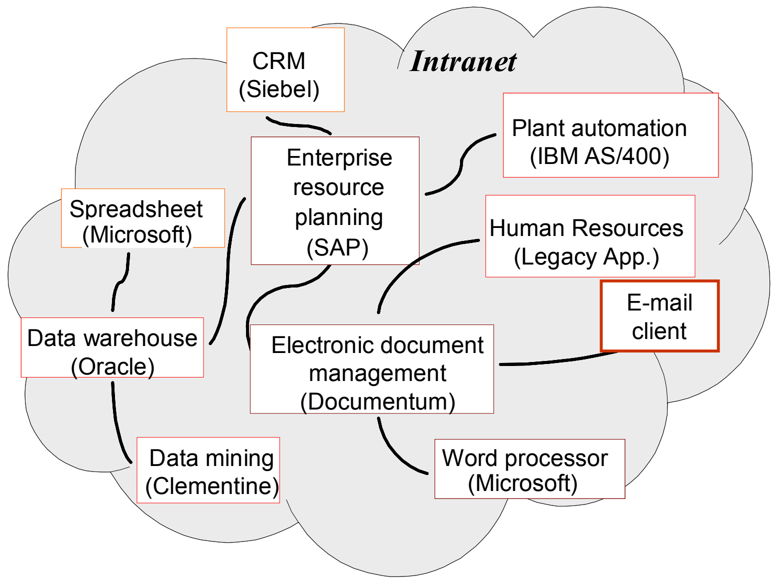

Figure 1. A Typical Information System of the Present

This new reality, the information system as an *heterogeneous application portfolio*, which is observable in actual field deployments made by the IT departments of the organizations, is at odds with the results from most traditional information systems development methodologies, whose deliverables are difficult to translate into solutions that can be feasibly deployed in this new context. Such a conflict is patent in the strong criticism to existing methodologies (Avison and Fitzgerald 1997, 1999). Our belief is that the ineffectiveness for which traditional methodologies are now being questioned can, in part, be attributed to the fact that they are rooted in practices and concepts that were relevant to completely different organizational and technical realities (Fitzgerald 1994, 2000).

Many such methodologies still put at the top of the agenda issues such as the excellence of the technical solution, unambiguity, rigor, and completeness, which were fundamental when information system solutions were to be built from scratch, with extremely detailed descriptions of logic structures and data models. The drive from building to buying, under the pressure of ever increasing demands of speed of development, and the availability of a growing market of ready-made solutions are putting the emphasis on values such as business precedence (that not only dictates needs, but also timing), speed of deployment, distinct allocation of resources according to expected return on investment, flexibility, continuous management of the evolving business need, and their relative priorities, and user involvement.

These challenges call for approaches that are *sufficiently light* to be used in an ongoing manner, lending themselves to *continuous reappraisals* of the information system and of its alignment with the changing business objectives. Information system development is becoming, indeed, much less of a destination and much more of a journey that goes on forever.

Much wider *visibility* over the "whole" information system and much *stronger links* between business needs and supporting solutions are needed if we want to effectively manage the allocation of resources in agreement with the value that each component offers to the business at a given time. A particular instance of this type of concern regards the decisions that need to be made along the build/buy/rent axis. Indeed, the approach to development must now assist in choosing the right combination of components to be integrated, which may be built in-house, bought, outsourced, or even, today, rented out as services granted by Application Service Providers (ASP), third parties that host and manage applications for a rental fee (Booker 1999; Keegan 1999; Mateyaschuk 1999; Nickell 1999; Seymour 1999).

This shows that there is a clear change in what is now perceived as the useful results of an information systems development methodology. The challenges are quite different from those that existed when most of the current information systems methodologies were conceived.

In order to meet these changed demands, we have proposed a new information systems development methodology (Cunha and Figueiredo 2000). This methodology did not evolve from any of the existing approaches; it has been conceived from the ground up to meet the challenges identified above.

Considerable thought has been given to the decision of developing yet another methodology, when the argument goes that there are too many already. In fact, there aren't actually many substantially different ways to design systems (Fitzgerald 1994). The differences between many methodologies are trivial (Veryard 1985), some of those differences are due to "personal ego and territorial imperative" (Constantine 1989), and others due to marketing purposes (Avison and Fitzgerald 1999). On the other hand, the number of so-called information systems development methodologies is highly inflated, since fundamentally different things are frequently grouped under the same umbrella (see Siau and Rossi 1998). Finally, it should be regarded as natural for new methodologies to emerge, since organizational and technological environments are also changing noticeably. Developing new methodologies is perfectly acceptable, as long as the problem that they are intended to solve is clearly identified and stated, so that methodologies can be easily discarded when their foundations no longer hold.

In section 4, we summarize our approach so as to set a common ground that lets us explain, in section 5, how it can deal with continuous development.

