Domain Repositories as Coordination Support in Collaborative Engineering

Kurt Sandkuhl, Andreas Billig

School of Engineering
Jönköping University
P.O. Box 1026, 55111 Jönköping, Sweden
kurt.sandkuhl@jth.hj.se, andreas.billig@jth.hj.se

.

Abstract: Collaborative engineering involves knowledge-intensive activities involving different specialists in collaboration processes tailored for the engineering domain under consideration. Focus of this paper is a specific aspect of coordination: how to support joint use of artefacts during collaborative engineering based on domain repositories. Starting from selected application examples, requirements towards domain repositories are identified. In this context, the domain repository ODIS offers basic functionality for coordination support by facilitating integrative, partitionable and user adaptable artefact management. Selected CSCW approaches, i.e. a groupware reference architecture and propositions for coordination mechanisms, are used to identify ODIS extensions.

1 Introduction

Collaborative engineering aims at supporting a distributed group of engineers sharing a common collaboration objective in jointly performing an engineering task, like product design, production planning, engineering change management or development of specifications. These engineering tasks are usually knowledge-intensive activities involving different specialists in collaboration processes tailored for the engineering domain under consideration. Coordination of collaboratively performed activities has been subject to research for more than two decades. Selected examples are approaches from CSCW[1] in coordination in workflows [RMRF01], awareness in shared workspaces [DB92] or coordination mechanisms in articulation work [SS96].

This paper will focus on a specific aspect of coordination, namely how to support joint use of artefacts during collaborative engineering based on domain repositories. Research on repositories is motivated by the increasing demand of enterprises to create and manage project entities like (meta-) models, document metadata, and classification ontologies so that the integrative usage of these domain artefacts is supported. Examples

[1] CSCW = Computer Supported Cooperative Work

for domains with a demand for highly developed repositories managing artefacts are software engineering [Sh+04], manufacturing [RG97] or civil engineering [AN99].

Starting from selected application examples, requirements for domain repositories are identified (chapter 2). Main concepts and architecture of the domain repository ODIS are introduced (chapter 3), which was developed for use in engineering disciplines, but not with the explicit intention to support collaborative work. Based on established approaches from CSCW for designing coordination mechanisms, suitability of ODIS for collaborative work and necessary extensions are discussed (chapter 4). A summary and discussion of future work concludes the paper (chapter 5).

2 Industrial Requirements

Three application examples from industry were selected to illustrate typical scenarios with a need for coordination support. After a brief description of these scenarios (2.1), requirements regarding coordination support are identified (2.2).

2.1 Application Examples

The application cases for collaborative engineering situations are taken from product development projects in three different domains. The first example is from mechanical engineering with development of a material specification as core subject. Example 2 is from software engineering. The third example considers development of a hardware/software system in embedded systems engineering.

	Mechanical Engineering	**Software Engineering**	**Embedded Systems Engineering**
Application Example	Establish material specification	Re-design of software systems	Development of dependable system
Roles	Material engineer, test engineer, product designer, production planner	Software architect, user interface specialist, database specialist, project manager	Chief architect, systems engineer, software engineer, quality manager, test engineer
Process	Establish draft, test specification, perform test, final release	According to Unified Process (iterative and incremental)	According to the V-model
Artefacts	Customer requirements, product specification, material specification, test method specification, test result documentation	Programming guidelines, documen-tation, component-models, data flow diagrams, architecture description, re-design strategy and process	Original requirements, hardware-/software-/ mechanical requirement specification, design and architecture speci-fication, simulation and test specification

Table 1: Application examples for collaborative engineering

Table 1 summarizes the characteristics of these three examples with regards to development process, roles involved and artefacts developed. More information on the application examples can be found in [St07], [Sa+98] and [SB06] respectively.

All application examples show a number of different types of artefacts capturing and specifying different aspects of the product under consideration. The examples involve a large group of individuals with different roles and specific competences. Although the development processes are similar to general problem solving processes, they are specific for the application domain.

