Contextual Graphs for a Real-World Decision Support System

Johann V. Nguyen, Brian C. Becker, Avelino J. Gonzalez

Intelligent Systems Laboratory
University of Central Florida
Orlando, Florida
{johann, bbecker, gonzalez}@isl.ucf.edu
http://isl.ucf.edu

Abstract

Decision support systems that capture, preserve, and reuse implicit knowledge can greatly benefit from explicitly using context. The development of this type of system can benefit from a context-based knowledge representation paradigm to be effective in real-world applications. This paper describes how the implementation of the contextual graphs formalism in the AlexDSS system addresses the system's use of context. Contextual graph's explicit use of context combats the common user interaction problems of irrelevancy and redundancy that plague knowledge-based systems. The development of AlexDSS supports the notion that the contextual graph's formalism is a viable solution for a real-world decision support system. This paper discusses how contextual graphs are employed in decision support systems.

Introduction

This paper describes the implementation of Contextual Graphs (CxG) as the knowledge representation paradigm in the Alex Decision Support System (AlexDSS). Previous use of CxG had been focused on knowledge acquisition and organization (Brezillon 2003). AlexDSS offers insight into the effectiveness of CxG in a real-world decision support system. After providing an overview of the purpose of AlexDSS, we examine the role of context. Subsequent sections present the paradigms that had also been under consideration for the system, and illustrate the features of the CxG paradigm that make an attractive paradigm for AlexDSS. The final sections conclude with an examination of the use of CxG within AlexDSS.

Scope and Purpose of AlexDSS

The Industry/University Cooperative Research Center (I/UCRC) Program of the National Science Foundation (NSF) has been forming research and development unions between industry and academia for over 30 years. In anticipation of the retirement of its long time program manager, NSF funded the development and deployment of a decision support system that encompasses the director's accumulated knowledge. The result is AlexDSS, named after the I/UCRC program manager Dr. Alex Schwarzkopf. The purpose of AlexDSS is to capture, preserve, and reuse Schwarzkopf's expertise in the form of an interactive program. The targeted users include Schwarzkopf's successor as well as center directors that regularly seek guidance from him. The goal of the system is not only to advise the user but also to address the situations in a process similar to Dr. Schwarzkopf's.

Role of Context

The role of context can greatly influence how knowledge is organized and utilized in a knowledge-based system. Brezillon defines context from an engineer's position "as the collection of relevant conditions and surrounding influences that make a situation unique and comprehensible" (2001). This definition illustrates the potential value of context for the purpose of providing focus in a system. Context allows groupings of knowledge for specific situations, thus the system can avoid superfluous questions unrelated to the current context.

Knowledge to be Represented

AlexDSS is designed to provide guidance on a myriad of issues. Each issue itself offers a myriad of situations. The reasoning process that provides this guidance largely derives from Schwarzkopf's implicit knowledge. In order to use implicit knowledge, it must first be transferred into explicit knowledge, a process Nonaka calls *externalization* (1995). Nonaka believes that "It [externalization] is a quintessential knowledge-creation process in that tacit knowledge becomes explicit, taking the shapes of metaphors, analogies, concepts, hypothesis, or models" (Nonaka, 1995 p.64). This defines the goal of our effort, to create new knowledge in the form of CxG models from the implicit knowledge of an expert.

Pomerol and Brezillon presented the concept of *know how* and *know that* to characterize a relation between implicit and explicit knowledge (Pomerol and Brezillon, 2003). An example which they presented is the fact that one may know the mechanics and operations of a motorcycle (know that), but practice and experience is essential in order to successfully ride one (know how)

(Pomeral and Brezillon 2001). Similarly, knowing how to ride a motorcycle may not be sufficient to explain the operations of one. An important aspect to emphasize is that having one type of knowledge doesn't necessarily translate well into another type. This introduces the difficulty of externalization. The development of AlexDSS has revealed that when the know how knowledge is exchanged, it yields either insufficient explicit knowledge to form proper procedures or too much explicit knowledge that presents irrelevancy to the problem. Both of these problems have been addressed through incremental knowledge acquisition and representation in the CxG formalism discussed in the paper.

The Role of Context in AlexDSS

The utilization of context in a system depends on the perspective of data, information, knowledge, and context. The general relationship between data, information, and knowledge illustrates a process in which raw data becomes information through interpretation, which then is assimilated with knowledge for use in the decision-making process (Pomeral and Brezillon 2001). The understanding of those concepts solidifies their roles in the system. Data gathered from external sources translates into information. An agent then may use its knowledge about the information to either infer new information or employ in the decision-making process. However, the precise role of context varies depending on the specific implementation. The specific needs of AlexDSS helps in identifying the important role of context to be considered.

