Towards an Ontology for Information Systems Development

Mauri Leppänen

Department of Computer Science and Information Systems
P.O. Box 35 (Agora), FI-40014 University of Jyväskylä, Finland
mauri@cs.jyu.fi

Abstract. Various frameworks, meta models and reference models have been proposed to describe information systems development (ISD) and ISD methods. Most of them are informal or focused on some specific aspects. This paper presents an ISD ontology, which aims to provide an integrated conceptualization of ISD through anchoring it upon the contextual approach. The ISD ontology is composed of concepts, relationships and constraints referring to purposes, actors, actions and objects of ISD. It is presented as a terminology with defined concepts and in meta models in a UML-based ontology representation language. We believe that although not being complete the ISD ontology can promote the achievement of a shared understanding of contextual aspects in ISD. It can be used to analyze and compare existing frameworks and meta models and as a groundwork for engineering new ISD methods, or parts thereof.

1 Introduction

To advance the understanding, management and improvement of an IS engineering process, a large number of frameworks, meta models and reference models (shortly ISD artifacts) have been constructed for information systems development (ISD) and ISD methods. Most of these artifacts view ISD from specific viewpoints based on some approache, e.g. a transformation approach, a decision-making approach, a problem solving approach, or a learning approach. As a result, ranges of concepts and constructs in these artifacts are rather narrow-scoped. To enable a more comprehensive view on ISD, ISD should be conceived as a context with all its facets, distinguishing purposes, actors, actions, objects, facilities, locations and time aspects.

The purpose of this study is to present an ISD ontology that is based on a contextual approach. An ontology is a kind of framework unifying different viewpoints, thus functioning in a way like a lingua-franga [4]. It is an explicit specification of a conceptualization of some part of reality that is of interest [11]. The *ISD ontology* provides a conceptualization of contextual aspects of ISD through a vocabulary with explicit definitions. To enhance the clarity and preciseness of the ontology, we deploy a UML-based ontology representation language in describing the ISD ontology in meta models.

The ISD ontology is intended for descriptive, analytical and constructive use. For the descriptive purposes, the ontology offers concepts and a terminology for conceiving, understanding, structuring and presenting contextual phenomena of ISD. In the analytical sense, the ontology can be used to analyze and compare existing ISD artifacts. In the constructive sense, the ontology is to support the engineering of new ISD artifacts, such as ISD models, techniques and methods, by providing a coherent and consistent groundwork for them.

The rest of the paper is structured as follows. First, we define the notions of context and contextual approach and apply them to define the ISD ontology. In the next five sections, we specify four main ISD domains (i.e. ISD purpose domain, ISD actor domain, ISD action domain, and ISD object domain) and inter-relationships between them. We end with discussions and implications to practice and research.

2 Contextual Approach

Based on a large literature review about the notion of context in several disciplines, such as knowledge representation and reasoning, pragmatics, computational linguistics, sociolinguistic, organizational theory and information systems, we came to the following generic definition: *context* means a whole that is composed of things connected to one another with contextual relationships. A thing gets its meaning through the relationships it has to the other things in that context. To recognize a proper set of contextual concepts we drew upon relevant meaning theories. We identified semantics (e.g. case grammar [10]), pragmatics [27], and activity theory [8] to be such theories. They concern sentence context, conversation context, and action context, correspondingly. Anchored on this groundwork, we can define seven domains, which serve concepts for specifying and interpreting contextual phenomena. These contextual domains are: purpose, actor, action, object, facility, location, and time. To structure the concepts within and between these domains, we define the Seven S's Scheme: *For **S**ome* purpose, ***S**omebody* does ***S**omething* for ***S**omeone,* with ***S**ome* means, ***S**ometimes* and ***S**omewhere*. Implied from the above, we define the contextual approach as follows: according to the *contextual approach*, individual things in reality are seen to play specific roles in a certain context, and/or to be contexts themselves. The contexts can be decomposed into more elementary ones and related to one another through inter-context relationships.