2.2 Requirements

From an organizational viewpoint, coordination support in a domain repository has to support development processes of systems and components, which are based on a variety of technical and organizational information. The integrative modelling of different perspectives is essential:

- the structure and behavior of the product under development
- vocabularies for used components and techniques
- metadata for project resources
- organizational structures

In product centred disciplines, structural and behavioural specifications of the components are the starting points.

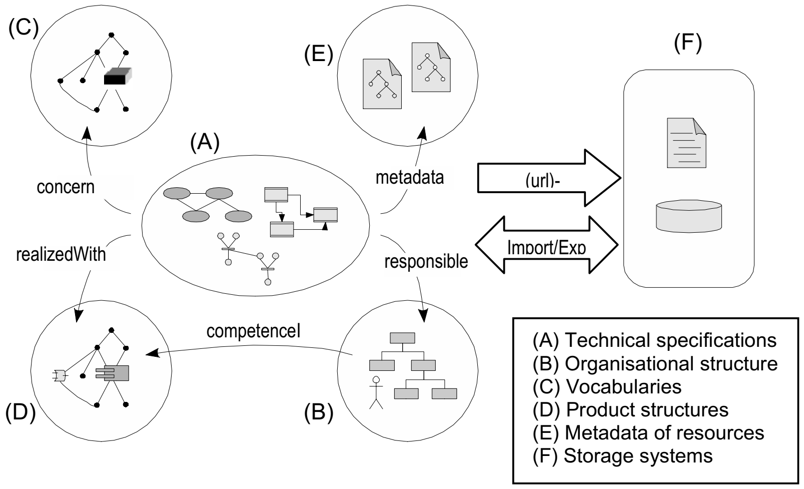

Fig. 1: Artefact groups and relations

According to these perspectives there are several artefact groups which are important in our context (see Fig. 1). The first group (A) includes specifications for components of the product under development. It consists of structure models[2] which determine the structure and coupling of components as well as behaviour models like State Charts and Petri nets which determine the synchronization of components. Group (B) consists of structures, relations and descriptions of organization units and their employees. Apart from these classical elements, groups (C, D) consist of vocabularies and product structures which are used for the classification of artefacts. Another group (E) contains metadata of project resources like documentations, tool files and other data which are not located at the repository but rather stored in persistent systems like databases (F). Beside the variety of artefacts a variety of interrelations between them exists, e.g., functions are related to product components, responsible employees, and realizations with special techniques (properties concerns, responsible and realizedWith) which are known by different employees (property competenceIn). With respect to the management of artefacts and their interrelations the repository system has to focus on the following criteria.

I Partitioning: the whole set of artefacts has to be separated into different units according to the groups introduced above. Furthermore, different contexts for concepts (namespaces) have to be introduced due to the fact that one term has different meanings in different contexts (polysemy/homonymy), e.g. the concept *Controller* can be seen in the context of hardware and in the context of project management.

II Derivations: the system has to offer functionality to derive new information on the base of logical rules. Example: a rule with the predicate *dealsWith*: "A *department deals with a component c if one of its employees is responsible for a hardware or software component which concerns c*".

III Querying: the system has to offer a query language integrating classical and text retrieval constructs as well as rule predicates. This is a prerequisite of finding the right information not only from fully structured but also semi-structured information. Especially terms in technical design documents build a bridge between the different areas like software and hardware design. Furthermore, both schema and instance elements have to be first class citizens due to the need to explore classes depending on their relationships to corresponding vocabularies (see Fig. 1).

IV User adaptation: both navigation through the artefact net and queries have to be adaptable to a user group. For this purpose roles and related views shall be defined. E.g., a detail view of components and metadata and its technical documentation is enabled for the role developer, whereas information concerning project state and organisation is related to the project manager role.

[2] ODIS manages also the corresponding metamodels or, more general, all model hierarchies.

