Association for Information Systems
AIS Electronic Library (AISeL)

ECIS 2013 Research in Progress | ECIS 2013 Proceedings

7-1-2013

iARIS - Supporting Enterprise Transformation Using An Iterative ISD Method

Matthias Steinhorst
University of Münster - ERCIS, Münster, NRW, Germany, matthias.steinhorst@ercis.uni-muenster.de

Follow this and additional works at: http://aisel.aisnet.org/ecis2013_rip

Recommended Citation
Steinhorst, Matthias, "iARIS - Supporting Enterprise Transformation Using An Iterative ISD Method" (2013). *ECIS 2013 Research in Progress*. 18.
http://aisel.aisnet.org/ecis2013_rip/18

This material is brought to you by the ECIS 2013 Proceedings at AIS Electronic Library (AISeL). It has been accepted for inclusion in ECIS 2013 Research in Progress by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

IARIS – SUPPORTING ENTERPRISE TRANSFORMATION USING AN ITERATIVE ISD METHOD

Steinhorst, Matthias, University of Münster - ERCIS, Leonardo Campus 3, 48149 Münster, Germany, matthias.steinhorst@ercis.uni-muenster.de

Abstract

Enterprise architecture frameworks can be used to guide the development process of an information system (IS). While these frameworks cover a comprehensive approach to IS development, most of them essentially proclaim a linear, stepwise development process. In the field of computer science, such linear, waterfall-based approaches have been replaced by so-called agile methods proclaiming an iterative development process. As these approaches are naturally only concerned with software engineering, they do not consider the business processes the software is supposed to support or the organizational structures the software is supposed to be employed in. The purpose of this paper is to propose an iterative information systems development method called iARIS. It combines the "best of both worlds" in joining the comprehensive approach to IS development of common EA frameworks with the iterative software engineering paradigm. The paper contributes to the research on enterprise transformation in proposing a method tailored to support IS development in a rapidly changing environment. The paper furthermore reports on an on-going IS development project providing insights into how iARIS can be applied. The paper thus contributes to the research on enterprise transformation in suggesting how an IS development method can be used in practice.

Keywords: Agile Information Systems Development, iARIS, compliance management, pattern matching.

1 The need for iterative IS development methods

Today, modern enterprises are subject to continuous transformations. New technologies, new markets, new customer requirements, and new regulations require enterprises to constantly change to meet the challenges of a rapidly fluctuating environment. These dynamics require the information systems supporting the business functions of an enterprise to change accordingly. Developing an information system is a time-consuming, error-prone, and high-risk activity (Lyytinen and Robey, 1999). The need to constantly adapt to changing environments additionally impedes the development process of an IS. Supplementary requirements need to be considered at any stage in the development process and errors in the information system's conceptual specification need to be identified as early as possible.

Recently, enterprise architecture (EA) management has been proposed as a means to guide "the development of the enterprise as a whole and the development of (…) [its] IT portfolio in particular" (Op't Land and Proper, 2007, p. 1956). Enterprise architecture management can be construed as a means to better align business and its supporting IT systems (Aier, Riege, and Winter, 2008, p. 299 or Saat et al., 2011). To do so, enterprise architecture frameworks provide a model based description of an enterprise's business processes, its IT systems, organizational structure, etc. Popular frameworks include ARIS (Scheer, 2002), Zachman (Zachman, 1987), and many others. Other than for enterprise modelling, some EA frameworks can also be used to structure and guide the development process of an information system (Scheer, 2002, p. 5). Such frameworks typically propose a set of views and layers to structure the development process of an IS. Each view describes a particular aspect of the IS like the business processes it is supposed to support or the data it needs to manage. Each layer contains conceptual models designed to incrementally translate the corporate reality the IS is supposed to capture into a working implementation. The idea of many of these frameworks is that each layer of each view is run through sequentially to develop a comprehensive model of the IS that is then implemented. The underlying definition of an information system is that of a socio-technical system that consists of three object types, "namely people (i.e., human task bearers), information and communications technology (i.e., technical task bearers), and organizational concepts (i.e., functions, structures, processes), and the interrelationships between them" (Österle et al., 2011, p. 8).

