



## LarKC

*The Large Knowledge Collider*  
*a platform for large scale integrated reasoning and Web-search*

**FP7 – 215535**

---

### D1.1.1

## An Overview of Relevant Work in Other Areas

---

**Elena Simperl, Uwe Keller, Florian Fischer, Eyal Oren, Barry Bishop, Zhisheng Huang, Gaston Tagni, Jose Quesada, Blaz Fortuna, Jia Hu, Yulin Qin**

<b>Document Identifier:</b>	LarKC/2008/D1.1.1
<b>Class Deliverable:</b>	LarKC EU-IST-2008-215535
<b>Version:</b>	version 1.3
<b>Date:</b>	September 25, 2008
<b>State:</b>	final
<b>Distribution:</b>	public



## EXECUTIVE SUMMARY

The main assumption taken in LarKC is that a distributed Semantic Web reasoning infrastructure has to go beyond current reasoning paradigms that are strictly based on logic in order to scale to the size of the current and future Web. In LarKC this paradigm shift will be obtained by fusing reasoning (in the sense of logic) with search (in the sense of information retrieval), and taking seriously the notion of limited rationality (in the sense of Herbert Simon). The reasoning approach followed in LarKC will be fundamentally different and highly innovative. It will combine inter-disciplinary problem solving techniques (inductive and deductive, Boolean and statistical, incomplete reasoning and limited rationality, quantum logics etc), which are otherwise being studied and used independently, with methods from information retrieval, economics, cognition, brain science, cognitive and social psychology in an effective and efficient manner. This deliverable gives an overview of these areas and disciplines and motivates their relevance to LarKC.



## DOCUMENT INFORMATION

<b>IST Project Number</b>	FP7 – 215535	<b>Acronym</b>	LarKC
<b>Full Title</b>	Large Knowledge Collider		
<b>Project URL</b>	<a href="http://www.larkc.eu/">http://www.larkc.eu/</a>		
<b>Document URL</b>			
<b>EU Project Officer</b>	Stefano Bertolo		

<b>Deliverable</b>	<b>Number</b>	1.1.1	<b>Title</b>	An Overview of Relevant Work in Other Areas
<b>Work Package</b>	<b>Number</b>	1	<b>Title</b>	Conceptual Framework and Evaluation

<b>Date of Delivery</b>	<b>Contractual</b>	M6	<b>Actual</b>	30-Sept-08
<b>Status</b>	version 1.3		final	<input checked="" type="checkbox"/>
<b>Nature</b>	prototype <input type="checkbox"/> report <input checked="" type="checkbox"/> dissemination <input type="checkbox"/>			
<b>Dissemination Level</b>	public <input type="checkbox"/> consortium <input checked="" type="checkbox"/>			

<b>Authors (Partner)</b>	Elena Simperl (UIBK), Uwe Keller (UIBK), Florian Fischer (UIBK), Eyal Oren (VUA), Barry Bishop (UIBK), Zhisheng Huang (VUA), Gaston Tagni (VUA), Jose Quesada (MPI), Blaz Fortuna (CycEur), Jia Hu (WICI) and Yulin Qin (WICi)			
<b>Resp. Author</b>	Elena Simperl (UIBK)		<b>E-mail</b>	elena.simperl@sti2.at
	<b>Partner</b>	UIBK	<b>Phone</b>	+43 (512) 507 96884

<b>Abstract (for dissemination)</b>	The main assumption taken in LarKC is that a distributed Semantic Web reasoning infrastructure has to go beyond current reasoning paradigms that are strictly based on logic in order to scale to the size of the current and future Web. In LarKC this paradigm shift will be obtained by fusing reasoning (in the sense of logic) with search (in the sense of information retrieval), and taking seriously the notion of limited rationality (in the sense of Herbert Simon). The reasoning approach followed in LarKC will be fundamentally different and highly innovative. It will combine inter-disciplinary problem solving techniques (inductive and deductive, Boolean and statistical, incomplete reasoning and limited rationality, quantum logics etc), which are otherwise being studied and used independently, with methods from information retrieval, economics, cognition, brain science, cognitive and social psychology in an effective and efficient manner. This deliverable gives an overview of these areas and disciplines and motivates their relevance to LarKC.
<b>Keywords</b>	economics, bounded rationality, cognitive theories, brain science, patterns

