PATTERN-BASED KNOWLEDGE ARCHITECTURE FOR INFORMATION LOGISTICS

Kurt Sandkuhl*, **[1]

*School of Engineering, Jönköping University
P.O. Box 1026, 55111 Jönköping, Sweden
**University of Rostock, Institute of Computer Science
Albert-Einstein-Str. 22, 18059 Rostock, Germany

ABSTRACT

Accurate and readily available information is a crucial basis for decision making, problem solving, or performing knowledge-intensive work. In networked organizations with geographically distributed work force and processes, like logistics networks or SME-clusters, quickly finding the right information for a given purpose often is a challenge. An improved information supply would contribute significantly to saving time and most likely to improving productivity. The paper aims at contributing to improved information logistics by bringing together experiences from knowledge modeling and pattern-based reuse in information system development. We propose a pattern-based knowledge architecture with several inter-working layers of services for implementing information logistics in networked organizations. The knowledge architecture forms a framework for selecting and configuring suitable resources for a given problem situation. The knowledge architecture principle and three types of knowledge patterns within the architecture framework are discussed: task patterns for representing enterprise knowledge of member organizations in a networked organization, information demand patterns addressing the information demand of typical roles in a networked organization, and ontology design patterns for capturing context information for decision support.

KEYWORDS: Information Logistics, Knowledge Architecture, Knowledge Pattern

MSC: 90B06

RESUMEN

Una base crucial para la toma de decisiones, resolver problemas, o para desarrollar trabajo intensivo de adquisición de conocimientos, es la disponibilidad de información acurada y obtenible. En organizaciones conectadas con fuerza laboral y procesamiento distribuidos geográficamente, como en las redes logísticas o SME-clústeres, hallar la información correcta para un propósito dado es usualmente un reto. El suministro de informaciòn mejorada contribuye significativamente a ahorrar tiempo y lo aun más deseable mejorar la productividad. Este paper tiene por finalidad contribuir a mejorar la logística informativa, uniendo experiencias del modelado del conocimiento y el re-uso de bases patrones en el desarrollo de sistemas de información. Proponemos una arquitectura de conocimientos, basada en patrones con varias capas de servicios trabajando entrelazadas, para implementar logísticas de información en organizaciones entrelazadas en una red. La arquitectura del conocimiento forma una marco de trabajo para seleccionar y configura recursos adecuados para una situación problèmica dada. El principio de la arquitectura de conocimiento y tres tipos de patrones de conocimientos, dentro del marco de trabajo de la arquitectura, son discutidos: patrones de tareas para representar conocimiento empresarial de miembros de los miembros de la organización , y diseño de patrones ontológicos para capturar información contextual para suportar decisiones.

1. INTRODUCTION

Accurate and readily available information is a crucial basis for decision making, problem solving, or performing knowledge-intensive work. In networked organizations with geographically distributed work force and processes, like logistics networks or SME-clusters, quickly finding the right information for a given purpose often is a challenge. Some studies show that users spend a lot of time in searching for the right information causing unnecessary delays and costs. An example is [1] showing that 2 out of 3 top or mid-level managers participating in a

[1] Kurt.Sandkuhl@jth.hj.se, Kurt.Sandkuhl@uni-rostock.de

study among Swedish enterprises perceive an information overload due to “far too much information” (37%) or “too much information” (29%). An improved information supply would contribute significantly to saving time and most likely to improving productivity.
The paper aims at contributing to improved information logistics by bringing together experiences from knowledge modeling and pattern-based reuse in information system development. We propose a pattern-based knowledge architecture with several inter-working layers of services for implementing information logistics in networked organizations. The knowledge architecture forms a framework for selecting and configuring suitable resources for a given problem situation. The contributions of the paper are: (a) it proposes a novel vision of architecting knowledge, which is considered promising for knowledge-based systems and decision support systems, and (b) it contains a general review of knowledge architecture and knowledge pattern concepts and (c) it shows the application potential of knowledge architectures for supply networks.
The remaining part of the paper is structured as follows: the next section describes the lifecycle of supply networks as a motivating background for this paper including an illustrative scenario from automotive supplier industries. Afterwards the knowledge architecture principle and several types of knowledge patterns within the architecture framework are discussed. Finally, the supply network scenario is revisited to investigate suitability of the different knowledge pattern types for this scenario. A discussion of achievement and future work concludes the paper.