As mentioned earlier, knowledge acquired for AlexDSS is not easily converted into procedures. Dividing and categorizing the knowledge by relevance to a task provides a logical solution. Pomerol and Brezillon have concluded that context always relates to a specific task (2001). Gonzalez has also noted that the amount of possible events occurring is restricted by a given context (1998). The above concepts emphasize that during a given task, only a portion of a knowledge-base is needed to make a decision for a particular situation. The use of context in this manner isolates only the knowledge necessary for the task at hand. Likewise, the use of context to structure knowledge meaningfully offers a composition that more closely resembles how humans naturally organize. A problem arises if a piece of necessary knowledge to perform a task has not been included. Such an event is resolved by incorporating the missing knowledge into the context. A function to resolve this has been integrated into the CxG formalism discussion later in the paper.

The purpose of AlexDSS is to provide guidance through interaction with the user. Expert systems have been widely criticized as lacking common sense (Gonzalez and Dankel 1993), which results in irrelevant or redundant questions from the system. Such a flaw diminishes the likelihood that users will use the system. This is especially true for AlexDSS. The reason is the intended use of the system entails a broad range of tasks that may be nested. Such a complex organization is prone to irrelevancy and redundancy as described above.

Contextualizing each task resolves these issues. The result of this is that the context constrains the decision-making process without explicit intervention (Brezillon 2003). The introduction of constraints ensures that the system only queries the user for data in the context of the current task. This eliminates irrelevancy in a system that contains a wide ranging knowledge-base such as AlexDSS. Redundancy occurs when information that should have been previously inferred, was solicited from the user. Explicit declaration of context allows the system to assume attributes about the situation. This increases the system's capability to make intelligent decision on well defined circumstances.

Paradigms Considered

The selection of an appropriate knowledge representation paradigm to best address the problem is essential in avoiding a paradigm shift (Gonzalez and Dankel 1993). A paradigm shift occurs when the method of representing the knowledge is changed during the development cycle, causing a costly loss of time and effort. AlexDSS implements contextual graphs, but several other paradigms had been considered.

Non Contextually Explicit Paradigms

Rule-base and case-based reasoning were briefly considered for AlexDSS. The mechanics of these paradigms are not discussed here in detail but can be found respectively in (Gonzalez and Dankel 1993) and in (Kolodner 1993). Rule-base reasoning does have the potential to contextualize the knowledge base through clever implementation. However this imposes a significant amount of system design to mask its lack of explicitly using context to constrain the knowledge space. This makes a rule-base approach an unreasonable candidate. Case-based reasoning has a highly implicit contextual property built into the paradigm (Kolodner 1993), as well as a sufficient facility to learn. The premise of case-based reasoning resides in having an adequate case library of well documented cases. Unfortunately the AlexDSS system's knowledge is largely comprised of the implicit knowledge of a single expert instead of well-documented cases. This invalidates case-based reasoning as a candidate.

Contextually Explicit Paradigms

The CommonKADS Methodology had also been considered for AlexDSS as a complete system development process. The methodology has many promising attributes including its knowledge modeling exercises that provide contextual properties to the knowledge. This methodology is designed as a complete development cycle for large scale systems that model its reasoning from the interaction between multiple agents in

an organization. The development of the models necessary for CommonKADS is out of scope of the AlexDSS system. The system's reasoning largely consists of Schwarzkopf's implicit reasoning and does not consist of interaction with others. This disables many of the benefits of CommonKADS modeling process. More information on the mechanics of CommonKADS can be found in (Schreiber et al. 2000).

Contextual Graphs had been chosen largely for its explicit declaration of context. Context-base Reasoning (CxBR) also declares the context explicitly. The CxBR paradigm is typically employed to represent human behavior for autonomous agents in tactical situations. Context is represented explicitly as active environmental variables for which the agent reacts to and makes decisions accordingly. This reactive property benefits action-oriented systems that interact with a dynamic environment. This however, does not apply to AlexDSS. More information on context-based reasoning can be found in (Stensrud et al. 2004).