We have previously applied the contextual approach to enterprises [25] and method engineering [26]. Here, we apply it to ISD. Based on the contextual approach, we see *information system development* as a context in which ISD actors carry out ISD actions to produce ISD deliverables contributing to a renewed or a new IS, by means of ISD facilities, in a certain organizational and spatio-temporal context, in order to satisfy ISD stakeholders' goals. The notion provides an extensive view on contextual aspects of ISD. ISD work is guided by ISD requirements and goals which, through elicitations and negotiations, become more complete, shared and formal [31]. ISD work is carried out by ISD actors with different motives, skills and expertise, acting in different roles in organizational units that are situationally established. ISD work is composed of various ISD actions, structured in concordance with the selected

ISD approaches and ISD method, and customized according to conventions in the organization. The final outcome of ISD is a new or improved information system composed of interacting social arrangements and technical components. ISD work consumes resources (e.g. money and time) and is supported by computer-aided tools (e.g. CASE tools). ISD actors, ISD deliverables and ISD facilities are situated in certain locations, and are present in certain times.

Based on the above, we define the ISD ontology as follows: the *ISD ontology* provides concepts and constructs for conceiving, understanding, structuring and representing contextual phenomena in ISD. The concepts and constructs in the ISD ontology have been derived in deductive and inductive manners. We have searched for disciplines and theories that address social and organizational contexts and derived a basic categorization of concepts into contextual domains from them. After that we have enriched the contents and structure of each contextual domain by a thorough analysis of existing artifacts, and by selecting, integrating and adapting those concepts in them that were found to be applicable. We have also closely examined empirical studies on ISD practice (e.g. [32]). Our aim has been to establish a common core from which concepts and constructs for specific ISD approaches could be specialized.

In the following, we define four of the ISD domains, namely the ISD purpose domain, the ISD actor domain, the ISD action domain, and the ISD object domain. For each domain, we define basic concepts, relationships and constraints. After that, we delineate relationships between the domains. Due to the page limit, the description of the ISD domains is brief. A more profound discussion is given in [24].

3 ISD Purpose Domain

The *ISD purpose domain* embraces all those concepts and constructs that refer to goals, motives, or intentions of someone or something in the ISD context (Figure 1). The concepts may show a direction to which to proceed, a state to be attained or avoided, and reasons for them. Reasons can be expressed in terms of requirements, problems, opportunities, threats, etc.

An *ISD goal* expresses a desired state or event with qualities and quantities, related to the ISD context as a whole, or to some parts thereof. *Hard ISD goals* have pre-specified criteria for the assessment of the fulfillment of ISD goals, while *soft ISD goals* have not [29]. An *ISD requirement* is some quality or performance demanded in and for the ISD context. It is a statement about the future [30]. ISD requirements are classified along three orthogonal dimensions [31]: specification, representation, and agreement. ISD requirements become goals after having been agreed on. All the requirements cannot be accepted to be goals, since their fulfillment may, for instance, go beyond the resources available. An *ISD problem* is the distance or mismatch between the prevailing ISD state and the state reflected by the ISD goals. ISD problems can be structured, semi-structured or non-structured.

Some of the ISD purposes concern an IS. They are called IS purposes, and they are further sub-divided into IS goals, IS requirements and IS problems. For the evaluation of IS designs, implementation and use, a large variety of IS criteria are used. An *IS criterion* is a standard of judgment presented as an established rule or principle for evaluating some feature(s) of an IS in terms of IS purposes. An *IS requirement* means a condition or capability of the IS needed by an IS client or an IS worker to solve a problem, or to achieve a goal [16]. IS requirements are divided into *functional requirements* and *non-functional requirements*.

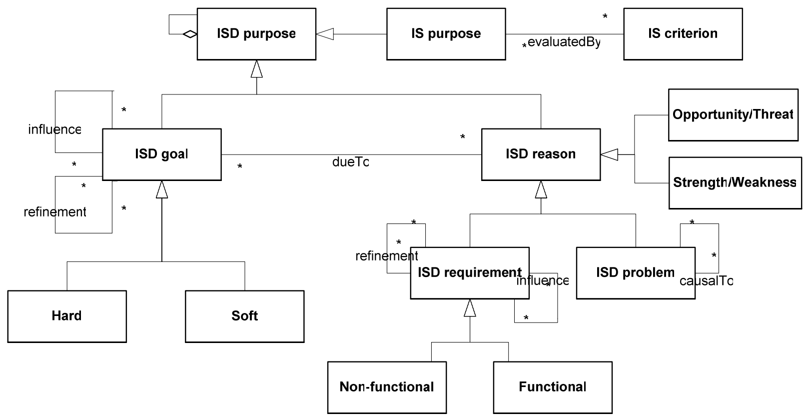

Fig. 1. ISD Purpose Domain

The ISD goals, as well as the ISD requirements, are related to one another through many kinds of relationships. A *refinement relationship* means that an ISD goal can be reached when certain ISD goals below it in the ISD goal hierarchy are fulfilled. An *influence relationship* means that an ISD goal has impacts, positive or negative, on the achievement of another ISD goal [20]. The ISD goals with negative interrelationships are referred to as conflicting requirements [23]. A *causalTo relationship* between two ISD problems means that the appearance of one ISD problem is at least a partial reason for the occurrence of another ISD problem.