2. SUPPLY NETWORK LIFECYCLE

A flexible supply network includes independent companies based on the principle of cooperation within a defined application domain, which are capable of coordinating their activities for production and delivery of the desired product/service. Organizations of this form use information and communication technologies to extend their boundaries and physical location and form multiple links across the boundaries to work together for a common purpose [9]. Such networks experience different phases, which form the “life-cycle” of a network organization and can be considered as organizational frame for competence supply. The most important phases are [2]:

- Community building: enterprises with joint objectives or interests gather in a community of loosely coupled members. Initial purpose is information exchange and communication within the network in order to prepare collaboration in joint business, load balancing between partners or sharing of production resources.
- Formation: based on specific requirements for a collaboration project (e.g. a joint engineering activity, like product development), the formation of a project team is started based on the capabilities of the members. As a result of this phase, potential partners with respect to the specific requirements have been identified.
- Integration: potential team members have been selected and negotiate the legal and financial conditions for joint project work. Furthermore, a collaboration infrastructure is being implemented for all relevant levels of collaboration. The result of this phase is a project network.
- Operation: the collaboration project is carried out within the project network. This operation is supported by the collaboration infrastructure.
- Discontinuation: the project network discontinues to exist. Dis-integration on all levels of the collaboration infrastructure and with respect to legal and financial issues is carried out.
- Community dissolvement: the joint objectives or interests within the community no longer exist. The network is dissolved.

Figure 1 illustrates the lifecycle of flexible supply networks. Usually it is not fully obvious to the network members, which competence and resources are available from which partner in which quantity to which expenses and how to access them. In this context, efficient support for configuration of collaborations and efficient reuse of existing knowledge is a critical success factor. Configuration includes selection of suitable partners based on their competences and integration of work processes and existing knowledge sources in order to ensure a common level of knowledge and commitment. This also includes providing relevant information for decision making and operations support.
In order to illustrate the concept of supply networks, this section presents a case from distributed product development in a networked organisation from automotive supplier industry, which originates from the MAPPER project [8]. The main partner is the business area “seat comfort components” of a first tier automotive supplier from Scandinavia, working with development and manufacturing of products for the automotive business world wide. The main products are seat comfort products, like seat heating, seat ventilation, lumber support and head restraint. Development of products in this business area includes identification of system requirements based on customer requirements, functional specification, development of logical and technical architecture, co-design of electrical and mechanical components, integration testing and production planning including production logistics, floor planning and product line planning.

Within the first tier supplier, this process is geographically distributed involving engineers and specialists at several locations and SMEs from the region. A high percentage of seat comfort components are product families, i.e. various versions of the components exist and have to be maintained and further developed for different product models and different customers. In this context, fast and flexible product development and integrated management of concurrently performed forward-development processes is of crucial importance. Smooth collaboration and information sharing is a key success factor to meet these basic needs.