CxG Suitability for AlexDSS

The contextual graph formalism offers a method of representing knowledge in a structured approach based on tasks. The formalism explicitly organizes knowledge into task oriented contexts. As the name implies, CxG also provides a method to visually represent the knowledge using graphs similar to decision trees. The graphs offer a straightforward means to directly review the knowledge with agents not familiar with knowledge-base system design (Brezillon 2003). CxG addresses the contextualization needs of AlexDSS that were previously discussed.

Mechanics and Components of CxG

The contextual graphs formalism is a knowledge representation paradigm specifically developed to take context into account for decision-making (Brezillon 2003). Brezillon defines that "a contextual graph is an acyclic, directed graph with a unique input, [and a] unique output" (Brezillon 2003). CxG uses various nodes connected by directed arcs to constitute the knowledge representation. The contextual, recombination, action, and activity node symbolizes the various characteristics used in the decision-making process.

Contextual nodes are the decision nodes represented by a circular shape as seen on the left in Figure 1. These nodes have a single input but may have several diverging branches, each of which represents the different contexts that can be determined by the node. AlexDSS uses these nodes to solicit information on context from the user. Recombination nodes are represented with a smaller circle, as seen on the right in Figure 1. All branches from a contextual node eventually converge back to the same recombination node when the context no longer plays an active role. The pair of the contextual and recombination nodes forms a contextual element. Action nodes are represented with a square. The action node depicts the actions that are executed. AlexDSS uses action nodes to either notify users, build an output, or both. Activity nodes are also known as sub-graphs, which are represented with a rectangle. An activity node consists of a sequence of actions, or an entire contextual graph which may consist of any nodes including other activity nodes. These nodes represent tasks and sub-tasks in AlexDSS.

The careful arrangement of the nodes in a graph presents a functional model of the decision-making process (Brezillon 2003). The reasoning mechanism follows the direction of the graph and determines the context at each contextual node. This evaluation of context decides which of the divergent paths to pursue. The path of all the nodes visited yields a specific practice, which also can be used to trace the logic of the system. A more detailed explanation of contextual graphs can be found in (Brezillon 2003).

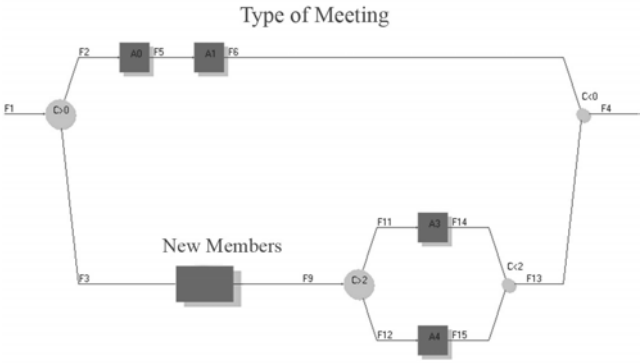

Figure 1: *Type of Meeting* contextual graph

Context and Knowledge in CxG

The identification of the role of context results from the distinction between procedures and practices. A procedure is an officially-defined process to solve a problem, and a practice is the process related to that problem, solved by an agent (human or automated) under a specific context. This is also the basis of how CxG incorporates new knowledge through a technique called incremental knowledge acquisition that is elaborated in the subsequent sub-section.

CxG divides context into three different categories: contextual knowledge, external knowledge, and proceduralized context. Contextual knowledge represents the knowledge that is used to constrain the decision-making process. All other knowledge at the beginning of the task is external knowledge, which is considered irrelevant to the problem. The contextual knowledge is represented by contextual nodes. When the contextual knowledge is instantiated, it transforms into a proceduralized context. This proceduralized context will de-proceduralize back into contextual knowledge.

Further consideration on the conversion between contextual knowledge and proceduralized context illustrates how CxG uses context. Each branch of a contextual element (a contextual and recombination node

pair) represents the possible alternative contexts for that element. The decision that is made at the contextual node determines the context that constrains the situation. Once the context is determined, the contextual knowledge now becomes a proceduralized context. The proceduralized context constrains the decision-making process by allowing only the subsequent actions, activities and contextual elements on its branch to be executed. The proceduralized context then reverts to contextual knowledge once the path has reached the recombination node of its element. This process illustrates the point when the context no longer has a constraining effect on the decision-making process (Brezillon 2002). The above process provides AlexDSS with the knowledge constraints needed to avoid irrelevancy.