V Transformations: an important requirement is a language which supports transformation of artefacts. This encompasses internal transformation between all artefact groups, as well as import-/export transformations involving external information sources. Related to this point, the standardized exchange of artefacts is required which supports the facile interworking of different organizations without large conversion effort.

3 Domain Repository ODIS

Due to the required flexibility with respect to integration, querying, and exchange of artefacts and vocabularies, concepts from the area of ontologies [UJ99, LAB+99, SM00] and the Semantic Web [W399, W303a, SD02, DSN02] form the basis for ODIS. In particular, the paradigm explained in [SM00], i.e. the management of concepts (like elements of thesauri and glossaries), classes, instances, and their interrelations as first class citizens as well as the expressibility of rules over these elements enables functionality required by criteria II. and III. given in the previous section. In the following, relevant concepts from the Semantic Web area and the related criteria are introduced.

The Resource Description Framework (RDF) is a metadata standard based on XML [W399]. It offers all relevant constructs for building information models. The key construct is the so-called statement

(subject, predicate, object),

where each element of the triple can be prefixed with a namespace. Due to their generality, these base elements are in particular adequate for modelling both fully structured and semi-structured artefacts. On the one hand, RDF supports the partitioning of the whole artefact set with the help of namespaces (see criterion I.) and on the other hand the RDF standardization enables the exchange and use of worldwide available domain artefacts via the Internet, thus meeting the requirements of criterion V. Moreover the RDF(S) framework enables the treatment of schema elements as first class citizens according to criterion III. As an rule and query language over RDF (criterion II) we rely on TRIPLE [SD02, DSN02] due to the facility to state contexts. Logical facts like statements are then valid in certain contexts:

(subject, predicate, object, context).

Contexts enable – together with RDF namespaces – the partitioning of the whole information space of the repository according to criterion I. The stating of a context for the definition of parameterized views, which are imposingly explained in [DSN02], supports criterion IV.

Two important additional concepts are introduced with ODIS: *Modlets and Spot Views*. Modlets are bulks of statements which are mainly used for bundling statements which belong together, e.g. all statements for material specification of mechanical engineering tasks. This is another support for criterion I. Spot views which are connected to *roles* allow for relating views on the artefact set to specific roles which guarantee the fulfilment of criterion IV. In particular during navigation in the artefact net along the relations (properties), we consider it necessary to associate possible views with user roles. E.g., if a user with the role manager is navigating to a certain node within the artefact set (for instance the anchor of organization units) then he is able to invoke a spot view which retrieves all responsible departments for his projects. Finally, the integration of text retrieval (see criterion III) enables the user to retrieve artefacts associated with technical documentation.

4 Coordination Support based on Domain Repository

Coordination in group work is in general concerned with ensuring coherent action of different, often geographically distributed actors. More specific, Malone and Crowston [MC91] define coordination as "the act of managing interdependencies between activities".

The requirements from typical application scenarios discussed in chapter 2 and the implementation of these requirements by ODIS as presented in chapter 3 clearly indicate that a domain repository contributes to supporting collaborative engineering. The main contribution is in the field of artefact management including an integrative, partitionable and user adaptable management of relevant taxonomies, product models and organisational structures.

However, ODIS was initially not developed with distributed groups of engineers in mind, i.e. there are no specific collaboration mechanisms included. The intention of this chapter is to investigate how ODIS could be enhanced for collaborative work with particular focus on coordination support. This investigation will be based on established approaches from CSCW, which will be discussed in the context of ODIS.

4.1 Coordinating Access to Artefacts

Coordinating access to artefacts in order to avoid conflicting changes, mutual exclusion of work on the same parts of an artefact, version management or annotation management has been subject of research activities in CSCW since more than 20 years. Furthermore, numerous approaches discuss integration of access coordination into groupware or CSCW systems.