4. THE PROPOSED APPROACH IN BRIEF

Our proposal handles organizations and systems design from a different perspective, around two key concepts: *organizational entities* and the *responsibilities* for which they account.

An *organizational entity* can represent several realities, ranging from the clearly defined "divisions," typical of more mechanist organizations, to versatile configurations, or even teams, characteristic of more organic types of organization (Morgan 1997). As to *responsibilities*, they are the major "services" that *entities* provide to their environment. The use of such services requires clients to employ predetermined interaction protocols. This is a high-level form of responsibility driven design, a well-supported theory that has been successfully applied in different contexts (Wirfs-Brock and Wilkerson 1989; Wirfs-Brock et al. 1990).

For a simple illustration of the use of these two concepts, we can think of a typical "human resources" division as being an *organizational entity* and the services it provides—such as "justification of absence"—as examples of *responsibilities*.

Figure 2 illustrates the diagrammatic representation of an *organizational entity* and its *responsibilities*.

One of the key characteristics apparent in the models built using only the *organizational entity* and *responsibility* concepts is their higher granularity when compared to those that result from traditional methodologies. There are two main reasons why this design approach was chosen:

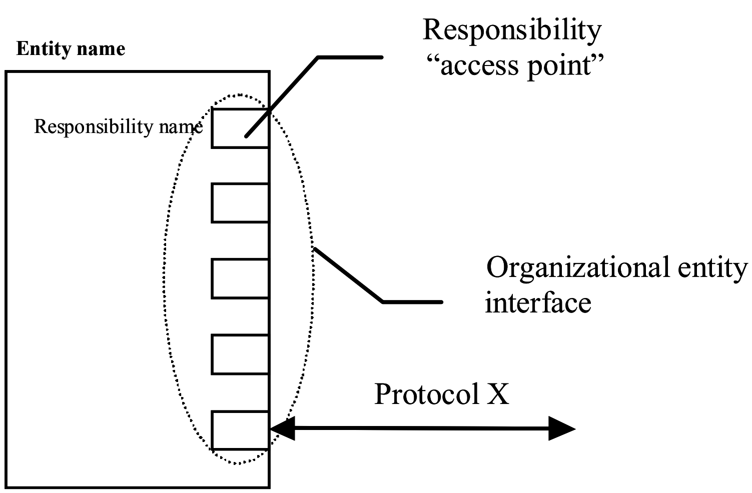

Figure 2. An Organizational Entity and Its Responsibilities

1. To perform their everyday duties, workers do not really need to know, or care, how the various *organizational entities* carry out their procedures internally. On the other hand, it is important for them to have a clear view of how the *responsibilities* are distributed inside the whole organization, and how to interact with the ones they need to use.
2. The encapsulation implied in the previous point is further backed by the fact that, nowadays, most *responsibilities* can be supported by ready-made applications that can be purchased. In this scenario, the great detail offered by the architectures that result from many traditional ISD methods is frequently excessive and useless, as much of the logic or data structure they express is already embedded in a software solution that may be purchased, outsourced, or rented.

   Recalling the earlier example of the human resources division, we should note that elaborate descriptions of the internal procedures in terms of objects or data structures and processes may prove excessive, as it is very likely that the data supplied by the worker (at the *responsibility* interface) will end up being entered to a "standard" human resource module of an enterprise resource planning (ERP) package.

Besides producing a better fit to current information systems reality, as shown, the proposed modeling approach is also closer to non-specialists: users and managers whose understanding and endorsement is critically necessary for any large-scale project. By centering the dialogue on concepts and issues that relate to their everyday work, we effectively narrow the communications gap still common in most traditional methodologies (Davenport 1997). People consequently adhere more easily to the planning projects and provide significant contributions much earlier.

This characteristic of the approach is reinforced by the use of a field instrument of renowned pedagogic and conversational qualities: the CRC card (Beck and Cunningham 1989; Cunningham 1994; Mitchell 1997; Taylor 1995; Wilkinson 1995).

Figure 3 shows the "front" of a CRC card used in our approach. A card corresponds to an individual *organizational entity*.

To accommodate the modeling of the organization, we have introduced changes to the original CRC card: instead of identifying, for each *responsibility*, whose collaboration it needs, we take a more customer-centered approach and ask, for each one, "whom does it serve." This is consistent with the desired encapsulation of the way in which *responsibilities* are handled inside *organizational entities*.

Two categories of information are obtained when filling the CRC cards that correspond to the various *organizational entities*:

Organizational entity: Human resources	Upper level Organizational entity:
Responsibilities	Client entities
Justification of absence	*Workers*
Access to workers contracts	*Workers* *Social security* *IRS*
Composing Organizational entities	

Figure 3. A CRC Card for Use in Organizational Modeling

1. Patterns of interaction among *entities*, derived from the use *entities* make of each other's *responsibilities*.
2. Details concerning each *responsibility* (recorded on the "back" of the CRC card), such as its aim and organizational procedure, significance to the business, interaction protocol, and information systems support. It is important to mention that both the present situation and the intended (future) improvements are collected in the descriptions of the organizational procedures and its information systems support. This enables a clear perception of the "roadmap" of the information system at any time, for any *responsibility*.