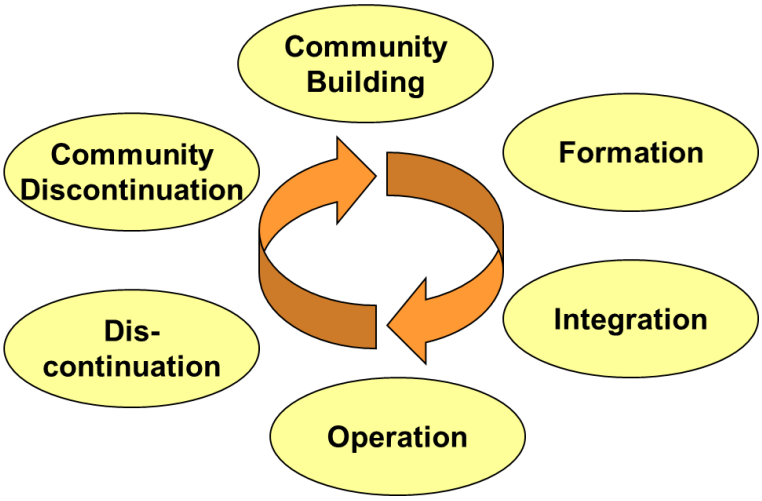

Figure 1. Lifecycle of flexible supply networks

Figure 2 shows a typical collaboration set-up for collaborative design. The customer for a new variant of a seat heating is an Original Equipment Manufacturer (OEM), e.g. for trucks. The first tier supplier receives the order for designing and manufacturing the seat heating and involves several sub-suppliers and partners. These partners are responsible for specific components, like the carrier material or the copper wires, or for specific services, like the controller design or manufacturing of the control unit. The first tier supplier controls the overall design process, contributes own components and services, and performs the system integration.

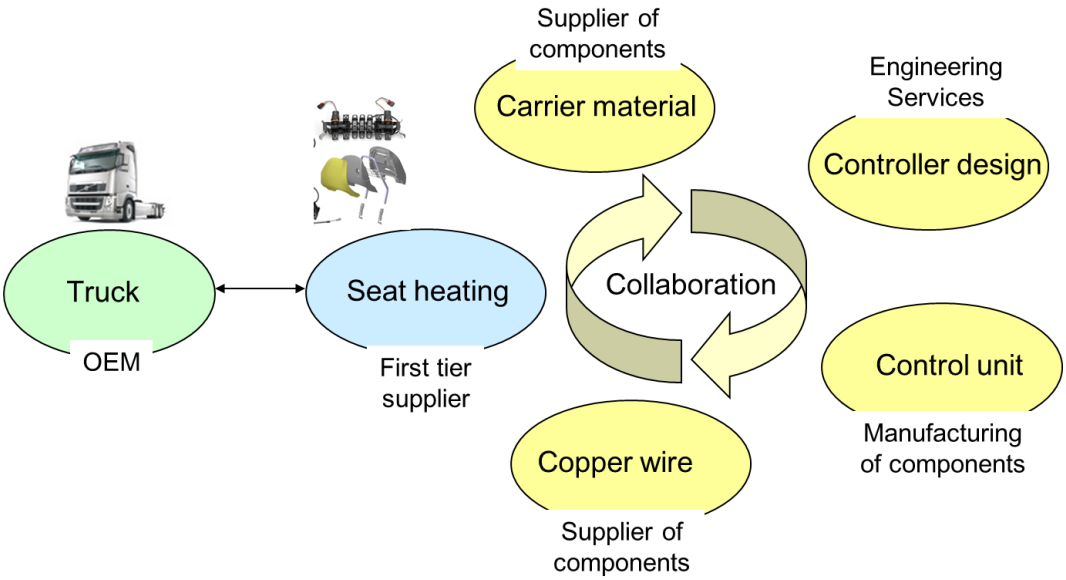

Figure 2. Example supply network in collaborative product design

3. KNOWLEDGE ARCHITECTURE APPROACH

The term knowledge architecture originates from knowledge modelling and knowledge representation, which is a well-researched area in computer science and discussed in [20], to take an example. However, the term "knowledge architecture" is less established and needs clarification: Architectures in general identify main building blocks of the system under consideration including their interfaces and structural relationships. Similarly, knowledge architectures

focus on the knowledge building blocks needed for a specific application and their relationships. In this paper, an enterprise perspective has to be taken in order to also capture potential dependencies between knowledge building blocks and business models. The term enterprise knowledge architecture will be used and defined as follows: *The enterprise knowledge architecture identifies elements of enterprise knowledge including their structural relationships and their context of use [18]*. The main difference in comparison to other architecture perspectives is that the context of knowledge use is modeled explicitly, since the context of use is essential for tailoring the knowledge to the demand at hand. In our case, "context" includes both, the characteristics needed to determine the decision situation in a member company of the supply network and the demands of an individual user. Furthermore, an essential part of the knowledge architecture is the knowledge about the services, which the network members possibly could provide.
The constituents of an enterprise knowledge architecture potentially include business processes, organization structures, products structures, IT-systems or any other perspective relevant for the system under consideration. Established approaches for modeling enterprise knowledge can be divided into at least two major communities: the enterprise engineering community and the artificial intelligence inspired community. Chen et al. [4] provide a detailed account of enterprise modeling and integration approaches from an enterprise engineering perspective. Fox and Gruninger [7] are prominent representatives of the AI-related approaches favoring ontologies for knowledge representation.
More concrete, the proposed knowledge architecture consists of three main layers:

