The Focus of Requirements Engineering in Workflow Application Development

Niko Kleiner

Dept. for Programming Methodology, Faculty for Computer Science
89069 University of Ulm, Germany
nikolaus.kleiner@informatik.uni-ulm.de

Abstract. The paper discusses the focus of Requirements Engineering in workflow application development. Its core thesis is that the workflow application and the new business process it supports emerge in parallel. As a conclusion, putting more effort into early development steps – like initial business process modeling and analysis – does not necessarily contribute to the project's overall success. Rather, Requirements Engineering should shift its focus from specification of system requirements to its enactment in operational use. Assuming that one of the main reasons for introducing a workflow application is to help to drive business processes towards their goals, detailled information about *how* users enact the workflow application and *where* and *why* they deviate from the intended working style becomes more important.

1 Introduction

The main focus of traditional Requirements Engineering is the (continuous) specification of the requirements of the system under consideration. This paper argues that the focus should be shifted from specification to operational use. Section 2 reports about a project history from automotive industry that motivated this research. Section 3 and 4 summarize our conclusions from recent results in organizational science. Finally, section 5 discusses briefly the impact of these findings on the Requirements Engineering job in workflow application development.

The paper uses the standard terminology suggested by the Workflow Management Coalition [1]. Further, for this paper I define a *workflow application* to be any software system that implements a workflow. The technology used for this implementation is of no concern. A *workflow management system* is a workflow application that stores executable process definition and uses an engine to execute it.

2 A Project History from Automotive Industry

This section describes three major iterations of a workflow development project in automotive industry lasting over seven years. All iterations yielded suboptimal solutions. The fourth trial is now in progress. After each iteration, those

statements of the project participants are described, which they, at that point in time, *assumed* to be the causes for the failure.

The data was collected by interviewing project participants and documentation reviews. The data of the last two years is also based on personal experience.

2.1 Project Site

In the process of car development, digital prototypes play an increasing role. CAD models of single parts or subunits of a car are arranged on a workstation in the way they will be arranged in the physical counterpart. They are checked for collisions. The business process of grouping, checking and rating parts and subunits is called *packaging*. Within this context, our task was to design an appropriate workflow for a given business process and to integrate the workflow application into the existing IT and business process infrastructure (engineering data management world and engineering and production processes).

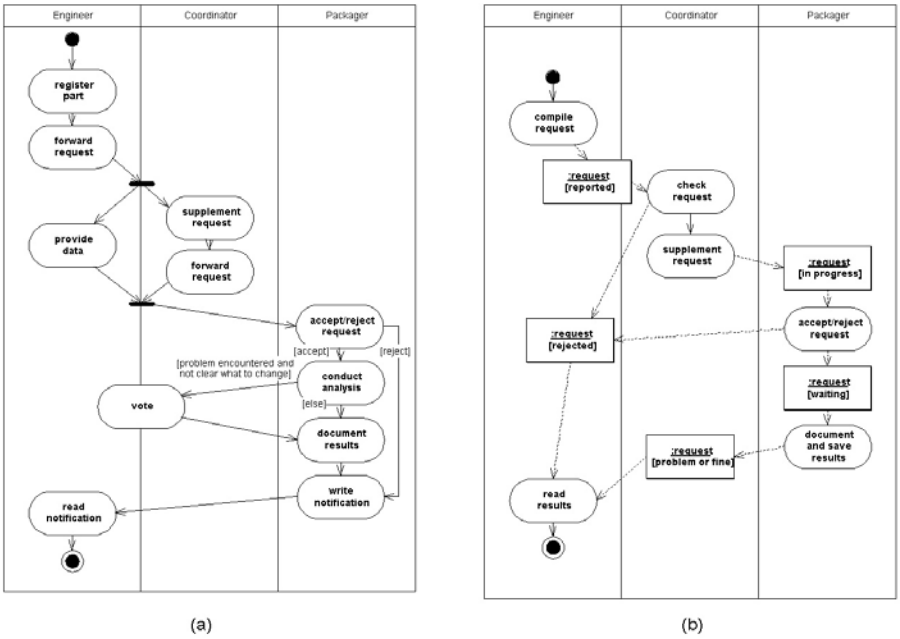

Fig. 1. Packaging Business Process and Workflow

The initial packaging business process is outlined in figure 1(a). An engineer constructs a part and registers it for packaging. He forwards the request to a coordination team. While this team checks the request for correctness and fills out additional data, the engineer provides the data necessary to conduct the

check. Then, the request is forwarded to the packager. He accepts or rejects the request and analyzes if the part fits into its future environment. If there is a problem, it may be that together with other engineers he needs to decide, if the part itself or a part of the environment needs to be changed (activity 'vote'). Then, he documents the result of his analysis and writes a short notification to the engineer that did the request. This engineer reads the notification and proceeds dependent on the packaging result.