Version Log			
Issue Date	Rev No.	Author	Change

## PROJECT CONSORTIUM INFORMATION

Partner	Acronym	Contact
Semantic Technology Institute Innsbruck <a href="http://www.sti-innsbruck.at">http://www.sti-innsbruck.at</a>	STI 	Prof. Dr. Dieter Fensel Semantic Technology Institute (STI) Innsbruck, Austria E-mail: dieter.fensel@sit2.at
AstraZeneca AB <a href="http://www.astrazeneca.com/">http://www.astrazeneca.com/</a>	ASTRAZENECA 	Bosse Andersson AstraZeneca Lund, Sweden E-mail: bo.h.andersson@astrazeneca.com
CEFRIEL SCRL. <a href="http://www.cefriel.it/">http://www.cefriel.it/</a>	CEFRIEL 	Emanuele Della Valle CEFRIEL SCRL. Milano, Italy E-mail: emanuele.dellavalle@cefriel.it
CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O. <a href="http://cyceurope.com/">http://cyceurope.com/</a>	CYCORP 	Michael Witbrock CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O., Ljubljana, Slovenia E-mail: witbrock@cyc.com
Höchstleistungsrechenzentrum, Universität Stuttgart <a href="http://www.hlrs.de/">http://www.hlrs.de/</a>	HLRS 	Dr. Georgina Gallizo Höchstleistungsrechenzentrum, Universität Stuttgart Stuttgart, Germany E-mail : gallizo@hlrs.de
Max-Planck-Institut für Bildungsforschung <a href="http://www.mpib-berlin.mpg.de/index_js.en.htm">http://www.mpib-berlin.mpg.de/index_js.en.htm</a>	MAXPLANCKGESELLSCH 	Dr. Lael Schooler, Max-Planck-Institut für Bildungsforschung Berlin, Germany E-mail: schooler@mpib-berlin.mpg.de
Ontotext Lab, Sirma Group Corp. <a href="http://www.ontotext.com/">http://www.ontotext.com/</a>	ONTO 	Atanas Kiryakov, Ontotext Lab, Sirma Group Corp. Sofia, Bulgaria E-mail: atanas.kiryakov@sirma.bg
SALT LUX INC. <a href="http://www.saltlux.com/EN/main.asp">http://www.saltlux.com/EN/main.asp</a>	Saltlux 	Kono Kim SALT LUX INC Seoul, Korea E-mail: kono@saltlux.com
SIEMENS AKTIENGESELLSCHAFT <a href="http://www.siemens.de/">http://www.siemens.de/</a>	Siemens <b>SIEMENS</b>	Dr. Volker Tresp SIEMENS AKTIENGESELLSCHAFT München, Germany E-mail: volker.tresp@siemens.com
THE UNIVERSITY OF SHEFFIELD <a href="http://www.shef.ac.uk/">http://www.shef.ac.uk/</a>	Sheffield 	Prof. Dr. Hamish Cunningham THE UNIVERSITY OF SHEFFIELD Sheffield, UK E-mail: h.cunningham@dcs.shef.ac.uk
VRIJE UNIVERSITEIT AMSTERDAM <a href="http://www.vu.nl/">http://www.vu.nl/</a>	Amsterdam 	Prof. Dr. Frank van Harmelen VRIJE UNIVERSITEIT AMSTERDAM Amsterdam, Netherlands E-mail: Frank.van.Harmelen@cs.vu.nl
THE INTERNATIONAL WIC INSTITUTE, BEIJING UNIVERSITY OF TECHNOLOGY <a href="http://www.iwici.org/">http://www.iwici.org/</a>	WICI 	Prof. Dr. Ning Zhong THE INTERNATIONAL WIC INSTITUTE Mabeshi, Japan E-mail: zhong@maebashi-it.ac.jp