The significance of the *responsibility* to the business is a key issue, since it is later used to support the correct choice of priorities and sourcing strategies for the various supporting information system components. In order to be effective, this classification mechanism needs to be comprehensible by non-specialists, simple enough to encompass natural variations in circumstances, unbiased toward any particular planning methodologies, and centered on business concerns. These requisites led us to the adoption of McFarlan's Strategic Grid (McFarlan 1984). The grid concept was originally conceived to evaluate the importance of information systems to businesses as a whole (such as banking or insurance), but has since been extended and refined to enable the classification of the various components of an information system (Edwards et al. 1991; Ward and Peppard 1996). Figure 4 shows a strategic grid. Information system components are positioned into one of its quadrants according to the key issues

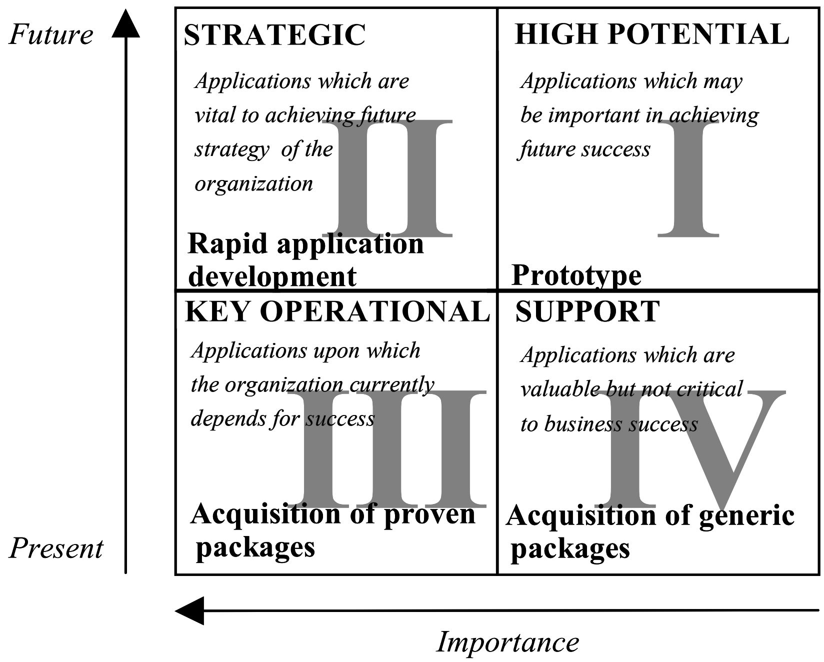

Figure 4. A Strategic Grid and Preferred Sourcing Alternatives According to System Type

listed. For each of these four categories, a preferred sourcing strategy (shown in bold) exists (Edwards et al. 1991).

The deliverables of the proposed approach are quite different from those that are obtained from traditional methodologies. They can be engineered to be immediately leveraged by IT departments, with no gaps between development and deployment strategies. On the other hand, they lend themselves nicely to the continuous development of the information system. Two examples of immediate application are presented below. The implications on continuous development will be addressed in the next section.

Various types of documents can be immediately built from consolidating, recombining, and grouping the various pieces of information collected for every *responsibility*. The descriptions of the (existing and improved) business procedures underlying the *responsibilities*, together with those regarding (existing and required) information systems support provide the foundation for the various approaches to deployment. The choice between approaches is carried out using the *significance* that has been established in terms of the strategic grid, also collected for each *responsibility*. Depending on this parameter, the information collected for each *responsibility* may be used differently: as the *terms of reference* for the acquisition of *key-operational* systems; as a statement of requirements for the team that takes over and proceeds with adequate software engineering techniques, such as fast *prototyping*, in the case of *high-potential* applications, or *rapid application development*, in the case of *strategic components*, and, finally, as the guide to a low-overhead market survey for subsystems belonging to the *support* quadrant.

A second major deliverable takes the form the interaction diagrams that can be derived from the use *entities* make of each other's *responsibilities*. This knowledge also enables the identification of access profiles and custom user interfaces for intranets and extranets that will support these interactions. Since interactions are derived from *responsibility* utilization, an immediate link also exists to the information system component that supports it, allowing global coherence in the design, in spite of the naturally heterogeneous technical environments.

Since we have taken the methodology out of the lab and into the field, in the context of our action-research approach, an important issue became apparent when considering the implications of producing these types of results from the information that is collected for each *responsibility*, belonging to every *organizational entity*: although the concepts on which the proposed approach is rooted ensure efficacy (doing the right thing), there is also a need to ensure efficiency (doing things adequately). For this purpose, a software tool has been developed to support the use of the approach in the field by automating tasks that would otherwise be very time-consuming, error prone, and dull.