4 ISD Actor Domain

The *ISD actor domain* consists of all those concepts and constructs that refer to human and active part of the ISD context (Figure 2). Actors own, communicate, transform, design, interpret, code etc. objects in the ISD context. They are responsible or responsive to trigger and cause changes in the states of objects. They are also aware of their intentions and capable, at least to some degree, of reacting to fulfill their goals.

An *ISD actor* is an ISD human actor or an administrative actor, who is involved, one way or another, in the ISD context. An *ISD human actor* means a person or a group of persons contributing to ISD work. An *ISD administrative actor* is an ISD position, or a composition of ISD positions. An *ISD position* is a post of employment occupied by a human ISD actor. It is identified with a title, composed of the defined ISD roles and equipped with a set of skill or capability characterizations. A *capability* means a skill or attribute of personal behavior, according to which behavior can be logically classified [1]. An *ISD role* is a collection of ISD responsibilities and authorities, stipulated in terms of ISD actions. Some ISD roles are not included in any ISD position but are anyhow played by one or more persons.

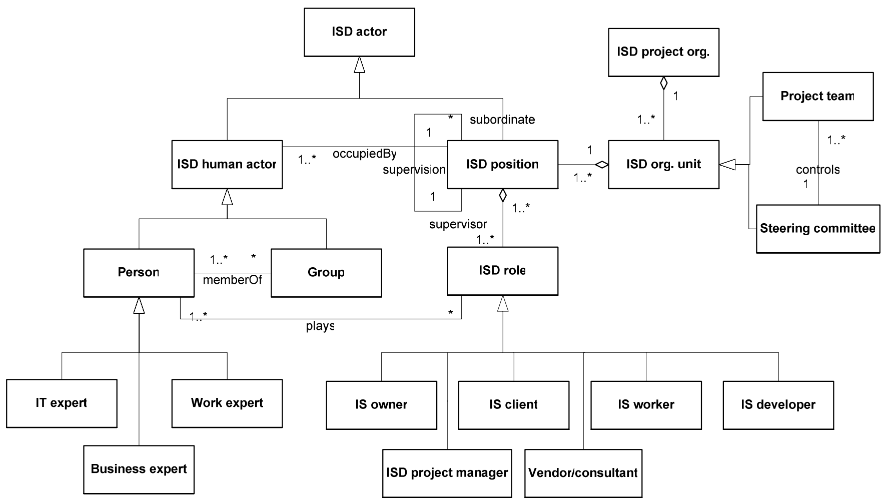

Fig. 2. ISD Actor Domain

The ISD roles are categorized in many ways in the ISD literature, for instance on the bases of social roles and technical roles. In this work, we base our categorization on [5], [2], and [28] and distinguish between six major ISD roles that unify the social and technical nature of ISD work. The roles are: an IS owner, an IS client, an IS worker, an IS developer, an ISD project manager, and a vendor/consultant.

An *IS owner* has the responsibility for, and the authority of, making decisions on the IS as though it were his/her property. An *IS client* is the ISD role player for whom the IS is to be developed. An *IS worker* works with the current IS, and/or will work with the new IS, in order to provide IS clients with information. An *IS developer* works for satisfying needs and requirements put forward by ISD actors in the other roles. An *ISD project manager* makes plans on how to organize an ISD effort. A *vendor / consultant* role is played by a person from outside the organization. With that role, more expertise on some specific organizational or technical issues is imported to the ISD project.

An *ISD project* is a temporary effort with the well-defined objectives and constraints, the established organization, the budget and the schedule, launched for the accomplishment of ISD. An *ISD project organization* is a composition of ISD positions, ISD roles and ISD teams wherein the responsibility, authority and communication relationships are defined [9]. A large project organization is composed of several *organizational units*. The most common units are a steering committee and a project team. A *steering committee* carries the responsibility for the overall management of the ISD project. The day-to-day management is delegated to the project manager**,** who directs and controls the actions of specialists in various disciplines. A *project team* is collected for the execution of the ISD effort.