Incremental Knowledge Acquisition

A contextual graph is initially developed to represent the established standard procedures. In reality, an agent adapts a procedure to accommodate the contexts of the situation to form a practice (Brezillon 2003). A practice can be considered the overall path that had been taken throughout the situation. The number of practices cannot be determined when the initial graph is built. The technique of incremental knowledge acquisition provides a method of incorporating new practices that had not been previously represented (Brezillon 2003). This technique is built upon the interaction between the user and the system. Once the user has solved a problem, the user can report the sequence of actions to the system. A discrepancy in the sequences of actions between the user and the system can possibly be viewed as a new practice. The logic between the system and the user is then compared to determine if any new contextual elements are needed to incorporate the new practice. This integration of new knowledge supports the characteristic of CxG that it may evolve to expand its knowledge base (Brezillon 2003). Likewise, in instances where insufficient knowledge had been originally included in a graph, this technique assists in adding additional knowledge to make the graph more complete.

CxG in AlexDSS

The developed AlexDSS program advises center directors on a number of different topics including intellectual property rights, funding issues, various meeting agendas, and important center policies. For example, one such topic is planning the itinerary for semiannual I/UCRC meeting. These meetings are typically held in different locations for multi-university centers. Any particular location could possibly host the meeting less than once a year, which implies a lack of experience for this task. The procedure of planning a meeting has been established, but each practice varies depending on a multitude of factors such as invited special guests, state of the center, number of projects, etc...

Implementation of CxG in AlexDSS

An object-oriented architecture provides a suitable approach that complements CxG's modular nature. The Java and Flash Technologies have been chosen for their multiplatform quality and flexible implementation, which is highly desirable for AlexDSS. Upon system completion, AlexDSS is expected to be distributed to over a hundred users via the Internet as an interactive webpage.

AlexDSS, similar to most modern knowledge-based systems, separates the knowledge and the reasoning mechanism (Gonzalez and Dankel 1993). To accomplish this, the graphs are designed with little inherent intelligence. The reasoning engine is described in further detail in later sections. The graphs are designed to be generic enough to represent any knowledge in the CxG formalism.

The nodes are represented by specialized classes that inherit from an abstract node class. This allows the reasoning engine to regard the nodes similarly during traversal. Each node contains a reference to its downstream node, except for contextual nodes which contain a list of downstream nodes for its branches. The nodes are arranged in groups of activities that are elaborated in the subsequent section. The graphs are stored using XML technology, which provides a straightforward method for editing.

Structure of the Knowledge Representation

The AlexDSS employs all of the CxG nodes that had been presented earlier. The activity node is heavily utilized as a means to organize the knowledge into activities (task in context). Every type of node, excluding the activity nodes, is a member of at least one activity node. Activity nodes, on the other hand, can be members of many other activity nodes or none. An activity node that doesn't belong to any other activity is considered to be a main activity. This type of grouping using activities adheres to the CxG formalism. Main activities may contain many nested sub-activities, which eventually create a hierarchal sequence of activities that maintain the directed acyclic nature of CxG (Brezillon 2003). An example of this hierarchy of nested activities is illustrated in figure 1 and figure 2. This activity is named *Type of Meeting*, which contains a sub-activity called *New Members*. The activity *New Members* is not only a member of the activity *Type of Meeting*, but also the main activity *Meeting Agenda* which is not illustrated here. Figure 1 depicts the *New Members* sub-activity as a node in the *Type of Meeting* activity. Figure 2 expands the activity to illustrate the sub-graph that the activity represents.

The use of activities to group modules of tasks together offers many benefits. First the logical grouping immediately gives the sense of context to all of the members in the activity. The grouping also allows the developer to build graphs on several layers. This increases readability. In figure 1, when the lower branch is taken in the first contextual node, one can see that the system enters the activity of *New Members*. Providing various levels of

abstraction promotes macro-management of the knowledge. Likewise, this also allows the same activity to be executed in several places if necessary.