The key characteristic of this tool is the use of a relational database to store all of the collected data. This enables diversified queries to lead to reports (in textual or diagram form) that provide different views of the information extracted from the model. The user-friendliness of the tool that implements an electronic version of the CRC cards previously introduced helps in leveraging the potential of the database.

A general view of the tool is show in Figure 5.

The same need for efficiency, felt in producing the "first iteration deliverables," also makes the tool vital in enabling the continuous development of the information system, as will become evident in the next section.

5. DEALING WITH CONTINUOUS DEVELOPMENT

Two issues that are crucial for effective continuous development are a clear visibility over the whole information system and a light process for intervention. The core concepts of the approach, backed by the software tool, go a long way in making this possible.

Holistic, yet manageable, views of the "whole" information system are enabled by the classification of each *responsibility* according to its value for the business, together with the direct links maintained between each one and systems that support them. A particularly useful representation is obtained by mapping the *responsibilities* onto a McFarlan grid, as shown in Figure 6.

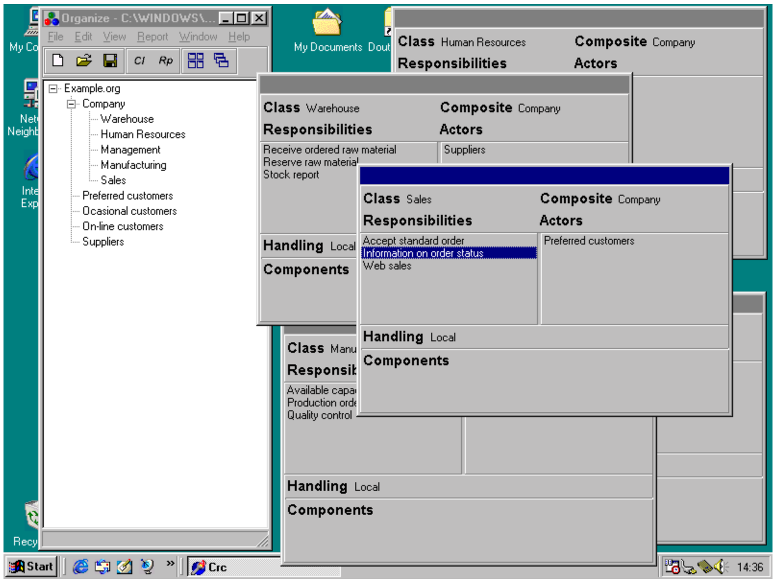

Figure 5. General View of the Modeling Tool

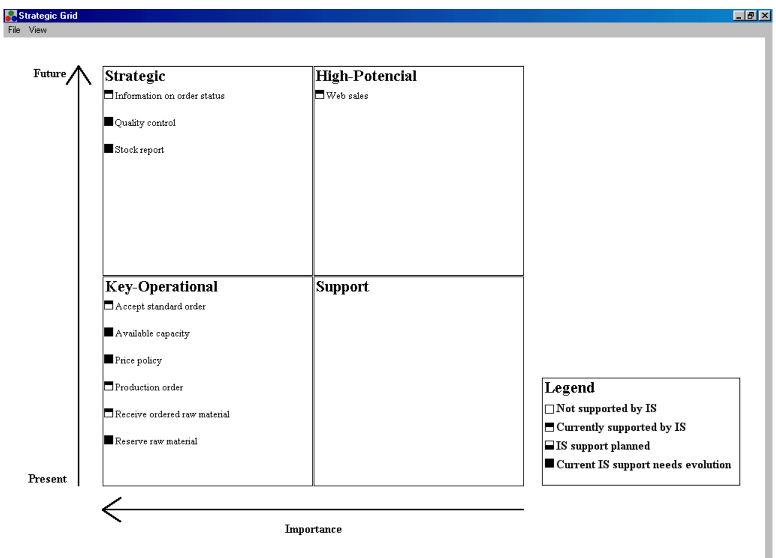

Figure 6. Responsibilities and Respective Information System Support Mapped to a Strategic Grid

Using this kind of overview, one can easily find out the status of information systems support for any given business *responsibility;* namely, if it is unsupported, supported, or if current support needs evolution, as the legend of the Figure 6 suggests.

Depending on the importance of a *responsibility* (i.e., of the grid quadrant in which it is placed), a suitable strategy can then be devised for intervention, including prioritization and decision regarding the appropriate form of sourcing. The full range of alternatives is considered, without any bias toward building instead of buying, as tends to happen with traditional methodologies.