For each ISD position the most suitable person is sought. For being suitable, the person's skill and experience profile has to match with the expertise profile stated for the ISD position [1]. According to their expertise, the persons involved in ISD can be categorized into *IT experts*, *business experts* and *work experts*.

5 ISD Action Domain

The *ISD action domain* comprises all those concepts and constructs that refer to deeds or events in the ISD context (Figure 3). *ISD actions* are carried out to manage and execute ISD efforts. By them, procedures, rules and policies are selected, incorporated, customized, and implemented to produce desired ISD deliverables. To manage this extensive variety of ISD actions, several categorizations of ISD actions and ISD processes have been presented (e.g. [7, 6]). We recognize four fundamental ISD action structures that are orthogonal to one another: the ISD management – execution structure, the ISD workflow structure, the ISD phase structure, and the IS modeling structure. In addition, there are three generic action structures: the decomposition structure, the control structure (sequence, selection, iteration), and the temporal structure (overlapping, parallel, disjoint). The aforementioned ISD action structures give a natural basis for specializing and decomposing ISD work into more specific ISD actions. Each ISD action is governed by one or more *ISD rules* with the ECAA structure [14]. An instance of an ISD action is called an *ISD process***.** In the following, we consider the four fundamental ISD action structures in more detail.

ISD execution actions produce required ISD deliverables under the guidance and control of ISD management. *ISD management actions* plan, organize, staff, direct, and control ISD work [35]. *ISD planning* refers to designing the goals of an ISD project and the strategies, policies, programs and procedures for achieving them. *ISD organizing* means establishing a formal structure of ISD execution actions and authority relationships between them. *ISD staffing* refers to actions to fill the ISD positions of the ISD project organization and to keep them filled. *ISD directing* means actions to clarify the assignments of ISD teams and persons. *ISD controlling* is needed to ensure that ISD execution actions are carried out according to plans.

The *ISD workflow structure* is composed of ISD workflows. An *ISD workflow* is a coherent composition of ISD actions, which are organized to accomplish some ISD process. They share the same target of action and produce valuable results for ISD actors. A part of an ISD workflow is called an *ISD task*. ISD workflows can be identified among the ISD management actions, as well as among the ISD execution actions. Here we consider them in the context of ISD execution actions. We distinguish between five core ISD workflows: IS requirements engineering, IS analysis, IS design, IS implementation, and IS evaluation [19]. Besides the core workflows, there are supporting workflows, like configuration and change management [22].

According to the *ISD phase structure*, the ISD is seen as being composed of sequential phases. An *ISD phase* means ISD actions that are executed between two