- Enterprise knowledge of the different member enterprises in the supply network, including process, product and service knowledge,
- Knowledge about the information demand of typical roles in the supply networks, like product design managers, process verifiers, quality managers or production logistics responsible,
- Knowledge about actual situations of these typical roles.

In addition to the above characteristics of knowledge architectures, which are primarily motivated by the application scenario of supply networks, such architectures also need certain technical qualities. A knowledge architecture is supposed to contribute to engineering knowledge-based systems and to capturing and reusing organizational knowledge. In this context, a proper representation is required. General requirements connected to this representation are that the terminology used in the knowledge architecture needs to be shared and made explicit. Furthermore, it must be possible to define the components of the architecture and express them as such with different levels of detail, and to express constraints guiding or limiting the composition of components. Additionally, the representation must also be suitable for discussing with business stakeholders, like in the supply network scenario.
Since so far there are no standards or proven approaches for knowledge architecture representation, related work from other areas might provide relevant technologies:

- In software engineering, architecture description languages have been developed, like ArchC [15], DAOP-ADL [14] or π-ADL [13]. Their focus is on specifying components, services offered by the components, interface to the components and constraints to be applied. However, in practice of software engineering, often block-diagrams or UML specifications are used [19].
- In enterprise modeling, ontology-based approaches and visual models are commonly in use, as already indicated above. From the perspective of knowledge architectures, active knowledge modeling is a promising approach due its combination of visual language and formality [10]. Furthermore, more formalized approaches, like DEMO, would provide extensive language support [5].
- In knowledge engineering, architecting knowledge bases recently has been investigated in the context of the semantic web. Examples here are ontology-based techniques and include ontology modules [6] and ontology patterns [1].

Looking back at the general requirements to the knowledge architecture representation, the software engineering approaches seem to be not suitable since component interfaces in terms of services are not a priority in knowledge architectures. But a combination of enterprise modeling (due to the requirement to be suitable for business stakeholders) and knowledge engineering (due to the requirement of formalization) seems appropriate. Since enterprise models and ontologies can be converted into each other, we decided to investigate the use of ontologies. Among the many ontology definitions available, we will use the following definition, which is based on [11]:

An ontology structure is a 5-tuple $O := \{C, R, H^C, rel, A^O\}$, consisting of

- two disjoint sets C and R whose elements are called concepts and relations respectively.

- a concept hierarchy H^C: H^C is a directed relation $H^C \subseteq C \times C$ which is called concept hierarchy or taxonomy. $H(C_1,C_2)$ means that C_1 is a subconcept of C_2.
- a function $rel : R \rightarrow C \times C$, that relates concepts non-hierarchically (note that this also includes attributes). For $rel(R)=(C_1,C_2)$ one may also write $R(C_1,C_2)$.
- a set of ontology axioms A^O, expressed in an appropriate logical language.

With this definition as a basis, we will investigate the use of different types of knowledge patterns as components of a knowledge architecture.

4. KNOWLEDGE PATTERNS

In computer science, patterns are considered a promising way to capture proven practices in order to facilitate reuse. Within the knowledge architecture approach, we propose to use task patterns for representing enterprise knowledge, information demand patterns for the typical information demand of roles and ontology design patterns for capturing the actual decision context. These three pattern types, which can be considered as knowledge patterns, will be briefly introduced in the following.