<p>INTERNATIONAL AGENCY FOR RE- SEARCH ON CANCER <a href="http://www.iarc.fr/">http://www.iarc.fr/</a></p>	<p>IARC2</p> 	<p>Dr. Paul Brennan INTERNATIONAL AGENCY FOR RE- SEARCH ON CANCER Lyon, France E-mail: <a href="mailto:brennan@iarc.fr">brennan@iarc.fr</a></p>
--	--	---

---

## TABLE OF CONTENTS

1	INTRODUCTION	1
2	APPROXIMATION THEORY	3
2.1	Case-based Reasoning . . . . .	3
2.2	Probabilistic Reasoning . . . . .	3
2.3	Granular Reasoning . . . . .	4
2.4	Relevance to LarKC . . . . .	5
3	QUANTUM LOGICS	6
3.1	Relevance to LarKC . . . . .	6
4	META-REASONING	8
4.1	Relevance to LarKC . . . . .	9
5	COGNITIVE ARCHITECTURES	11
5.1	SOAR . . . . .	11
5.2	ACT-R . . . . .	12
5.3	Detractors . . . . .	12
5.4	Relevance to LarKC . . . . .	13
6	SIMILARITY THEORIES	14
6.1	The Trouble with Similarity . . . . .	14
6.2	Metric Models . . . . .	14
6.3	Feature Models . . . . .	14
6.4	Structural Mapping Models . . . . .	15
6.5	Transformation Distance Models . . . . .	16
6.6	Relevance to LarKC . . . . .	16
7	BRAIN SCIENCE/NEUROSCIENCE	18
7.1	Relevance to LarKC . . . . .	18
8	ECONOMICS	20
8.1	Supply-Chain Management . . . . .	20
8.2	Cost-Benefit Analysis . . . . .	20
8.3	Risk Management . . . . .	20
8.4	Relevance to LarKC . . . . .	21
9	BOUNDED RATIONALITY	23
9.1	Relevance to LarKC . . . . .	23
10	PATTERNS AND PATTERN LANGUAGES	25
10.1	Core Concepts . . . . .	25
10.2	Relevance to LarKC . . . . .	26
11	DISTRIBUTED AND PARALLEL COMPUTING	27
11.1	Distributed Systems and Distributed Computing . . . . .	27
11.2	Parallel Computing . . . . .	28



11.3 Relevance to LarKC . . . . .	28
12 SOFTWARE ENGINEERING	29
12.1 Waterfall Model . . . . .	29
12.2 Spiral Model . . . . .	30
12.3 Agile Software Development . . . . .	30
12.4 Relevance to LarKC . . . . .	31
13 CONCLUSIONS AND FUTURE WORK	32

# 1 INTRODUCTION

This deliverable provides a high-level survey of a number of areas of Computer Science, and beyond, which address problems relevant for the overall goal of the LarKC project, the development of a distributed Semantic Web reasoning platform which can scale to the size of the current and future Web. The main assumption taken is that such an infrastructure must go beyond current reasoning paradigms that are strictly based on logic. By fusing reasoning (in the sense of logic) with search (in the sense of information retrieval), and taking seriously the notion of limited rationality (in the sense of Herbert Simon) LarKC will realize the paradigm shift that is required for reasoning at Web scale. The reasoning approach followed in LarKC will be fundamentally different and highly innovative. It will combine inter-disciplinary problem solving techniques (inductive and deductive, Boolean and statistical, incomplete reasoning and limited rationality, quantum logics etc), which are otherwise being studied and used independently, with methods from information retrieval, economics, cognition, brain science, cognitive and social psychology in an effective and efficient manner. The conceptual foundations of this very ambitious work are laid out in work package WP1.