[SaM00] propose a global reference architecture for groupware applications including different views on the architecture. For our discussion, the logic view and in particular the levels "components and services for CSCW" and "domain-specific extensions of CSCW components" are of particular interest, as they indicate required coordination functionality in general and the specialisation of this functionality for specific application areas. For coordination, the following service description is given:

"(B) Coordination in the sense of scheduling access to shared resources: Reservation, allocation, de-allocation and optimization of resources; usage, mutual exclusion, and recovery after unresolved access conflicts; managing the involved transactions."

ODIS implements these services only partly. Extension towards transaction management would be necessary. The strength of ODIS is an extensive support of defining and managing role-dependent views which follow the global reference architecture. Especially the definitions of views can be stated in a declarative way by means of logical rules. These rules can access the whole artefact set so that an integration of all architectural elements can be defined in dependence on the roles of the system users.

More concrete, it is proposed to enhance ODIS by introducing *coordination maps* in order to support the access coordination at a very early stage. Coordination maps show the partitioning of the artefact set regarding its usage by the different roles of system users.

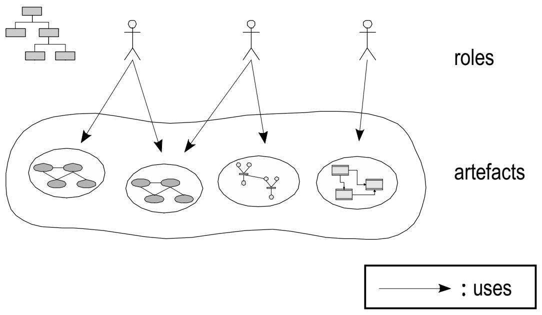

Fig. 2: Coordination map

Fig. 2 shows such a coordination map where three users have potentially access to four artefacts. Furthermore it can be seen that only two roles share an artefact. The derivation of this map is based on a special evaluation function *eval* that operates on the declarative view definitions. *eval* analyses queries within the view and returns those artefacts which can finally be accessed by the view. Let *view(r)* be the set of views related to role *r* then $uses(r) := \bigcup_{v \in view(r)} eval(v)$ defines the set of artefacts to which the roles have access. More precisely, we have to speak of a static coordination map because they are only based the organisational structures represented in the ODIS system. But dynamically generated coordination maps can be realized by supporting dynamic organisational structures. At least dynamic assignment of users to roles can benefit from static coordination maps.

4.2 Coordination Mechanisms

CSCW research considers coordination as one of the main elements to be supported in group work besides communication and collaboration [EGR91]. Nature and complexity of coordination were investigated in numerous research activities and empirical investigations. In this context, the concept of coordination mechanisms was developed and thoroughly investigated by Schmidt and Simone [SS96]. They define: "*A coordination mechanism is a construct consisting of a coordinative protocol (an integrated set of procedures and conventions stipulating the articulation of interdependent distributed activities) on the one hand and on the other hand an artifact (a permanent symbolic construct) in which the protocol is objectified.*" [SS96, pp. 165-166].

Schmidt and Simone develop in their work about coordination mechanisms in CSCW a number of propositions on computational coordination mechanisms. Table 2 provides an overview of these propositions, which are numbered 19 to 26 in [SS96], and how ODIS implements these propositions or should be extended to do so .

No.	Proposition	ODIS
19	Actors should define the protocol of the coordination mechanism Possibility to redefine the protocol for changing organizational requirements	Not provided; would require extension towards work flow management
20	Possibility to local and temporary modification to the behavior to cope with unforeseen contingencies	Not provided; would require extension towards task management
21	Incomplete initial specification of the protocol	Not provided; would require adaptable work flows
22	Definition and specification of the protocol makes sense to actors	Ontology establishes prerequisite for this proposition
23	Relationship between components of mechanisms and field of work	Implemented to large extent by reflecting and integrating the concepts, models and organization structures used
24	Actors control the propagation of changes	Implemented for the work space provided by the application
25	Linked to other coordination mechanisms in the organizational context	Partly implemented: ODIS is designed as central (single) domain repository
26	General notation for constructing computational coordination	Not implemented

Table 2: Proposition of computational coordination mechanism [SS96]