While the frameworks that are based on this definition cover a comprehensive approach to IS development, most of them essentially proclaim a linear, stepwise development process. Consider the example of ARIS that divides the development process into three phases (or layers). Each phase is executed once and provides an output that is further worked on in the following phase. In doing so, business requirements are incrementally transformed into a working implementation. ARIS and similar frameworks can thus be considered the IS equivalent to the waterfall approach of software engineering (Plümicke, 2004).

The waterfall approach, however, suffers from significant drawbacks. It assumes that all software requirements are a priori known and can thus be included in the conceptual specification. This, however, is seldom the case, as customer requirements constantly change (Abrahamsson et al., 2002, p. 71). To consider these requirements, an existing prototype has to be modified or altogether developed anew. This causes significant extra work and the application of additional resources. Furthermore, developers might only realize during implementation that particular aspects of the conceptual specification cannot be realized as envisioned. If noticed too late, this again causes significant rework. Against the backdrop of enterprise transformation, waterfall-based approaches are consequently not suitable to handle the complexity of an IS development project.

In the domain of computer science, waterfall-based software development methods have since been replaced by so-called agile methods proclaiming an iterative development process that starts with a small subset of requirements that are implemented first (e.g. DSDM or ASD (Abrahamsson et al., 2002)). The resulting prototype is then iteratively augmented to include the remaining requirements. In doing so, additional customer requirements can flexibly be included at any stage and errors in the

software's conceptual specification can be detected early in its development process. Stemming from the domain of computer science, these methods are naturally only concerned with software engineering. Consequently, they do not consider the business processes the software is supposed to support or the organizational structures the software is supposed to be employed in. Hence, these methods are only concerned with software development, not *information systems* development.

This paper introduces an iterative IS development method called iARIS. It is based on ARIS and incorporates the idea of an iterative, small step development process. iARIS combines the "best of both worlds" in joining the comprehensive approach to IS development of common EA frameworks with the iterative software engineering paradigm. The paper contributes to the research on enterprise transformation in proposing a method tailored to support IS development in a rapidly changing environment. The paper furthermore reports on an on-going IS development project providing insights into how iARIS can be applied. The paper contributes to the research on enterprise transformation in suggesting how an IS development method can be used in practice (Nafiz et al., 2004).

Following a design science research process (Peffers et al., 2004) the remainder of this paper is structured as follows. In Section 2, related work is presented to frame the objectives of our solution. To design and develop the solution, the iARIS method is introduced in Section 3 by explaining its underlying framework, its principles, and its basic modelling techniques. To demonstrate and preliminarily evaluate the solution, Section 4 provides insights into an on-going iARIS-based IS development project that is concerned with the design and construction of a process compliance management system. The paper closes with a summary and an outlook to future research.

2 Related work

In the domain of information systems, Plömicke (2004) proposes the ARIS Unified Process, a combination of the original ARIS framework and the Rational Unified Process (Abrahamsson et al., 2002). Plömicke adopts the three layers of the ARIS framework and combines them with the five development phases known from RUP. In each phase all layers are run through with varying intensity. In the construction phase for instance the focus is on implementing the solution, whereas in the early inception phase the focus is on defining requirements. In doing so, the strictly linear development method of ARIS is mitigated towards a more iterative approach. As opposed to that, in iARIS all layers are of equal importance in all views and must be run through in one development iteration. Keller and Teufel (1998) propose an iterative approach to customizing and deploying standard ERP software. The approach is also based on ARIS. Its focus is not on developing an information system, but configuring an existing (commercial) solution.

As opposed to linear frameworks like ARIS or Zachman, TOGAF (Open Group, 2013) proclaims an iterative development process for enterprise architectures. This process contains a set of eight phases during which four architectures are iteratively developed. Conceptually, TOGAF is similar to ARIS in the sense that what TOGAF refers to as a phase is called a layer in ARIS. What TOGAF refers to as an architecture is called a view in ARIS. However, we chose to base our method on ARIS, because ARIS was explicitly designed to be applicable for IS development (Scheer, 2002, p. 5), whereas TOGAF is designed to primarily be an EA framework. Although TOGAF could probably be augmented accordingly as well, we chose ARIS as the basis of our work because of its explicit purpose of being both an EA framework and an IS development method. In addition, we intend iARIS to be applicable in the context of *enterprise transformation*, in which quick development iterations are necessary to respond to rapidly changing environments. TOGAF has been criticized for being overly complex (Keller, 2012), which might impede its applicability in the context of enterprise transformation.