2.2 Three Major Iterations

The first trial to design a workflow application followed the ARIS method. Consultants analyzed and modelled the existing business processes. These models were then further refined to an executable process definition specified within the language of the workflow management tool selected for implementation. The first trial proceeded very systematically and in a top-down manner. Even though much effort was put into business process analysis and modelling, the resulting application was rejected by the users and never came into productive use. During this time, the project members thought that the failure was due to the choice of a workflow management system.

Consequently, the second iteration chose traditional implementation techniques (an intranet solution based on PHP and ORACLE). Implementation was assigned to an external supplier. The intranet solution was not fully integrated into the existing Engineering/Production Data Management (EDM/PDM) infrastructure. Thus, engineers had to search the data in the EDM system and feed the packaging application. Additional problems arose, since the existing EDM system was replaced in a smooth process lasting several months. The packaging application had to provide several features to be able to deal with both system types. Again, the iteration failed. Two reasons were mentioned during this time: (1) Due to the high change ratio, the resulting application was of bad quality; (2) further, the lack of integration with the EDM/PDM infrastructure led to data inconsistencies.

Trial three returned to workflow management technology. The workflow management system was an add-on component of the existing EDM system and thus the problem of data inconsistencies was easily solved. Different workflow designs were tried out. First, a simple workflow design using three states to track process progress was implemented. It turned out, that many packaging requests accumulated unprocessed somewhere in the process. To be able to better control the packaging request's progress, the second design was based on eight states. Two major reasons caused this design to fail again. First, users had problems interpreting the states in a consistent way and second, the many states implied many transactions and many transactions led to unacceptable performance. The third design used five states (close to the workflow outlined in figure 1(b)). At first, this seemed to be a feasible solution, but during use it turned out that people tended to interpret an important process state in different ways. Additionally, the problem of dead packaging requests was still not solved. Finally, a study about why users interpreted this certain state in so different ways and why

packaging notifications tended to accumulate yielded the insight that the problems are due to a wrong assumption concerning the synchronization between the packaging and another, related business process.

This misconception has been resolved in the meantime. Iteration four is still in progress.

3 Recent Results in Organizational Science

Assuming that one of the main reasons for introducing a workflow application is to help to drive business processes towards their goals brings up the question how an organization can be influenced with IT. Organizational science already uses a well accepted theory from social science [2] to analyze organizational processes [3]. In the early nineties, Robey and Orlikowski used the same theory to start the development of a model that explains the role of (information) technology in organizations. The life-cycle model presented in figure 2 is a conclusion of Orlikowski's more recent findings [4].

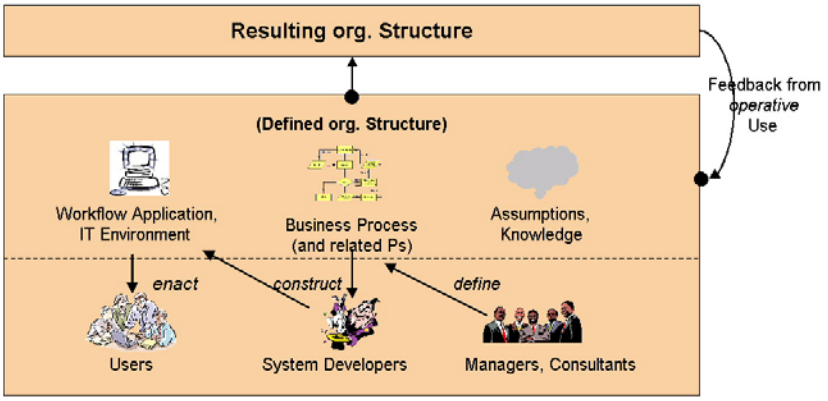

Fig. 2. A Life-Cycle Model for Workflow Applications

The life-cycle model from figure 2 integrates organizational and technical as well as human aspects into one framework: Managers and/or consultants define a new way of doing business. System developers[1] mostly use those descriptions as a starting point for the design of the workflow application. After testing and deploying the new application, users are confronted with it. They enact the new application in their daily work. The new working style – with the new application – results in a new organizational structure. For the purposes of this paper, the term 'resulting organizational structure' comprises all work do be done to achieve the intended business goals operationalized by the business process under consideration.