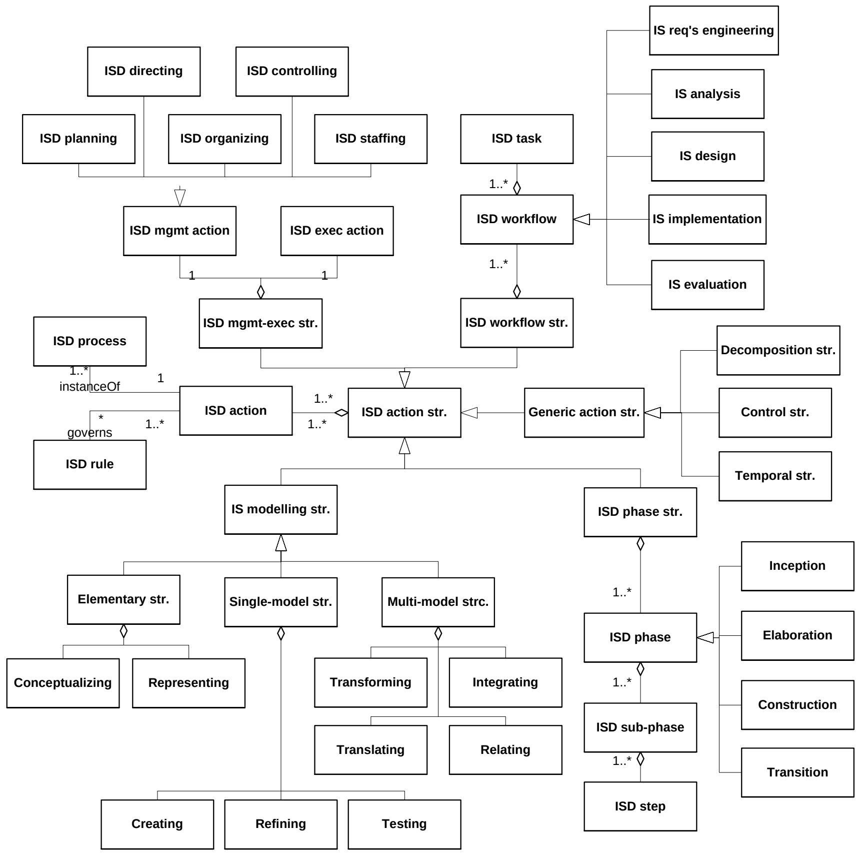

Fig. 3. ISD Action Domain

milestones. By these actions a well-defined set of goals is met, ISD deliverables are completed, and decisions are made on to move or not to move into the next phase [22]. *Milestones* are synchronization points where ISD management makes important business decisions and ISD deliverables have to be at a certain level of completion [15]. Major milestones are used to establish baselines. In ISD methods a large variety of phases with different names are presented. Without wanting to commit to any of them, we have selected the set of phases suggested in [19] and [22] to be an example of the ISD phase structure. It comprises four phases: inception, elaboration, construction, and transition.

Modeling has a focal role in all ISD actions. It is a necessary and frequently used means in the ISD management actions, as well as in the ISD execution actions. Here, we focus on modeling in the latter case, and refer to it as *IS modeling.* The target of IS modeling can be an existing IS or a new IS. There are three kinds of *IS modeling structures*: elementary modeling structure, single-model action structure, and multi-model action structure. The *elementary modeling structure* comprises actions that are always present in IS modeling. These actions are *conceptualizing* and *representing*. The *single-model action structure* comprises IS modeling actions that involve a single model at a time. These actions are *creating*, *refining* and *testing*. The *multi-model action structure* is composed of IS modeling actions that involve, some way or another, two or more IS models at the same time. These actions are *transforming*, *translating*, *relating*, and *integrating* (see the definitions in [24]).

6 ISD Object Domain

The *ISD object domain* comprises all those concepts and constructs that refer to something which ISD actions are directed to (Figure 4). In the ISD literature these are commonly called deliverables, artifacts, decisions, products, work products, and design products. We use the generic term '*ISD deliverable'*. On the elementary level, an ISD deliverable is an assertion, a prediction, a plan, a rule, or a command, concerning the ISD itself, the existing IS, the new IS, the object system (OS), or the utilizing system. We use the term 'OS_{ISD} *construct'* to denote all these parts in the object system of the ISD. The *signifies relationship* expresses a relationship between an ISD deliverable and an OS_{ISD} construct.

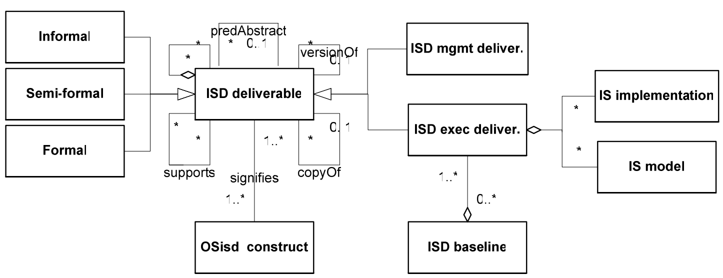

Fig. 4. ISD Object Domain

The *ISD management deliverables* mean plans for, decisions on, directives for, and assessments of goals, positions, actions, deliverables, locations, etc. in the ISD context. The *ISD execution deliverables* refer to descriptions and prescriptions about why, what, and how information processing is carried out, or is to be carried out, in the current IS, or in a new IS, respectively. The ISD execution deliverables comprise informal drafts and scenarios, as well as more formal presentations. The former include instructions and guidelines in the form of training materials, handbooks, and manuals. The latter are presented in the form of *IS models* (e.g. class diagrams, component diagrams), or they are *implementations* of those models (e.g. software components, data bases).

Some of the ISD execution deliverables may be specified to be parts of the ISD baselines with milestones in a project plan. An *ISD baseline* is a set of reviewed and approved ISD deliverables that represents an agreed basis for further evolution and development, and can be changed only through a formal procedure (cf. [19]). The ISD deliverables are presented in some language(s). Presentations may be informal, semi-formal, or formal, including texts, lists, matrices, diagrams, program codes, etc.