In the domain of computer science, many software development methods have been proposed proclaiming an iterative approach to software engineering. Among the most prominent are the Dynamic Systems Development Method (DSDM) and Adaptive Software Development (ASD) (Abrahamsson et al., 2002) to name only a few. As mentioned above, these methods focus on

developing software. The focus of developing an information system is wider than that encompassing the definition of organizational structures and business processes the software is supposed to support. The purpose of the paper at hand is thus to propose and apply an iterative IS development method.

3 iARIS

3.1 Framework

As specified in the original ARIS method, the iARIS framework disaggregates the development process into views and thereto orthogonal layers. It contains the four views "data", "function", "process", and "organization", as well as the three layers "requirements definition", "design specification", and "implementation" of the original ARIS framework (cf. Figure 1). Each view captures a subsystem of the entire information system. In each view, a particular solution is developed (cf. the icons within the view circles in Figure 1; the icons in the overall circle represent the entire information system). The data view produces a database capturing all required data. The organization view produces organizational structures required for running the IS. The function view produces the application software capturing the functional requirements of the IS. The process view integrates all other views by defining processes that describe the sequence, in which organizational units execute functions that produce data (Scheer, 2002, p. 36). The process view thus defines dependencies between all other model views. Particular dependencies have to be defined for each individual information system development project.

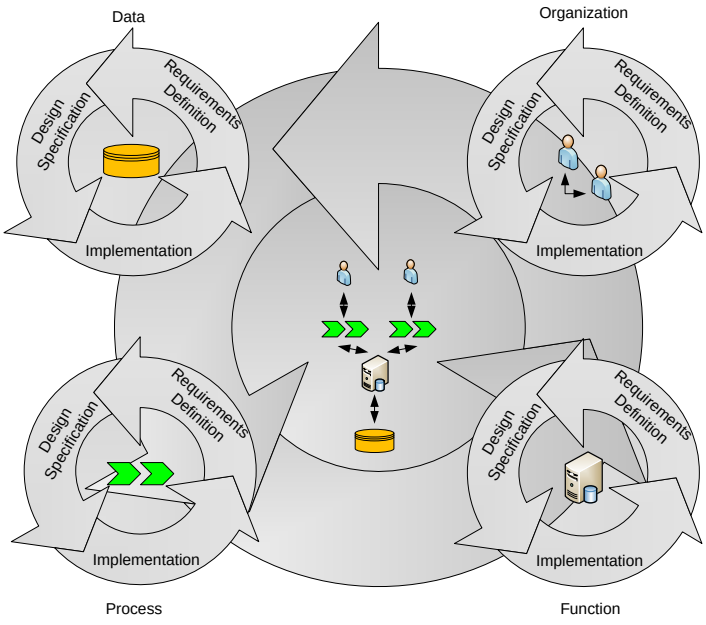

Figure 1: The iARIS framework

Each layer describes the respective view on a particular level of abstraction starting from a purely conceptual requirements definition that is incrementally translated to a working implementation. The implementation layer of a given view also includes testing the developed prototype which triggers an additional development iteration for that view. This is necessary, as during implementation and testing additional requirements and errors in the design specification might turn up that were not conceived of before. As specified below, changes to a particular view need to be reflected with respect to their impact on other views. Each inner-view iteration therefore triggers additional iterations in all other views represented by the circular arrows in the framework.

3.2 Principles

The basic principles of iARIS guide the design and development process of an IS. Many of the iARIS principles are similar to those of other agile software development methods (Abrahamsson et al., 2002). The iARIS principles furthermore cover all views and layers of the original ARIS framework thus proclaiming a more comprehensive view on IS development than agile software engineering methods. The principles contain a set of roles that need to be represented in an IS development team. These roles are the software developer, the business process analyst, the organizational designer, and the data analyst. The iARIS principles are as follows. Note that the order of the principles is not intended to be an assessment of their relative importance. All principles are equally important.

Collaborate (P1): Software developers need to work together with business process analysts as well as organizational designers and data analysts. The team structure needs to reflect all views of the ARIS architecture, i.e. all roles defined by iARIS need to be represented in the development team to comprehensively cover all aspects of developing an information system.

Develop iteratively (P2): An information system is developed iteratively. All layers of a given view are run through iteratively. In each iteration, the focus is on developing a working prototype that can be tested and improved upon early. Feedback is required from all team members to consider all aspects of the development process.