This is made simple because the emphasis on the (high level) definition of *responsibilities*, and on how they can be used, encapsulates the internal complexity of their implementation. Any system, state-of-the-art or "legacy," is valid as long as the ultimate organizational objective is met. Systems of different ages, based on different technologies, and obtained from different sources, can be employed and made to coexist in a seamless way. Through time, brand new systems can be added to support new or previously unsupported responsibilities, and existing systems can be upgraded or phased out and replaced if and when deemed necessary from a business viewpoint, rather than out of mere technical motivations.

A roadmap for the information system can also be derived at any time, by customizing the information in Figure 6 and taking advantage from the fact that, during modeling, information is collected regarding both: current situation and intended improvements for information system support. As an example, Figure 7 shows a view of the *responsibilities* currently supported by the information system and Figure 8 shows another view regarding planned evolutions.

Note that current situation and intended improvements are also collected for the organizational procedures themselves underlying every *responsibility* (whenever applicable). This dual focus—present and future—was embedded on the modeling approach so that, working in concert with the lower level of detail requested, it could lessen the probability of occurrence of goal displacement (De Grace and Stahl 1993). This problem, common in traditional methodologies, frequently leads to the production of very accurate descriptions of the *status quo* at the expense of losing the objectives of change (Davenport 1997; Fitzgerald 1994).

Being based on *responsibilities*, the approach also becomes modular. Not only can several teams work in parallel, with minimum coordination, when performing the initial organization-wide design of the information system, but its subsequent continuous development can also be done in this way.

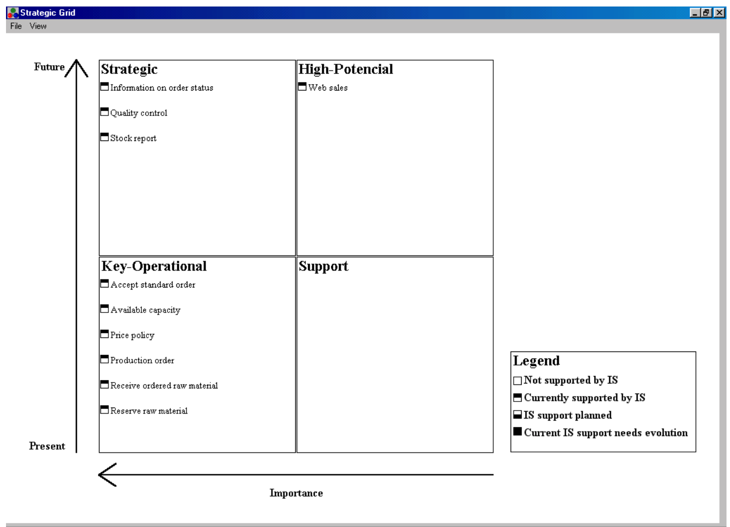

Figure 7. Current Information System Support Coverage

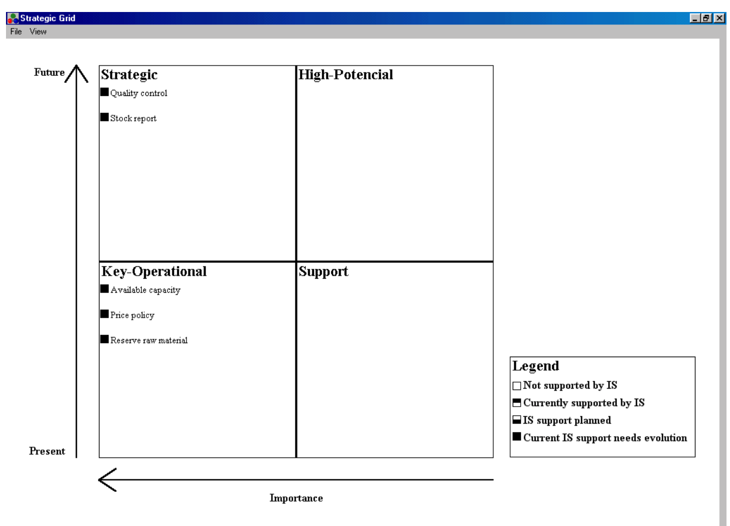

Figure 8. Planned Interventions on Information System Support Coverage

Thus, the reasoning and systems views just illustrated for the whole organization can be performed at the level of any single *organizational entity*, such as "warehouse," "manufacturing," "sales," or any other, in the earlier example. Other views over the information collected on the CRC cards are also useful in the continuous development process, such as the interaction diagrams, that promote awareness over the relationships among the various *entities*. Figure 9 and Figure 10, for example, show the "clients" for various *responsibilities* of the "manufacturing" *entity*, and its use of services from other *entities*, respectively.