4.1. Task Patterns (TP)

The concept of task pattern is a result of the EU-FP6 project MAPPER [8]. In this project, collaborative engineering was supported by adaptable models capturing best practices for reoccurring tasks in networked enterprises. These best practices were represented as active knowledge models using the POPS* perspectives. Active knowledge models are visual models of selected aspects of an enterprise, which cannot only be viewed and analyzed, but also executed and adapted during execution. The POPS* perspectives include the enterprise's processes (P), the organization structure (O), the product developed (P), the IT system used (S) and other aspects deemed relevant when modeling (*). In the context of task patterns, these other aspects were the competences required to fill roles in the organization structure and business requirements attached to different processes.
The term "task patterns" was introduced for these adaptable visual models, as they are not only applicable in a specific company, but are also considered relevant for other enterprises in the application domain under consideration. Task pattern in this context is defined as "self-contained model template with well-defined connectors to application environments capturing knowledge about best practices for a clearly defined task" [16]. In this context, self-contained means that a task pattern includes all POPS* perspectives, model elements and relationships between the model elements required for capturing the knowledge reflecting a best practice. Model template indicates the use of a well-defined modeling language and that no instances are contained in the task patterns. Connectors are model elements representing the adaptation of the task pattern to target application environments.
In order to use task patterns as components in our knowledge architecture, we need to formalize the concept more and match it onto the formalization shown in the previous section, i.e. onto the ontology representation defined for the knowledge architecture.

We define a task pattern as a tuple $TP := \{E, R, sub, fol, resp, \sup, req\}$, consisting of:

- Two disjoint sets E and R whose elements are called entity and relations respectively.
- A function $sub : R \mapsto E \times E$ that relates entities and sub-entities. With $sub(R) = (E_1,E_2)$ we define E_2 as a sub-entity of E_1. With this function, it is possible to express hierarchical relationships between processes, organisation units, products and IT-systems in a task pattern.
- A function $fol : R \mapsto E \times E$ that relates entities with following entities. With $fol(R) = (E_1,E_2)$ we define E_2 as following the entity E_1 from a time perspective. This function allows for capturing process chains.
- A function $resp : R \mapsto E \times E$ that relates entities with a responsible entity. With $resp(R) = (E_1,E_2)$ we define E_1 as responsible for E_2. This function is used for expressing the responsibility of an entity of the organisation structure for a process, an IT-system of a product entity or sub-entity.
- A function $req : R \mapsto E \times E$ that relates required entities. With $req(R) = (E_1,E_2)$ we define E_2 as a required entity for E_1. This function is used for expressing the need of a resource (product or IT system entity) for a process.
- A function $\sup : R \mapsto E \times E$ that relates supportive entities. With $sup(R) = (E_1,E_2)$ we define E_2 is supported by E_1. This function is applied for showing the support of a resource (e.g. an IT system entity) for a process or a competence for an organisation structure entity.

Table 1: TP to EM mapping

Task Pattern	Enterprise Ontology	Remark
E	C	Entities are represented as concepts
sub(E1,E2)	H (C1,C2)	Entity hierarchy is represented as concept hierarchy
fol(E1,E2)	fol(C1,C2)	Other entity relations are represented as the same relationship type in the ontology
resp(E1,E2)	resp(C1,C2)	
req(F1,F2)	req(C1,C2)	
sup(F1,F2)	sup(C1,C2)	

The above definitions provide a basic formalization for the POPS* perspectives of task patterns and allow for a rich set of relations between various entities. This formalization could be extended by typing the entities and adding a wider set of functions for reflecting all elements of visual modeling languages. As ontologies have a clearly richer descriptive power than task patterns, the mapping on a notation level does not cause serious technical problems. The approach proposed in this paper is to preserve existing hierarchies between entities by mapping an entity sub-entity structure to a concept hierarchy in the ontology. The other relationships between entities in the task pattern, i.e. follows, is_responsible_for, is_required_for and is_supporting, are represented in the ontology by creating the respective relationship types in the ontology. Table 1 summarizes the proposed mapping.