[1] The term 'System Developers' here refers to the whole system development team

4 Insights from Organizational Science

Uncertainty in Design Decisions

> *Design decisions in the development process are influenced by personal knowledge and assumptions and are therefore usually suboptimal.*

The business process is usually designed from a macro perspective with business objectives in mind. The designers involved mostly do not know the problems of work practice (micro view), and hence such business process designs often conflict with the latter. Further, system developers have only partial knowledge of both the organizational view and the work practice and mainly base their application designs on a technical background. An example for the problem of choosing an adequate workflow design is the third iteration where designs based on three, eight and five states were tried out, all supporting the same business process. Requirements Engineering already contributes a lot to the problem of choosing an adequate design but usually takes the business process for granted.

Emergent Work Practice

> *The consequences of system deployment and the eventual enactment in daily work practice can usually not be foreseen (in its entirety). Rather, the enactment emerges dependent on context and history. One of the goals of introducing the workflow application is to help to drive this emerging organizational structure towards the business objectives.*

A number of researchers already reported that the full consequences of system deployment lay beyond managerial power [5]. In addition, the Orlikowski theory says that the eventual enactment of the application *emerges*, dependent on context and history. Thus, the resulting organizational structure also emerges. The model from figure 2 comprises three major context defining factors[2]: (1) The workflow application under consideration and the related IT infrastructure (supporting IT resources), (2) the business process under consideration and related business processes (norms to be followed), (3) knowledge and assumptions of the end users (knowledge about system usage, assumptions about business objectives to be achieved etc). Work practice is concerned with accomplishing daily tasks (micro view) and neglects – and thus often conflicts – with the higher level business objectives (macro view).

These theoretical insights help to explain the project history. The causes for failure mentioned by the project participants were reasons, of course, but just some among others (and with respect to the theoretical insights less important). After iteration one and two they mainly blamed technology for the failure (the use of a workflow management system, missing integration into existing IT infrastructure). They were not aware, that they had to go through a natural learning process, too, to find out a workflow design that integrates smoothly into the rest of the organization.

[2] The black dot at the source of the arrow to 'Resulting org. Structure' in figure 2 is supposed to indicate the fact that it is a result emerging from all three factors

5 Impact on Requirements Engineering?

The key insight is that the business process as well as the work practice are suboptimal, each neglecting the other in the way mentioned in the preceeding section. Thus, to steer towards an optimal organizational structure[3] they have to converge and learn from each other. Thus,

> *If the requirements engineering job is to help to drive the software system towards its real world goals and one of them is to help to drive the business process it supports towards its business goals, isn't then the requirements engineering job to support this convergence?*

If so, the traditional tasks of Requirements Engineering do not become obsolete, since they still help to improve the quality of the implementation with respect to the current knowledge about the business process. But gaining knowledge about the information of *how* users eventually enact the application and *where* and *why* their working style deviates from the intended one becomes even more important[4]. This information then serves as input for the next development cycle. The black dot in figure 2 at the target of the arrow originating at 'Resulting org. Structure' is supposed to indicate the fact that the results of this analysis can require changes in all three context defining factors and not solely in a change in application requirements. But this would extend the traditional requirements engineering job. Should this really become a job of an requirements engineer? If not, who else should do it?

References

1. Fischer (editor), L.: The Workflow Handbook. Future Strategies Inc., Lighthouse Point, FL, USA (2002)
2. Giddens, A.: The Constitution of Society: Outline of the Theory of Structure. University of California Press, Berkley, CA (1984)
3. Manning, P.K.: Symbolic Communication. MIT Press, Cambridge, MA (1989)
4. Orlikowski, W.J.: Using technology and constituting structures: A practice lens for studying technology in organizations. Organization Science **11** (2000) 404–428
5. Powell, W.: Review essay: Explaining technological change. American Journal of Sociology **93** (1987) 185–197

[3] This optimum usually changes, too, since the organization's environment changes over time

[4] Remark: This shift in focus can further be exemplified by iteration two and three: Iteration two demonstrates that the enactment of the application depends heavily on the related IT – e.g. data inconsistencies. Iteration three shows how related business processes can have an impact on final system usage – requests tended to accumulate. Both problems would not have been detected in a traditional test. The paper does not argue that there are no existing techniques to deal with studying usage (cf. task analysis, scenario-based requirements engineering) but that this problem should be the focus in workflow application development and receive more attention in the sense outlined in this paper.