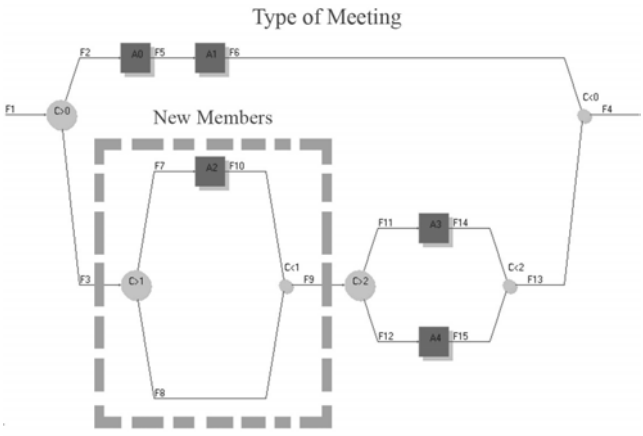

Figure 2: *Type of Meeting* with *New Members* activity expanded

Contextual Graph Selection

Because AlexDSS's knowledge varies widely, contextual graph selection becomes problematic. One user may be interested in IP rights and another may have a center policy question. A proposed solution was let a master graph ask a series of questions, focusing the context until the appropriate graph was identified. However, this method exhibited little intelligence. Instead, a search engine was employed, thereby allowing the user to ask a question of the system. Using an ontology as a linkage mechanism for common terms and phrases used in the domain, the search engine could match the user's query to the most appropriate graph. Additionally, based on the context, AlexDSS is capable of suggesting related graphs for further understanding of a topic, emulating Schwarzkopf's tendency to give auxiliary information about a question.

Reasoning Mechanism

The reasoning mechanism for the CxG formalism operated as expected in AlexDSS. Some special considerations had to be incorporated for the inclusion of the activity membership property as well as the nature of the output.

The reasoning engine utilizes CxG models and user interaction for the decision making process. Once a main activity is chosen, AlexDSS engine navigates through the graphs in it. When a contextual node is reached, the engine uses a graphical user interface (GUI) to solicit information from the user. The GUI then sends the engine the user's reply which the engine uses to process the next step. The system employs this interaction with the user to determine the context of the problem.

A CxG that doesn't contain activity nodes would traverse through the graph from one node to the next being directed by proceduralization of context when necessary, which continues until the end of the graph is reached. When an activity is introduced, traversal through that graph halts at the activity node. Traversal then takes place inside the activity until that activity is complete, after which time traversal in the parent graph is resumed. To effectively manage nested activities in a computer system, a last-in-first out stack had is used. This stack pushes an activity into the stack when active and pops the activity once it is done. This process is similar to proceduralization and de-proceduralization of context (Brezillon 2003).

The typical output of AlexDSS would consist of a practice to be followed. This practice does not always have a temporal relationship with the actions that build the practice. In the meeting agenda example, a typical action would have been "Reserve 15 min after lunch for the guest speaker's presentation". Such an output presented to the user immediately when that action node is executed may not have an intuitive bearing with the other actions. This is observed when actions that scheduled events after dinner had been executed from a previous proceduralized context. To effectively output a meaningful practice, the path of the main activity traversal had been maintained. This path retains the action sequence as well as all of the proceduralized context and the activities that had been executed. The path is specific to the main activity which would then present the practice to the user once the main activity had been accomplished.

AlexDSS Evaluation

The use of CxG in AlexDSS succeeds in organizing the knowledge, separating knowledge from reasoning, and addressing the issue of irrelevancy and redundancy. Through CxG, many independent graphs, each handling a specific situation, are organized into the knowledge base. Each graph is self-contained and, when executed by a CxG reasoner, reduces irrelevancy by gathering only information relevant to the situation at hand. Redundancy is avoided by the use of CxG's proceduralization.

Knowledge acquired directly from the expert had resulted in a transcript similar to a conversation about the topic. Analysis of the knowledge partitions it into contexts. These contexts not only organize the knowledge meaningfully, but also identify additional contexts. These additional contexts can be added to the graph or re-organized incrementally without disturbing the knowledge contained in the rest of the graph.

Contextual graphs enforce the separation of knowledge and the reasoning mechanism. This is especially useful for adding and altering the graphs. Once the reasoning engine had been developed, the developer's only concern is to build the graphs. AlexDSS uses XML to store the knowledge, enabling quick creation and revision. This had been the case when activity *New Members* had originally been excluded from activity *Type of Meeting*. Altering the knowledge in AlexDSS is accomplished quickly, leaving the reasoning engine unaltered.

The above example of moving the *New Members* activity into the *Type of Meeting* activity displays CxG's effective built-in property of reducing redundancy by separating commonly used nodes into activities to be

nested in other activities. Likewise, the explicit declaration of contexts in CxG through branches ensures that contextual elements and activities would only execute in the proper context. Irrelevant knowledge in context is easily identified in the graphs through path exploration and is easily removed.