The ISD deliverables are related to one another with five kinds of relationships. An ISD deliverable can be *composed of* other ISD deliverables. An ISD deliverable can be used as input to, or as a prescription for, another ISD deliverable (i.e. the *supports relationship*). An ISD deliverable can be the next *version of*, or a *copy of*, another ISD deliverable. Finally, an ISD deliverable may be more abstract than another ISD deliverable in terms of predicate abstraction (i.e. the *predAbstract relationship*).

7 ISD Inter-Domain Relationships

In the sections above the ISD concepts and constructs have been discussed from the viewpoint of one ISD domain at a time. The ISD domains are, however, inter-related in many ways. Figure 5 presents, on a general level, a meta model illustrating the most essential inter-domain relationships. In the meta model one or few essential concepts from each of seven ISD domains are depicted and related to concepts of the other domains. It goes beyond the limit of this paper to discuss these relationships in more detail.

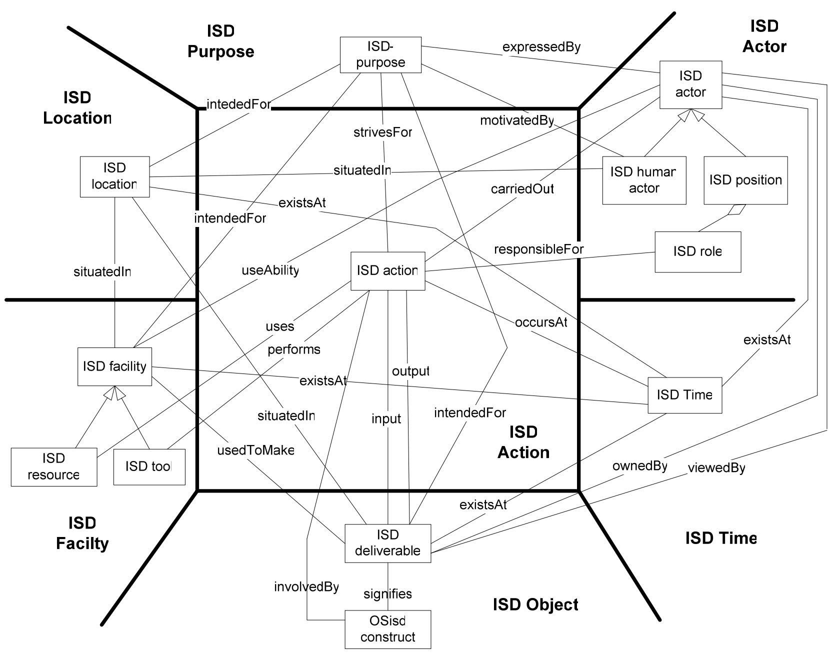

Fig. 5. ISD Inter-Domain Relationships

8 Discussions and Implications

In this paper we have presented a coherent, consistent and comprehensive conceptualization of ISD in the form of ISD ontology. Instead of giving preference to some narrow-scoped ISD approach, we have adopted an integrated view through which ISD is conceived as an aggregate of contexts. This view provides groundwork for the specification, analysis and integration of more specific views. Based on fundamental theories [10, 27, 8] with special interest in contextual phenomena, we have derived the contextual approach and the Seven S's Scheme composed of seven contextual domains. For four of these ISD domains we have defined concepts, relationships and constraints and presented them in meta models.

In the literature (e.g. [12, 36, 3]), a large variety of quality criteria are suggested for ontologies. Most commonly, these criteria concern clarity, consistency, coherence, comprehensiveness, accuracy, extendibility, and applicability. It is not possible here to consider the quality of the ISD ontology in terms of all these criteria. We can only say that by the selection of the contextual approach we have pursued to achieve a conceptualization that is natural and understandable. With the use of semi-formal meta models we have advanced the achievement and evaluation of clarity, consistency and coherency of our ontology. Comprehensiveness is relative to the needs for which an ontology has been built. Extendibility has been furthered by the use of a modular structure of the ontology.