Think small, then get bigger (P3): A set of core features has to be defined that constitute the main functionality of the IS. This set of core features is implemented and tested early. It is then incrementally augmented to include additional features that should, respectively could be included in the information system.

Think holistically (P4): In one development iteration all views of the framework need to be considered. Changes to one view need to be reflected in terms of the resulting changes to all other views, i.e. changes to one view trigger a development iteration in all other views.

Communicate (P5): Design decisions need to be communicated early and clearly. Communication techniques have to be employed that are understood by all team members. Workshop sessions need to be conducted to communicate progress on the development project.

Plan ahead (P6): To deliver results on time, a development iteration must have a clearly defined timeframe. Project plans must be communicated early to raise awareness of the next milestones.

3.3 Techniques

In many EA frameworks, specific modelling techniques are prescribed in each layer of each view. iARIS on the other hand does not prescribe particular techniques. Rather, the use of existing (well established) modelling and implementation techniques is recommended. The development team is required to define the set of modelling and implementation languages that best fit the intended purpose of the particular IS development project. We follow the argument of Scheer (2002, p. 4) in proposing a language independent IS development method. Scheer argues that language independence guaranties the method to be applicable for any development purpose (2002, p. 4.). Particularly in the context of enterprise transformation, the IS development method has to assure a high level of flexibility in order to allow the development team to adapt to changing requirements. In our opinion, this cannot be achieved by prescribing a fixed set of modelling and implementation languages. Instead, the development team has to choose respective techniques given the particular development purpose. In terms of conceptual modelling languages (e.g., ER models for the data view or BPMN for the process view), it is furthermore advisable to configure existing languages to meet the intended purpose. This can achieved by adding new or deleting obsolete language constructs. To do so, meta-modelling tools having been numerously proposed in the literature need to be used.

4 Application of iARIS

In cooperation with a German IT service provider we are currently developing a design-time compliance management system (CMS) concerned with detecting compliance violations in process models. Such violations translate to particular patterns that need to be found in the overall model graph (Bräuer et al., 2013). We are implementing a pattern matching approach as a plugin for a meta-modelling tool that was available from a previous research project. This tool in combination with the pattern matching approach comprises the CMS. Compliance management requires an enterprise to constantly transform its business processes to comply with changing laws and regulations. Business processes that are not compliant can potentially be very costly for the enterprise. To develop a CMS, it is therefore necessary to apply a development method like iARIS that is designed to be applicable in this context of enterprise transformation and allows for quickly implementing a working prototype that is then iteratively augmented as new requirements arise. The core development team consisted of four people each holding all four iARIS roles (P1). Daily scrum meetings were hold to inform all team members of current developments and plan the next milestones (P5 and P6). The pattern matching approach and its application in the domain of compliance checking are published elsewhere (Bräuer et al., 2013). The purpose of this section is to reflect on its development process against the backdrop of iARIS. We distinguish two development iterations and define prototypical solutions surrounding the introduction of such a CMS at the IT service provider (P2 and P3). We will elaborate on the effect of particular design decisions in one view on the design in all other views (P4). We will elaborate on the modelling and implementation techniques we used.

4.1 First iteration

In terms of *functional* requirements for the pattern matching approach, it recognizes any conceptual model as a graph consisting of the set O of its objects (i.e. graph nodes) and the set R of its relationships (i.e. graph edges). The approach offers set-altering functions that take these two basic sets as input and perform particular operations on them. The approach allows for nesting these functions. The result of one function serves as input for the next, resulting in a tree-like pattern query, which can be run on a process model returning all model subgraphs that comply with it. To conceptually model these requirements we formally specified all functions using set theoretical formulas. The application logic was implemented using C# as a programming language, because the underlying meta-modelling tool is also implemented in this language. In terms of *data* requirements the query is parsed to a single string value that can be efficiently stored and accessed in a relational database. Hence, we avoid storing and accessing complex tree structures. An ER model captures these data requirements. The pattern query is transformed into a string using an open source parser generator. NHibernate is used to map pattern objects to a relational database. In terms of *organizational* requirements, an enterprise-wide role that we call a compliance manager needs to be created that consistently defines pattern queries for all compliance rules relevant to the organization. The compliance manager is responsible for unambiguously defining pattern queries. The compliance manager also needs to manually check all models containing pattern occurrences in order to determine if they indeed represent compliance violations. This role requires in-depth knowledge of the approach as well as of legal and process modelling issues. In terms of *processes*, the use of the CMS induces a model analysis process that consists of the following iterative sequence of activities: model processes, identify relevant regulations for processes, represent regulations as pattern queries, run pattern queries on model base, analyse returned compliance violations, redesign processes and corresponding models.