The objective of WP1 is the conceptual integration of the different models of reasoning offered by many of the disciplines previously mentioned. In order to tackle this integration work package WP1 will

- build bridges to relevant work in these disciplines that may provide either single means or patterns of integration of these means.
- define algorithmic schemata as operational instruments for integrating complementary techniques.
- reflect our integration of inductive and deductive techniques into a coherent framework at the meta-scientific level.
- derive a paradigm of heuristic problem solving from this together with means for evaluation and measurement of utility.

This deliverable addresses the first of the four challenges listed above. It gives an overview of areas and disciplines which might provide interesting insights, approaches, or techniques which are potentially applicable to LarKC and its use cases. The selection of the relevant areas and disciplines is the result of several discussion rounds within the consortium, notably among the partners contributing to this deliverable (UIBK, VUA, CycEur, MPI, WICI). In a first round of discussions a list of areas and disciplines to be further looked into was compiled. This list, deliberately non-restrictive, included all suggestions made by the participants during the first round of discussions. The contributing partners created a summary (of several pages) of the topics selected, which covered an introduction to topic, the most important concepts, notions and principles, the main areas of applications, as well as a motivation for further consideration of this topic in the context of LarKC. This input was evaluated in a second round of discussions with respect to its potential impact on LarKC, resulting in a second list of topics, which are presented in this deliverable. They provide a starting point for an in-depth, systematic analysis in order to identify bridges to relevant work, methods,

techniques, and integration patterns, which are applicable to LarKC. Deliverable D1.1.2 will report on the results of this analysis.

The rest of this deliverable is organized as follows. First we look into several reasoning areas which are relevant for both the technical work and the conceptual grounding of the operational integration framework to be defined in this work package. Then we discuss cognitive architectures, similarity theories as well as the basics of brain science and neuroscience. These areas have a longstanding tradition in studying reasoning processes and mechanisms of humans, known for their astonishing ability to cope with high amounts of heterogenous, incomplete, and rapidly changing information. A third part of the deliverable considers fundamental aspects of economics, notably the theories which have studied means to analyze trade-offs, costs and benefits of an artifact, or situation. Such theories could be applied to the definition of the evaluation framework of LarKC, and motivate new techniques to assess the quality of reasoning results and to define strategies for heuristics-based reasoning pipelines. Then we give a short insight into the theory of patterns and pattern languages, originating from architectural design. Patterns are important to LarKC not only from an engineering perspective, but also as fundamental means to define a conceptual integration framework and to operationalize it. The last two areas surveyed (distributed and parallel computing, software engineering) are relevant for the realization of the LarKC platform. We conclude with a summary of the most important findings and an outline of the future work to be carried out in task T1.1.

## 2 APPROXIMATION THEORY

Approximation theory is an important component of Web-scale reasoning and problem solving. Instead of searching for a computationally expensive, optimal solution, we are, more often than not, happy with approximate, sub-optimal solutions, which provide a feasible trade-off between computational costs and level of quality.

Approximation theory relies on so-called "approximate algorithms", which aim to find a nearly optimal solution to a specific problem that cannot be solved exactly within a reasonable time. In this context approximate reasoning implements algorithms that draw uncertain conclusions from a set of uncertain premises. Many of these algorithms, including the probabilistic, fuzzy, non-monotonic, case-based or qualitative reasoning, are fundamental to human thinking and behavior. This chapter gives an overview of three of these classes of algorithms.

### 2.1 Case-based Reasoning

Case-based reasoning (CBR) is a core methodology in the field of Artificial Intelligence. The basic idea of CBR is to tackle new problems by referring to similar problems that have already been solved in the past, more precisely, by resorting to individual experiences in the form of cases. The generalization beyond these experiences typically relies on a kind of regularity assumption which can be formulated as "similar problems have similar solutions."

CBR uses different forms of approximate reasoning and reasoning under uncertainty, notably probabilistic and fuzzy set-based techniques. Hullermeier elaborated on the relationship between the two areas in particular with respect to the development of novel approaches and hybrid systems [37].