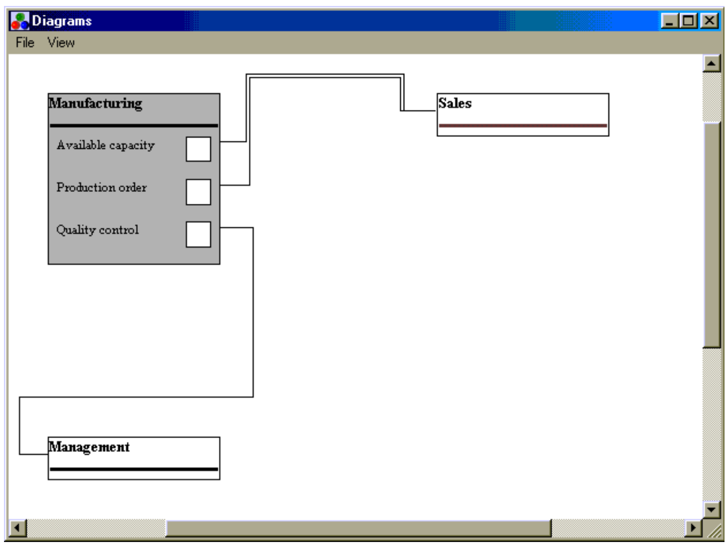

Figure 9. View Showing the Clients for a Given Entity

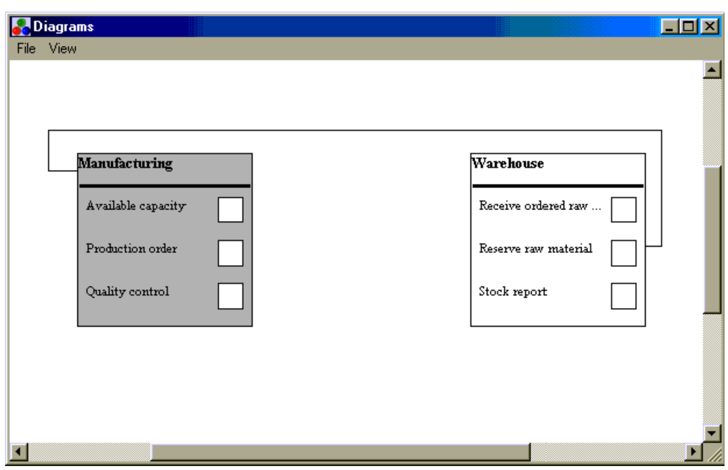

Figure 10. View Showing the Services Used from Other Entities

This type of diagram can frequently help in performing some organization level diagnostics. For instance, the load on any specific *responsibility* (the number of clients as conveyed by the number of links that diverge from it) becomes clear. If too big, it may suggest relocating the *responsibility* to another *entity* or splitting it up into simpler ones.

Since *organizational entities* and the *responsibilities* they are accounted for are not technically minded modeling elements, but rather concepts easily related to everyday work in an organization, this simplicity and modularity can be leveraged by enabling people directly related to any *organizational entity* to continuously analyze its role in the organization, as reflected in the CRC cards. Each *organizational entity* can continuously assess whether the relative values of its *responsibilities* have changed, if their clients remain the same, whether it should deploy new *responsibilities*, whether it feels the need for some "service" from other *entities*, or whether the "internal" systems that support its *responsibilities* remain adequate to the load and importance of the *responsibility* they serve.

Once again, the basic philosophy of the approach, the use of simple field instruments, and the support of the software tool work together to make continuous development of the information system a light process.

The use of software packages to support theoretical approaches or techniques that are continuous in nature is not new in this field. An example is the implementation of the Balanced Score Card concept—a kind of dashboard for strategic management (Kaplan and Norton 1992, 1993, 1996). We also take into account situations where the package used for initial systems design can be left in the organization to facilitate continuous analysis and reshaping of the information system according to changes in organization strategy and environmental conditions. *Organizational entities* and *responsibilities* can be maintained just like "Quality Manuals" are in ISO 9000 certified organizations, the difference being that, in this case, information system support is readily supplied in the form of a tool for users and managers. It should be noted, however, that this does not mean that we share the belief in some trends of the "post-methodology era" that see ISD essentially as a technological process whose problems can be overcome by resorting to newer and better CASE products (see Avison and Fitzgerald 1999). We do believe, however, that tools can support a solid ISD process, just like they do in other businesses such as aeronautics, where similar tools are used in the design and maintenance of their complex products.

The modularity of the approach and the clear links to interested parties are natural facilitators to the establishment of practices that keep the information up to date (just like "Quality Manuals" are). Nevertheless, additional incentives are possible. For instance, the information stored in the database of the design tool can be leveraged to change radically the way in which some searches are performed on enterprise intranets. We are currently exploring this avenue.

6. CONCLUSION

The fast changing nature of competition and collaboration in business environments is transforming business model innovation into a central concern of present day organizations. This is converting information systems development into an endless process, where senior managers and the generality of the users across the organization become deeply involved. The changing nature of this environment is strained even further by a range of additional challenges, such as the growing complexity of the applications, the shorter windows of opportunity, the availability of enterprise software applications that must be coherently integrated, the preservation of the legacy resulting from quickly aging applications, and the need to leverage the continuous investment in information systems. All those challenges call for new ISD paradigms that can grant *continuous development*, *wholeness*, *integration* of disparate and complex parts, and *friendliness* toward the various actors involved.