4.2. Information Demand Patterns (IDP)

The general idea of information demand patterns is similar to most pattern developments in computer science: to capture knowledge about proven solutions in order to facilitate reuse of this knowledge. In this paper, the term information demand pattern will be defined as follows: *An information demand pattern addresses a recurring information demand problem that arises for specific roles and work situations in an enterprise, and presents a conceptual solution to it*. An information demand pattern consists of a number of essential parts used for describing the pattern [17]:

- A statement about the organizational context where the pattern is useful.
- Problems of a role that the pattern addresses. The tasks and responsibilities a certain role has are described in order to identify and discuss the challenges and problems, which this role usually faces in the defined organizational context.
- The conceptual solution that resolves the problem includes the information demand of the role, quality criteria for the different parts of the information and a timeline indicating the points in time when the different information parts should be available at the latest
- The effects that play in forming a solution. If the needed information part should not be available or arrive too late this might affect the possibility of the role to complete its task and responsibilities.

The above parts of a pattern are described in much detail in the textual description of the pattern. Additionally, a pattern can also be represented as a kind of enterprise model. This model representation can be used to show the link to the relevant task patterns introduced in the previous section, including the relation of the role to co-workers and other roles in the organization, the relation between the different parts of the information demand and IT system in the enterprise, which are potential source of this information; and the relation of tasks and responsibilities to processes in the organization.
In order to use information demand patterns as components in our knowledge architecture, we again need to formalize this pattern concept and match it onto the ontology representation of the knowledge architecture. Since information demand patterns also use an enterprise model representation very similar as to task patterns, we can use the same formalization as introduced for task patterns. However, it has to be emphasized that only if an enterprise model representation exists, an information demand pattern can be used as component in the knowledge architecture. The textual representation alone would not be sufficient, since it is not precise enough for matching to the ontology structure of the knowledge architecture.

4.3. Ontology Design Patterns (ODP)

Ontology design patterns can be defined as a set of ontological elements and construction principles that solve a clearly defined particular ontology engineering problem [3]. Ontology design patterns are considered as encodings of best practices, which help to reduce the need for extensive experience when developing ontologies, i.e. the well-defined solutions encoded in the patterns can be exploited by less experienced engineers when creating ontologies. The two types of ODP probably receiving most attention are logical and content ODP. Logical ODP focus only on the logical structure of the representation, i.e. this pattern type is targeting aspects of language expressivity, common

problems and misconceptions. Content ODP often are instantiations of logical ODP offering actual modeling solutions. Due to the fact that these solutions contain actual classes, properties, and axioms, content ODP are considered by many researchers as domain-dependent. For the purpose of representing knowledge about the decision context of an information logistics situation, ODP such as "situation" or "role-actor" can be applied. Since ODP use the same ontological formalization as the knowledge architecture representation, no transformation of mapping needs to be applied. ODP can be used as components in the knowledge architecture.

5. KNOWLEDGE ARCHITECTURE AND PATTERN USE IN SUPPLY NETWORKS

Based on the lifecycle model for supply networks introduced earlier in this paper, this section will investigate how knowledge architectures in general and the different pattern types in particular can support the lifecycle phases. For this paper, formation phase and integration phase are of specific interest, since these phases are decisive for finding the right partners for a collaborative project and creating a working network. The formation phase requires an efficient way to describe the competences and services required for the joint network activity, e.g. for collaborative product design, completely on a sufficient level of detail. The integration phase adds requirements regarding selection of partners. Table 2 summarizes the requirements from both phases, i.e. what do knowledge architecture and knowledge patterns have to support?

Table 2: Requirements from lifecycle phases to competence demand models

Formation Phase	*Integration Phase*
Tasks to be completed by the complete supply network or by single partners	Approaches and tools for matching the demand of the collaboration project with the existing network members' competences
Results to be delivered by the task	Selection of individual team members at the selected member organizations,
Express, store and access competence profiles of organisations and individuals in an information system	Support for integration of IT infrastructure and enterprise systems required for the collaboration.
Capacity needed in terms of machinery or equipment	
Instruments and tools required	

Table 3 uses the requirements from table 2 and shows how the knowledge architecture and the pattern types can support implementation of the requirements.