### 2.2 Probabilistic Reasoning

A randomized algorithm or probabilistic algorithm is an algorithm which employs a degree of randomness as part of its logic. In common practice, this means that the machine implementing the algorithm has access to a pseudo-random number generator. The algorithm typically uses random bits as an auxiliary input to guide its behavior, in the hope of achieving good performance in the "average case". Formally, the performance or quality of the algorithm is a random variable determined by these random bits, with adequate expected value. The expected value is called the expected running time. The algorithm is designed in such a way that it minimizes the probability that the "worst case" occurs to a level that such events can be ignored.

The probably approximately correct (PAC) learning model is fundamental to computational learning theory [66]. It provides a probabilistic framework for the study

of learning and generalization. The learner receives samples and must select a generalization function (called the “hypothesis”) from a certain class of possible functions. The goal is that, with high probability (the “probably” part), the selected function will have low generalization error (the “approximately correct” part). The learner must be able to learn the concept given any arbitrary approximation ratio, probability of success, or distribution of the samples. An important innovation of the PAC framework is the introduction of complexity theory concepts to machine learning. In particular, the learner is expected to find efficient functions (time and space requirements bounded to a polynomial of the example size). In addition, the learner itself must implement an efficient procedure (requiring an example count bounded to a polynomial of the concept size, modified by the approximation and likelihood bounds).

## 2.3 Granular Reasoning

Granular computing deals with reasoning and information processing with multi-level hierarchical structures [70, 68, 69]. Data, information, and knowledge are arranged in multiple levels according to their granularity. A higher level contains more abstract or general knowledge, whilst a lower level contains more detailed or specific knowledge. Reasoning can be performed on various levels. Results from a higher part of the hierarchy may be more imprecise but can be obtained faster or with less computational costs. By contrast, moving to a lower level in the hierarchy increases the quality of the results, and implicitly, the resources which are required to obtain them. In this way, granular computing offers a multiple-resolution reasoning scheme. One may choose a proper level of granularity to draw a desirable conclusion under certain resource constraints. In fact, such a reasoning scheme is commonly used by human for practical and real-time decision-making.

Variable precision logic may be viewed as a logic implementation of the general ideas of granular computing [42]. It concerns reasoning with incomplete information under time constraints. More specifically, it is based on a trade-off between precision of inferences and the computational cost to derive them. The building block of the variable precision logic is a rule of the form “if A then B unless C”, where C is the exception, A and B represent larger granules and C represents a smaller granule. If there is no sufficient time, one can simply draw conclusion B if A holds, although when condition C also holds, one cannot conclude A. If more time is available, one can examine condition C and draw new conclusions. In general, one can associate measures to such rules in order to perform quantitative inference.

Ripple-down rules offer a slightly different organization of rules [21]. A rule is typically associated with another rule of exceptions, this rule may also have another rule of further exceptions, and so on. Ripple-down rules can be used to organize general and not 100% correct knowledge with another modification rule in a similar manner. In this way the approach provides a reasoning method similar to variable precision logic. One can stop the reasoning process in the middle of a ripple-down rule, if time constraints must be met. Although the conclusion drawn may

be incorrect, the structure of ripple-down rules leads to a minimal number of errors.

## 2.4 Relevance to LarKC

Reasoning in LarKC must consider multiple strategies. Depending on special requirements or constraints, one may choose a particular strategy as given above. The use of multiple strategies may speed up the reasoning process. Web reasoning also needs to combine rule-based reasoning with case-based reasoning. Web information sources cannot be captured as knowledge (rules), but only in form of cases, due to the complexity and variety of Web information and Web service provisioning. The experience gained from the cases can also be used to improve the modeling and implicitly the rules. Based on this observation, we can develop hybrid reasoning methods on the Web information at multiple levels. These issues will be addressed at a conceptual level in work package WP1 (Conceptual Framework and Evaluation) at a technical level in work package WP4 (Reasoning and Deciding).