We have presented our view of how these goals can be accomplished. Proceeding from a brief introduction of an ISD approach that has been developed within an action-research program, we explained how its core concepts, field instruments, and supporting software tool work together to meet the new challenges.

Meanwhile we are exploring new avenues, such as the leveraging of the information collected during modeling to change radically the way in which some searches are performed on enterprise intranets. We are also looking into the identification of key business indicators related to each *responsibility*, which may generate their own information system needs. The indicators will enable performance analysis, such as the satisfaction of critical success factors (Rockart 1979) and links to higher-level objectives that, in turn, influence information system support for the *responsibility*. Finally, we are considering the original project management implications of the utilization of the approach in the field, with particular emphasis on quality, risk management, and various kinds of metrics.

7. ACKNOWLEDGMENTS

This work is partially supported by POSI- Programa Operacional Sociedade de Informação of Portuguese Fundação para a Ciência e Tecnologia and European Union FEDER, and by PRODEP under grant 5.2-nº1/96.

8. REFERENCES

Asbrand, D. "Legacy Systems: The Keepers and the Goners," *ERP Hub–Interface Development*, 1999 (http://www.erphub.com/interface_p1.html).

Avison, D. E., and Fitzgerald, G. *Information Systems Development: Methodologies, Techniques and Tools*, London: McGraw-Hill, 1997.

Avison, D. E., and Fitzgerald, G. "Information Systems Development," in *Rethinking Management Information Systems*, W. L. Currie and B. Galliers (edsl), New York: Oxford University Press, 1999.

Avison, D., Lau, F., Myers, M., and Nielsen, P. "Action Research," *Communications of the ACM* (42:1), 1999, pp. 94-97.

Baskerville, R. L. "Investigating Information Systems with Action Research," *Communications of the AIS* (2:19), 1999.

Baskerville, R. L., and Wood-Harper, A. T. "A Critical Perspective on Action Research as a Method for Information Systems Research," *Journal of Information Technology* (3:11), 1996, pp. 235-246.

Beck, K., and Cunningham, W. "A Laboratory For Teaching Object-Oriented Thinking," in *Proceedings of the Conference on Object-Oriented Programming: Systems, Languages, and Applications (OOPSLA'89),* N. K. Meyrowitz (ed.), New Orleans, LA, October 1-6, 1989.

Booker, E. "ASP Model Gains Legitimacy," *Internet Week*, 1999 (http://www.internetwk.com/story/INW19991028S0012).

Constantine, L. "The Structured Design Approach," *Byte*, 1999, pp. 232-233.

Cunha, P. R., and Figueiredo, A. D. "Information Systems Design Under a Different Light," in *Proceedings of the Sixth Americas Conference on Information Systems*, H. M. Chung (ed.), Long Beach, CA, August 10-13, 2000.

Cunningham, W. "How Do Teams Shape Objects? How Do Objects Shape Teams?" in *Proceedings of the Ninth Annual Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA'94),* Portland, OR, October 23-27, 1994.

Davenport, T. H. *Information Ecology: Mastering the Information and Knowledge Environment*, New York: Oxford University Press, 1997.

De Grace, P., and Stahl, L. *The Olduvai Imperative: CASE and the State of Software Engineering Practice*, Englewood Cliffs, NJ: Prentice Hall, 1993.

Dick, B. "Rigour and Relevance in Action Research," 1997 (http://www.scu.edu.au/schools/gcm/ar/arp/rigour.html).

Dick, B., and Swepson, P. "Appropriate Validity and Its Attainment Within Action Research," 1994 (http://www.scu.edu.au/schools/sawd/arr/sofsys2.html).

Edwards, C., Ward, J., and Bytheway, A. *The Essence of Information Systems*, London: Prentice-Hall, 1991.

Fitzgerald, B. "Whiter Systems Development: Time to Move the Lamppost," in *Proceedings of the Second Conference on Information Systems Methodologies*, Swindon, England: BCS Publications, 1994.

Fitzgerald, B. "Systems Development Methodologies: The Problem of the Tenses," *Information Technology & People* (13:2), 2000, pp. 13-22.

Kalakotta, R., and Robinson, M. *e-Business: Roadmap for Success*, Reading, MA: Addison Wesley Longman, Inc., 1999.

Kaplan, R. S., and Norton, D. P. "The Balanced Scorecard: Measures that Drive Performance," *Harvard Business Review on Measuring Corporate Performance*, 1992, pp. 123-145.

Kaplan, R. S., and Norton, D. P. "Putting the Balanced Scorecard to Work," *Harvard Business Review on Measuring Corporate Performance,* 1993, pp. 147-181.