The ultimate measure of the quality of an ontology is, naturally, its applicability. The ISD ontology is intended for descriptive, analytical and constructive use. In [24] we have deployed the ISD ontology to analyze and compare a set of existing frameworks, meta models and reference models for ISD and ISD methods [13, 15, 17, 18, 30, 33, 34]. The ISD ontology appeared to be a useful means in uncovering the orientation, emphases and limitations of these ISD artifacts in how they reflect contextual features of ISD, as well as in comparing the artifacts with one another. The analysis showed that the artifacts mostly lack theoretical backgrounds and they are narrow-scoped as to the collections of their contextual concepts. Our ISD ontology comprises a large array of concepts and constructs within four contextual domains, organized into flexible and easy-to-adapt structures. We have also deployed the ISD ontology as groundwork for engineering an ISD method ontology and a methodical skeleton for method engineering [24]. Also here the ISD ontology appeared to be helpful. It provided concepts and structures for specifying and elaborating the semantic content of an ISD method and for distinguishing and structuring approaches and actions of method engineering.

The ISD ontology is not without limitations. It should be enhanced with concepts and constructs of the ISD facility domain, the ISD location domain, and the ISD time domain. Many of the concepts included in the ontology can be further specialized to reflect more specific phenomena of the ISD. The set of constraints expressed by multiplicities in the meta models should be supplemented with more ISD specific constraints. The ISD ontology should also be employed in different kinds of situations to gain more evidence on its applicability. This is the way of validating the ontology.

In the research to come our aim is, besides completing the ontology, to apply it in the analysis of empirical ISD research and ISD approaches. For the former purpose, we have collected conceptual models underlying empirical studies on "how things are in ISD practice”. These models are, typically, quite specific which hinders to constitute an integrated understanding about results of the studies. The ISD ontology may serve as a coherent and comprehensive foundation to define, analyze and integrate conceptual models, in the way as an ontology for software maintenance [21] is suggested to be used. For the latter purpose, we will examine more closely ISD artifacts applying specific ISD approaches to find out how their commitments are visible in aggregates of concepts within each ISD domain.

References

1. Acuna, S. & Juristo, N. 2004. Assigning people to roles in software projects. Software – Practice and Experience, Vol. 34, No. 7, 675-696.
2. Baskerville, R. 1989. Logical controls specification: an approach to information systems security. In H. Klein & K. Kumar (Eds.) Proc. of the IFIP Working Conf. on Systems Development for Human Progress. Amsterdam: North-Holland, 241-255.
3. Burton-Jones, A., Storey, V., Sugumaran, V. & Ahluwalia, P. 2005. A semiotic metric suite for assessing the quality of ontologies. Data & Knowledge Engineering, Vol. 55, No. 1, 84-102.
4. Chandrasekaran, B., Josephson, J. & Benjamins, R. 1999. What are ontologies, and why do we need them? IEEE Intelligent Systems, Vol. 14, No. 1, 20-26.
5. Checkland P. 1988. Information systems and system thinking: time to unite? International Journal of Information Management. Vol. 8, No. 4, 239-248.
6. Curtis, B. & Kellner, M. & Over, J. 1992. Process modeling. Comm. of the ACM, Vol. 35, No. 9, 75-90.
7. Dowson, M. 1987. Iteration in the software process. In Proc. of the 9th Int. Conf. on Software Engineering. New York: ACM Press, 36-39.
8. Engeström, Y. 1987. Learning by expanding: an activity theoretical approach to developmental research. Helsinki: Orienta-Konsultit.
9. Fife, D. 1987. How to know a well-organized software project when you find one. In R. Thayer (Ed.) Tutorial: Software Engineering Project Management. Washington: IEEE Computer Society Press, 268-276.
10. Fillmore, C. 1968. The case for case. In E. Bach & R. T. Harms (Eds.) Universals in Linguistic Theory. New York: Holt, Rinehart and Winston, 1-88.
11. Gruber, T. 1993. A translation approach to portable ontology specification. Knowledge Acquisition, Vol. 5, No. 2, 119-220.
12. Gruber, T. 1995. Towards principles for the design of ontologies used for knowledge sharing. International Journal of Human-Coputer Studies, Vol. 43, No. 5/6, 907-928.
13. Harmsen, F. 1997. Situational method engineering. University of Twente, Moret Ernst & Young Management Consultants, The Netherlands, Dissertation Thesis.
14. Herbst, H. 1995. A meta-model for business rules in systems analysis. In J. Iivari, K. Lyytinen & M. Rossi (Eds.) Advanced Information Systems Engineering. LNCS 932, Berlin: Springer, 186-199.
15. Heym, M. & Österle, H. 1992. A reference model for information systems development. In K. Kendall, K. Lyytinen & J. DeGross (Eds.) Proc. of the IFIP WG 8.2 Working Conf. on the Impacts on Computer Supported Technologies on Information Systems Development. Amsterdam: North-Holland, 215-240.
16. IEEE 1990. Standard Glossary of software engineering terminology. IEEE Standard 610.12-1990.