The different actors in collaborative engineering are on the one hand side acting autonomously when solving their specific tasks using own strategies and heuristics. These practices are shared and discussed with the other members of the project, but at the moment of execution there is a high degree of “local”, i.e. task-related, autonomy. On the other hand side, the necessity to align, integrate and schedule the tasks of different actors exists, i.e. the individual and yet interdependent activities must be coordinated. Schmidt and Simone explain that a coordinative protocol is showing “*task interdependencies which actors, for all practical purposes, can rely on to reduce the space of possibilities by identifying a valid and yet limited set of options for coordinative action in any given situation*.” [SS96, p. 174].

In our case, the essential coordinative protocol is the development process for the engineering project in question including best practices to handle unforeseen situations. This coordinative protocol is for some situations defined in detail (like how to perform a material test) and in other situations rather weak. Supporting such an coordinative protocol by a computational coordination mechanism would require an extension of ODIS towards task management or work management. This extension will rely on basic features of ODIS, namely all components which are focussed on process definition and execution. Work management would also establish the basis for adaptable work flows, which is required for meeting proposition 21, i.e. the possibility of an initially incomplete specification of the protocol.

5 Summary and Future Work

Based on a discussion of requirements for coordination supply from different industrial cases, the paper investigates the use of domain repositories in collaborative engineering. The domain repository ODIS offers basic functionality for coordination support by facilitating integrative, partitionable and user adaptable artefact management. However, ODIS was initially not developed to support collaborative engineering and requires enhancements. Selected CSCW approaches, i.e. a groupware reference architecture and propositions for coordination mechanisms, are used to identify four ODIS extensions:

- Coordination maps showing which roles jointly use what artefact sets (see 4.1)
- Version management for supporting allocation, de-allocation and scheduling of resources.
- Workflow management in order to support flexible definition of coordination protocols (see 4.2, proposition 19)
- Task management for being able to flexibly adapt local tasks to user demands (see 4.2, proposition 20)

Future work will investigate the implementation of these extensions. An initial concept for deriving coordination maps with mechanisms already implemented in ODIS has already been discussed in section 4.1. Version management would be more difficult to realise, but could be considered a core functionality of domain repositories.

The implementation of basic workflow management would be eased by the basic concepts used in ODIS. We think that not only the facility of defining process-oriented (meta-)models on the base of so-called information ontologies is an advantage of ODIS but also the flexible definition of execution plan of process-models by means of a declarative language like TRIPLE. However, workflow management seen in combination with task management would require many additional functions, which can be found in standard tools on the market. Integrating a workflow or task management engine into ODIS should be seriously considered as an option.

The main contributions of this paper to the research field are to introduce and discuss the use of integrative artefact management in general and modlets/spot-views in particular in collaborative engineering. Furthermore, it is shown, that the theory of coordination mechanisms can be applied for the purpose to decide on further development of the domain repository. Main limit of research so far is that there are no experimental results from using ODIS in collaborative engineering.

Acknowledgement

Part of this work was financed by the Swedish Knowledge Foundation (KK-Stiftelsen), grant 2003/0241, project SEMCO. The authors would like to thank Christer Thörn for comments and contributions to the paper.

References

[AN99] Abdelsayed, M. and Navon, R. (1999). An information sharing, Internet-based, system for project control. Civil Engineering and Environmental Analysis 16.

[DSN02] Decker, S., Sintek, M., and Nejdl, W.: A logic for reasoning with parameterized views over semi-structured data. Technischer bericht. Universit¨at Hannover. 2002. URL: http://www.kbs.uni-hannover.de/Arbeiten/Publikationen/2002/.

[DB92] Dourish, P. and Bellotti, V. 1992. Awareness and coordination in shared workspaces. In *Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work* (Toronto, Ontario, Canada, November 01 - 04, 1992). CSCW '92. ACM Press, New York,

[EGR91] Ellis, C., Gibbs, S., Rein, G.: Groupware: Some Issues and Experiences. Communications of the ACM, Vol. 34, No. 1 (1991).