Kaplan, R. S., and Norton, D. P. "Using the Balanced Scorecard as a Strategic Management System," *Harvard Business Review on Measuring Corporate Performance,* 1996, pp. 183-211.

Keegan, P. "The Death of Software?" *Upside Today*, 1999 (http://www.upside.com/texis/mvm/story?id=37d936dc0).

Kruchten, P. *The Rational Unified Process: An Introduction*, Reading, MA: Addison Wesley Longman, Inc., 2000.

Lau, F. "Toward a Framework for Action Research in Information Systems Studies," *Information Technology & People* (12:2), 1999, pp. 148-175.

Mateyaschuk, J. "Leave the Apps To Us! ASPs Offer Benefits Through Economies of Scale," *InformationWeek*, 1999 (http://www.techweb.com/directlink.cgi?IWK19991011S0032).

McFarlan, F. W. "Information Technology Changes the Way You Compete," *Harvard Business Review*, 1984, pp. 98-103.

Mitchell, I. "The Card Deal," *OMO – Object Magazine Online*, 1997.

Morgan, G. *Images of Organization*, Thousand Oaks, CA: Sage Publications, 1997.

Nickell, J. A. "ASPs: Software Superstores of the Future?" *webreview.com*, 1999 (http://webreview.com/ pub/1999/09/03/feature/index2.html).

Rockart, J. F. "Chief Executives Define Their Own Data Needs," *Harvard Business Review* (57:2), 1979.

Seymour, J. "Send Out for Software," *PC Magazine*, 1999 (http://www.zdnet.com/pcmag/stories/reviews/0,6755,2344646,00.html).

Siau, K., and Rossi, M. "Evaluation of Information Modeling Methods: A Review," in *Proceedings of the Thirty-First Hawaii International Conference on System Sciences,* Los Alamitos, CA: IEEE Computer Society Press, 1998.

Stephens, E. "Ready, Set, Go! (Again)," *Software Development*, 1998, pp. 38-44.

Tapscott, D. "New Business Models and the Creation of Wealth," forward to *e-Business: Roadmap for Success*, R. Kalakotta and M. Robinson (eds.), Reading, MA: Addison Wesley Longman, Inc., 1999.

Taylor, D. A. *Business Engineering with Object Technology*, New York: John Wiley & Sons, Inc., 1995.

Veryard, J. "What Are Methodologies Good For?" *Data Processing*, July/August 1985, pp. 9-12.

Ward, J., and Peppard, J. "Reconciling the IT/Business Relationship: A Troubled Marriage in Need of Guidance," *Journal of Strategic Information Systems* (5:1), 1996, pp. 37-65.

Wilkinson, N. M. *Using CRC Cards: An Informal Approach to Object-Oriented Development,* New York: Cambridge University Press, SIGS Books, 1995.

Wirfs-Brock, R., and Wilkerson, B. "Object-Oriented Design: A Responsibility-Driven Approach," in *Proceedings of the Conference on Object-Oriented Programming: Systems, Languages, and Applications (OOPSLA'89),* N. K. Meyrowitz (ed.), New Orleans, LA, October 1-6, 1989.

Wirfs-Brock, R., Wilkerson, B., and Wiener, L. *Designing Object-Oriented Software*, Englewood Cliffs, NJ: Prentice Hall, 1990.

About the Authors

Antonio Dias de Figueiredo is a professor of Informatics Engineering at the University of Coimbra, Portugal, since 1984. He obtained his Ph.D. in Computer Science from the University of Manchester, U.K., in 1976. His current research interests center on business information systems and ICT in learning and education. He has been vice president for Western Europe of the

Intergovernmental Informatics Programme of UNESCO and he is the current chairman of the European initiative PROMETEUS. He has participated in various European projects and acted on various occasions as a consultant to the European Commission. He was awarded an Honoris Causa by the Portuguese Open University in 1997 and the Sigillum Magnum of the University of Bologna, Italy, in 2000. He is a member of the editorial board of *Information & Management*, an Elsevier North-Holland journal, and of the *Portuguese Management Journal*. Antonio can be reached by e-mail at adf@dei.uc.pt.

Paulo Rupino da Cunha is an assistant professor of Informatics Engineering at the University of Coimbra, Portugal, since 1993. He graduated in Electrical Engineering (Informatics), from the University of Coimbra, Portugal, in 1992. He obtained his M.Sc. in Informatics Engineering from the University of Coimbra, Portugal, in 1997. He is presently waiting for the public examination of his Ph.D. thesis. His current research interests are information systems development and research methods themselves. He is a deputy director of an innovation and technology transfer laboratory (IPNlis). He has also been elected the Coordinator of the Informatics Engineering Chapter, for the center region of Portugal, of the Portuguese Engineering Association. Paulo can be reached by e-mail at rupino@dei.uc.pt.