Table 3: Support of knowledge architecture and patterns regarding requirements from lifecycle phases

Requirement from Formation/Integration	*Knowledge Architecture*	*Knowledge Patterns*
Tasks to be completed by complete network or single partners	Defines overall structure how to model tasks in a consistent and inter-related way	TP and IDP: Tasks are common elements of enterprise models (EM) and information demand descriptions ODP: Task type can be modeled as concepts with appropriate attributes
Results to be delivered by task	Defines controlled vocabulary for specification of resources	TP: Outputs of tasks are common EM elements and can be used to model results IDP: shows dependencies between information demand and results of a task ODP: Result types can be modeled as concepts; relationship type between task and result expresses dependency
Instruments and tools required	Defines controlled vocabulary for specification of instruments and tools	TP: Resources of tasks are common EM elements and can be used to model instruments/tools ODP: Instrument types and tool types can be modeled as concepts; relationship type expresses dependency
Capacity needed in terms of machinery or equipment	No support	TP: Most EM approaches allow for attributes of resource, which can be used for storing capacity ODP: Can be expressed as attribute of the instrument or tools concepts
Express, store and access competence profiles of organisations and individuals in an information system	Contributes to the design of the knowledge base for such an information system	TP: Requires usually a conversion in highly structured format, which is searchable and can easily be processed ODP: Ready for use in knowledge bases
Matching competence demand with the existing competence profiles	No support	ODP: Exact matching possible based on inference; semantic matching is emerging technology
Selection of individual team members at member organization	No support	TP and ODP: possible; depends on granularity level of EM / ontology
Support for integration of IT infrastructure and enterprise systems required for the collaboration.	No support	TP: EM include information required for this purpose on logical level, but usually not all technical information ODP: Can be used as element in interoperability solutions

The above table shows that all pattern types and the knowledge architecture contribute to the requirements identified in formation and integration phases. Most promising seems to be the task pattern type, which is contributing to all requirements except the requirement of matching competences. Ontology patterns seem to be a suitable complement to task patterns. This indicates that a knowledge architecture consisting of orchestrated task patterns and an integrated controlled vocabulary as part of it would be most promising. Information demand patterns contribute only marginally to the requirements.

6. SUMMARY AND FUTURE WORK

The paper investigated the use of different pattern types within a knowledge architecture in the context of information logistics. Based on a lifecycle model for networked organizations and an illustrative example, the requirements from the lifecycle phases were contrasted with the characteristics of the pattern types. From an application perspective, the conclusion of the work so far is that a knowledge architecture consisting of orchestrated task patterns and supporting ontology design patterns would be most promising.
From a technical perspective, it was shown that an ontological representation of the knowledge architecture could serve as a technical basis for accommodating both, task patterns transformed to an ontology representation and ontology design patterns. However, when constructing an actual knowledge architecture from task patterns and ODP from different authors or sources, the known problems of integrating heterogeneous information into a homogeneous representation can be expected. Examples are semantic mismatches between concepts from different sources, different levels of granularity or different modeling styles. When exploring the use of the knowledge architecture approach for a logistics application [18], we found it useful to initialize the knowledge architecture with a defined vocabulary, i.e. a domain ontology for the field under consideration, and adjust all knowledge patterns to this vocabulary.
Future work will be of experimental and conceptual nature. From an experimental perspective, the proposed approach has to be implemented and evaluated in controlled environments or real-world cases. This will most likely lead to changes, refinements and improvements of the proposed approach. The conceptual work includes to further elaborate the aspects of knowledge architecture formalization, identifying integration and combination potential between the different pattern types, and adequate software infrastructure for implementation.

RECEIVED OCTOBER, 2012
REVISED APRIL 2013

REFERENCES

[1] BLOMQVIST, E. (2009): **Semi-automatic Ontology Construction based on Patterns**. PhD thesis, Linköping University, Department of Computer and Information Science.

[2] BLOMQVIST E, LEVASHOVA T, ÖHGREN A, SANDKUHL K and SMIRNOV, A. (2005): Formation of Enterprise Networks for Collaborative Engineering. **Proceedings CCE 05**, Sopron (Hungary) :, April 2005, ISBN 91-975604-1-3.