17. Iivari, J. 1990. Hierarchical spiral model for information system and software development. Part 1: Theoretical background. Information and Software Technology, Vol. 32, No. 6, 386-399.
18. Iivari, J. 1990. Hierarchical spiral model for information system and software development. Part 2: Design process. Information and Software Technology, Vol. 32, No. 7, 450-458.
19. Jacobson, I., Booch, G. & Rumbaugh, J. 1999. The Unified Software Development Process. Reading: Addison-Wesley.
20. Kavakli, V. & Loucopoulos, P. 1999. Goal-driven business process analysis application in electricity deregulation. Information Systems, Vol. 24, No. 3, 187-207.
21. Kitchenham, B., Travassos, H., von Mayrhauser, A., Nielssink, F., Schneiderwind, N., Singer, J., Takada, S., Vehvilainen, R. & Yang, H. 1999. Towards an ontology of software maintenance. Journal of Software Maintenance: Research and Practice, Vol. 11, No. 6, 365-389.
22. Kruchten, P. 2000. The Rational Unified Process: An introduction. Reading: Addison-Wesley.
23. Lee, J., Xue, N.-L. & Kuo, J.-Y. 2001. Structuring requirement specifications with goals. Information and Software Technology, Vol. 43, No. 2, 121-135.
24. Leppänen, M. 2005. An ontological framework and a methodical skeleton for method engineering. Ph.D thesis, Jyväskylä Studies in Computing 52, University of Jyväskylä, Finland.
25. Leppänen, M. 2005. A context-based enterprise ontology. In G. Guizzardi & G. Wagner (Eds.) Proc. of Int. Workshop on Vocabularies, Ontologies, and Rules for the Enterprise (VORTE'05), Enchede, The Netherlands, 17-24.
26. Leppänen, M. 2005. Conceptual analysis of current ME artifacts in terms of coverage: A contextual approach. In J. Ralyté, Per Ågerfalk & N. Kraiem (Eds.) Proc. of the 1st Int. Workshop on Situational Requirements Engineering Processes (SREP'05), Paris, 75-90
27. Levinson, S. 1983. Pragmatics. London: Cambridge University Press.
28. Mathiassen, L. 1998. Reflective systems development. Scandinavian Journal of Information Systems, Vol. 10, No. 1/2, 67-117.
29. Mylopoulos, J., Chung, L., Liao, S. & Wang, H. 2001. Exploring alternatives during requirements analysis. IEEE Software, Vol. 18, No. 1, 92-96.
30. NATURE Team 1996. Defining visions in context: models, processes and tools for requirements engineering. Information Systems, Vol. 21, No. 6, 515-547.
31. Pohl, K. 1993. The three dimensions of requirements engineering. In C. Rolland, F. Bodart & C. Cauvet (Eds.) Proc. of the 5th Int. Conf. on Advanced Information Systems Engineering (CAiSE'93). LNCS 685, Berlin: Springer-Verlag, 275-292.
32. Sabherwal, R. & Robey, D. 1993. An empirical taxonomy of implementation processes based on sequences of events in information system development. Organization Science, Vol. 4, No. 4, 548-576.
33. Saeki, M., Iguchi, K., Wen-yin, K. & Shinokara M. 1993. A meta-model for representing software specification & design methods. In N. Prakash, C. Rolland & B. Pernici (Eds.) Proc. of the IFIP WG8.1 Working Conf. on Information Systems Development Process. Amsterdam: North-Holland, 149-166.
34. Song, X. & Osterweil, L. 1992. Towards objective, systematic design-method comparison. IEEE Software, Vol. 9, No. 3, 43-53.
35. Thayer, R. 1987. Software engineering project management – a top-down view. In R. Thayer (Ed.) Tutorial: Software Engineering Project Management. IEEE Computer Society Press, 15-56.
36. Uschold, M. 1996. Building ontologies: towards a unified methodology. In Proc. of 16th Annual Conf. of the British Computer Society Specialist Group on Expert Systems. Cambridge, UK.