[MC91] Malone, T. W. and Crowston, K. Toward an interdisciplinary study of coordination. Center for Coordination Science, MIT. (1991).

[JBR99] Jacobson, I., Booch, G.,Rumbaugh, J. (1999). "The Unified Software Development Process", Addison-Wesley.

[LAB+99] Liao, M., Abecker, A., Bernardi, A., Hinkelmann, K., and Sintek, M.: Ontologies for knowledge retrieval in organizational memories. In: Proceedings of the Learning Software Organizations (LSO) Workshop, Kaiserslauten, Germany. S. 19–29. June 1999.

[RG97] Regli, W. C. and Gaines, D. M. (1997). A repository for design, process planning and assembly. Computer-Aided Design 29(12): 895-905, 1997.

[RMRF01] Raposo, A., Magalhaes, L., Ricarte, I., Fuks, H.: Coordination of Collaborative Activities: A Framework for the Definition of Tasks Interdependencies. P*roc. of the 7th Workshop on Groupware - CRIWG'2001, p.170 - 179. Darmstadt, Germany. IEEE Computer Society Press, 2001.*

[Sa+98] Sandkuhl, Kurt; Nentwig, Lutz; Manhart, Sonia; Lafrenz, Petra: Redesigning CSCW-Applications for Network Computing. In: Proceedings of the Sixth Euromocro Workshop on Parallel and Distributed Processing (PDP '98), Madrid, 21.-23. Jan. 1998. Los Alamitos: IEEE Computer Society, 1998.

[SaM00] Sandkuhl, K.; Messer, B.: Towards Software Reference Architectures for Distributed Groupware Applications. Proc. of PDP2000, Rhodes, Greece, IEEE Computer Society Press, Los Alamitos (USA), 2000.

[SB06] K. Sandkuhl, A. Billig: Towards Ontology-based Requirements Management in Automotive Electronics. 12th IFAC Symposium on Information Control Problems in Manufacturing, Special session on: Ontology-Based Information Management for Industrial Applications, Saint-Etienne, France, May 2006.

[SS96] Schmidt K. and Simone, C.: Coordination Mechanisms: Towards a Conceptual Foundation of CSCW Systems Design. CSCW, Vol. 5, Kluwer (1996).

[St85] Strauss, A.: Work and the Division of Labor. The Sociology Quarterly, Vol. 26, No. 1 (1985).

[St07] Stirna, J., Persson A., Sandkuhl K., (2007) Participative Enterprise Modeling: Experiences and Recommendations, to appear in Proceedings of the 19th International Conference on Advanced Information System Engineering (CAiSE'07), Trondheim, Norway, Springer LNCS

[SD02] Sintek, M. and Decker, S.: Triple - a query, inference, and transformation language for the semantic web. In: Proceedings of International Semantic Web Conference ISWC 2002. Lecture Notes in Computer Science, Bd. 2342, Springer. 2002.

[Sh+04] Shull, F. et al (2004). Knowledge Sharing Issues in Experimental Software Engineering. Empirical Software Engineering, 9, 111-137, 2004.

[SM00] Staab, S. and Maedche, A.: Knowledge portals: Ontologies at work. The AI Magazine. 22(2):63–75. 2000.

[UJ99] Uschold, M. and Jasper, R.: A framework for understanding and classifying ontology applications. In: Proceedings of the IJCAI99 Workshop on Ontologies and Problem-Solving Methods(KRR5), Stockholm, Sweden. 1999.

[W399] W3C: Resource description framework (rdf) model and syntax specification. February 1999. URL: http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

[W303a] W3C: Rdf vocabulary description language 1.0: Rdf schema. January 2003. URL: http://www.w3.org/TR/rdf-schema/.

[W303b] W3C: Web ontology language (owl) use cases and requirements. March 2003. URL: http://www.w3.org/TR/2003/WD-webont-req-20030331/.