The use of path tracking to build the results proved effective. Once the main activity had been accomplished, the system compiles information in the path to form an output. This is an important step because the practice includes many static events such as lunch, but their inclusion is required in the output. Depending on the activity completed, the final output may be represented with different documents or formatting. The output of the meeting agenda activity displays the practice as an itinerary in two forms. The simplified version lists the proceedings throughout the meeting. The descriptive version also adds an explanation to all proceedings including static events. This provides insight into the structure of a meeting that is normally provided by the expert. The itinerary closely resembles real-world itineraries approved by the expert.

Verification and Validation

AlexDSS is only as good as the user's perception of it. Thus, a significant effort has been expended to ensure system validity and usability. An integrated questionnaire presented to each user provides immediate feedback as to whether the system's output is applicable and correct. The questionnaire also inquires about other aspects of the system to maintain a friendly and easy-to-use interface.

Future Addition into the System

The AlexDSS's only concern is to preserve the knowledge of a single expert. The use of incremental knowledge acquisition to introduce new practices that a user has learned would invalidate the system's true intention. However incremental knowledge acquisition does provide a valuable tool for validation and learning. The majority of the knowledge introduced into the system would consist of real world and theoretical practices supplied by the user. Once enough practices have been assimilated to form a procedure, incremental knowledge acquisition could allow the expert to modify contextual elements of the system as needed. This is important since it follows the knowledge-base system lifecycle which allocates a large portion of time for verification, validation, and incremental addition of functional modules (Gonzalez and Dankel 1993).

Conclusion

The use of contextualized knowledge within a DSS exhibits promise in the development of effective real-world applications (Hiti et al. 1998). AlexDSS uses models of knowledge created from the process of externalization. The integration of context in these models requires a knowledge representation paradigm that explicitly utilizes context. These concepts demonstrate the inherent needs of AlexDSS that make the CxG approach ideal. CxG ensures the realization of AlexDSS's goals to capture, preserve, and reuse Schwarzkopf's expertise. CxG's technique of incremental knowledge acquisition assists the capture of new knowledge into the system. The use of graphs to organize knowledge components offers a capable means of preservation. The structure of CxG enforces the use of context explicitly, effectively reusing knowledge in an interactive manner and diminishing irrelevancy and redundancy. The AlexDSS presents evidence that these unique features of CxG provide a catalyst to facilitate the implementation of a real-world DSS.

Copyright © 2006, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

References

1. Brezillon, P. Context Dynamic and Explanation in Contextual Graphs. CONTEXT 2003, LNAI 2680(2003)94-106
2. Brezillon, P. Context-based Modeling of Operators' Practices by Contextual Graphs. Human Centered Process: 14th Mini Euro Conference, Luxembourg (2003)
3. Brezillon, P. Using Context for Supporting Users Efficiently. Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS '03) , IEEE (2002)
4. Brezillon, P. Representation of Procedures and Practices in Contextual Graphs. The Knowledge Engineering Review Vol. 18 Issue 2. Cambridge University Press (2003)
5. Gonzalez, A. J., Dankel, D. D. *The Engineering of Knowledge-Based Systems: Theory and Practice*. Prentice Hall, Englewood Cliffs, NJ (1993)
6. Gonzalez, A. J., Ahlers, R. H. Context-Based Representation of Intelligent Behavior in Training Simulations. Naval Air Warfare Center Training Systems Division Conference (1998)
7. Hiti, I., Cestnik, B., Selan, F., Janezic, J. Analysing Context in DSS for Optimal Selection of Telecommunication Services and Technologies for Business Support. *Context Sensitive Decision Support Systems*. Chapman and Hall (1998)
8. Kolodner, J. *Case-Based Reasoning*. Morgan Kaufmann Publishers, Inc., San Mateo, CA (1993)
9. Nonaka, I., Takeuchi, H. *The Knowledge-Creating Company*. Oxford University Press, New York, NY (1995)
10. Pomerol, J.-Ch, Brezillon, P. About some relationships between Knowledge and Context. CONTEXT-01, Lecture Notes in Computer Science. Springer Verlag (2001)
11. Schreiber, G., Akkermans, H., Anjewierden, A., deHoog, R., Shadbolt, N., VandeVelde, W., Wielinga, B. *Knowledge Engineering and Management: The CommonKADS Methodology*. MIT Press, Cambridge, MA (2000)
12. Stensrud, B. S., Barrett G. C., Trinh, V. C., Gonzalez, A. J. Context-Based Reasoning: A Revised Specification. FLAIRS Conference (2004)