4.2 Second iteration

A prototypical implementation of the pattern matching approach served to carve out additional *functional* requirements. In particular, some compliance rules require additional information (e.g.

comparison of labels in the 4-eye-principle (Bräuer et al., 2013)) to be included in the matching process. The approach was thus augmented such that some functions also take variables as input, which can be compared to one another. Such a comparison can be expressed in the form of variable equations. During a search run on a particular model all possible variable assignments and all pattern occurrences for which the equations hold are determined. A caching mechanism was included to speed up matching performance. In terms of *data* requirements, these additional features required changing the query parser to also include variables and variable equations. No changes need to be made to the CMS in terms of *organizational* requirements or supported business *processes*.

5 Summary and outlook

This paper introduces an iterative information system development method taking into account not only software engineering related aspects but also organizational and process related issues. The IS development method is prototypically applied in the context of a compliance management system to demonstrate its applicability. Future research will focus on actually introducing this compliance management system to an IT service provider for the financial industry.

References

Abrahamsson, P., Salo, O., Ronkainen, J. (2002). Agile software development methods. Review and analysis. VTT Electronics.

Aier, S.; Riege, C. and Winter, R. (2008). Enterprise Architecture – Literature Overview and Current Practices. In: Business & Information Systems Engineering 50(4), p. 292-304.

Bräuer, S., Delfmann, P., Dietrich, H., Steinhorst, M. (2013). Using a Generic Model Query Approach to Allow for Process Model Compliance Checking – An Algorithmic Perspective. In: Proceedings of the 11th International Conference on Wirtschaftsinformatik. Leipzig, 2013.

Keller, W. (2012). TOGAF 9.1 Quick Start Guide for IT Enterprise Architects. http://www.objectarchitects.biz/TOGAF9/TOGAF_9_1_Quickstart_%28V0_9%29.pdf. [Accessed: 2013-03-14].

Keller, G.; Teufel, T. (1998): R/3 Process Oriented Implementation: Iterative Process Prototyping, Addison-Wesely.

Lyytinen, K and Robey, D (1999). Learning failure in information systems development. Information Systems Journal 9(2), 85-101.

Nafiz, M., Harmsen, F., van Slooten, K., Stegwee, R. (2004). An Agile Information Systems Development Method in Use. Turkish Journal of Electrical Engineering, 12(2), 127-138.

Open Group (2013). TOGAF 9.1 http://pubs.opengroup.org/architecture/togaf9-doc/arch/index.html. [Accessed: 2013-03-14].

Österle, H., Becker, J., Frank, U., Hess, T., Karagiannis, D., Krcmar, H., Loss, P., Mertens, P., Oberweis, A., Sinz, E. (2010). Memorandum on design-oriented information systems research. European Journal of Information Systems, 20 (1), 7-10.

Op't Land, M and Proper, E. (2007). Impact of Principles on Enterprise Engineering. In Proceedings of the 15th European Conference on Information Systems (ECIS 2007), p. 1965, St. Gallen.

Peffers, K., Tuunanen, T., Rothenberger, M., Chatterjee, S. (2008). A design science research methodology for information systems research. Journal of MIS, 24(3), 45–77.

Plümicke, M (2004). ARIS meets RUP. The ARIS Unified Information System Development Process. In: Proceedings of the 2004 Workshop on Business Process Management with EPCs.

Saat, J., Winter, R., Franke, U., Lagerström, R., Ekstedt, M. (2011). Analysis of IT/Business Alignment Situations as a Precondition for the Design and Engineering of Situated IT/Business Alignment Solutions. In Proceedings of the 44th Hawaii International Conference on System Sciences, USA.

Scheer, A.-W. (2002): ARIS – Vom Geschäftsprozess zum Anwendungssystem. Springer, Berlin.

Zachman, J.A. (1987): A Framework for Information Systems Architecture. In: IBM Systems Journal, 6 (3), 277-321.