[3] BLOMQVIST E and SANDKUHL, K. (2005): Patterns in Ontology Engineering. **Proc. 7th ICEIS**, Miami, USA, May 2005.

[4] CHEN D, DOUMEINGTS G, and VERNADAT, F . (2008): Architectures for enterprise integration and interoperability: Past, present and future. **Computers in industry**, Vol. 59 No. 7: 647-659.

[5] DIETZ J. (2006) : **Enterprise Ontology – Theory and Methodology**, Springer, 2006.

[6] DORAN P, TAMMA V and IANNONE, L. (2007): Ontology module extraction for ontology reuse: an ontology engineering perspective. **Proceedings of the sixteenth ACM conference on Conference on information and knowledge management** (CIKM '07):. ACM, New York, NY, USA, 61-70. DOI=10.1145/1321440.1321451

[7] FOX M.S and GRUNINGER, M. (1998): Enterprise Modelling. **AI Magazine**, Vol 19, No 3.

[8] JOHNSEN S, SCHÜMMER T, HAAKE J, PAWLAK A, JØRGENSEN H, SANDKUHL K, STIRNA J, TELLIOGLU H and JACCUCI, G. (2007) : Model-based Adaptive Product and Process Engineering. In: RABE M and MIHÓK, P. (Eds) : **New Technologies for the Intelligent Design and Operation of Manufacturing Networks**. Fraunhofer IRB Verlag: Stuttgart (Germany) :.

[9] LAUDON K.C and LAUDON, J. P.(2000) : **Management Information Systems: Organisation and Technology in the Networked Enterprise.** Prentice Hall International: New York.

[10] LILLEHAGEN F and KROGSTIE, J. (2009): **Active Knowledge Modelling of Enterprises**. Springer, 2009. ISBN: 978-3-540-79415-8.

[11] MAEDCHE, A. (2003): **Ontology Learning for the Semantic Web**. Kluwer Academic Publishers: Norwell.

[12] ÖHGREN A and SANDKUHL, K. (2008): Information Overload in Industrial Enterprises - Results of an Empirical Investigation. **Proceedings ECIME 2008**, London, UK.

[13] OQUENDO, F. (2004): π -ADL: an Architecture Description Language based on the higher-order typed π -calculus for specifying dynamic and mobile software architectures. **SIGSOFT Softw. Eng. Notes** 29, 3 (May 2004):, 1-14. DOI=10.1145/986710.986728

[14] PINTO M, FUENTES L, et al. (2003): DAOP-ADL: An Architecture Description Language for Dynamic Component and Aspect-Based Development. In PFENNING F and SMARAGDAKIS Y.: **Generative Programming and Component Engineering**. Springer: Berlin Heidelberg. 2830: 118-137.

[15] RIGO S, ARAUJO G., et al. (2004): ArchC: a systemC-based architecture description language. **Computer Architecture and High Performance Computing**. SBAC-PAD 2004.

[16] SANDKUHL, K. (2010): Capturing Product Development Knowledge with Task Patterns. **Quarterly Journal of Control & Cybernetics**, Issue 1, 2010.

[17] SANDKUHL, K. (2011) : Information Demand Patterns. **Proc. PATTERNS 2011**, The Third International Conferences on Pervasive Patterns and Applications, pp. 1-6. September 25-30; Rome, Italy.

[18] SANDKUHL, K. (2011): Towards Intelligent Information Logistics Services: Case Study from Transportation. In: IVANOV D, KOPFER H, HAASIS H.D and SCHÖNBERGER, J. (Eds.) :: **Dynamics and Sustainability in International Logistics and Supply Chain Management**, pp. 317-328. Cuvillier Verlag: Göttingen.

[19] SOMMERVILLE I. (2007): **Software Engineering** (Edition 8) :, Pearson Education.

[20] SOWA, J.F. (2000) : **Knowledge Representation: Logical, Philosophical, and Computational Foundations**, Brooks Cole Publishing Co.: Pacific Grove, CA.