### 3 QUANTUM LOGICS

Quantum logics are non-classical logics arising from the mathematical formalism of quantum theory. They were first proposed by Birkhoff and von Neumann in their article entitled “The logic of quantum mechanics” in 1936 [12].

Quantum logics use the concept of a mathematical “phase-space” from quantum theory, which originates from classical mechanics and classical electrodynamics. Any physical system  $S$  is at each instant hypothetically considered to be associated with a point in a fixed phase-space  $\Sigma$ . This point is supposed to represent mathematically the state of  $S$ . Moreover, the state of  $S$  is supposed to be ascertainable by maximal observation. Maximal pieces of information about physical systems are called “pure states”. They can be represented as a sequence of real numbers  $\langle r_1, \dots, r_n \rangle$ . Thus, the phase-space can be identified with the set of  $n$ -tuples of real numbers  $\mathbf{R}^n$ .

In quantum logics, a proposition  $p$  is associated with a subset  $X$  (alternatively called an event) of the phase-space, which consists of all the pure states for which  $p$  hold. For an event  $X$  and a pure state  $s$ , we say that the system in a state  $s$  verifies both  $X$  and the proposition  $p$  if and only if pure state  $s$  is a member of the event  $X$ .

In quantum theory, pure states assign only probability values to quantum events. For a quantum system  $S$  and a proposition  $p$ , we say that  $p$  is true in the system  $S$  if  $S$  assigns to  $p$  probability-value 1, written  $S(p) = 1$ , and  $p$  is false in the system  $S$  if  $S$  assigns to  $p$  probability-value 0, written  $S(p) = 0$ .

Quantum logics allow us to define the non-classical notion of negation. The orthogonal complement  $X'$  of the mathematical representative  $X$  of a proposition  $p$  is defined as the set of all vectors that are orthogonal to all elements of  $X$ . Namely,  $\chi \in X'$  iff  $\chi \perp X$ , which denote that for any  $\phi \in X$ , the inner product of  $\chi$  and  $\phi$  is 0. Variants of conditionals and implications, such as positive conditionals, implications-connectives, polynomial conditionals, counterfactual conditionals, strict-implication, are also supported [20].

An important application of quantum logics is the “Theories of Quasiset”, first developed by Takeuti in 1981 [63]. The basic aim there was to provide a mathematical description for a collection of micro-objects, which can avoid the counter-intuitive properties of the classical identity relation, such as the problem of “extensional equality” with the sense that “two quantum sets that are extensionally equal do not necessarily share all the same properties”. Avoidance of this Leibniz-substitutivity principle may find interesting applications in intensional logics.

#### 3.1 Relevance to LarKC

For many applications on the Semantic Web, in particular for those for LarKC, we are interested in non-classical logics which may be used to deal with large-scale semantic data. In particular, non-classical negations in the sense of an orthogonal complement may provide an intuitive interpretation of negation in ontologies

and Description Logics. The extensional equality and the Leibniz-substitutivity principle are also highly relevant to various reasoning requirements of typical semantic systems, such as closed world vs. open world assumption, unique name assumption, and so on.

The semantic space model at the core of work package WP2 (Retrieval and Selection) shares some commonalities with quantum mechanics, i.e., those of quantum logics, in which vector spaces stand in for the neuro-physics and mechanisms such as Hilbert spaces and uncertainty are investigated as vehicles for computation.

## 4 META-REASONING

A number of years ago, the research community on Artificial Intelligence has put substantial efforts in investigating a particular aspect of reasoning that lies at the heart of human intelligence and problem solving behavior, that is, the idea that much can be gained in problem solving by thinking of one’s own behavior during the problem solving process. In the following, we consider a simple conceptual model for meta-reasoning proposed in [25].

In cognitive science and AI, thinking (or reasoning) was understood as a central component of the action-perception cycle. Assume an (intelligent) agent which acts within some environment to achieve a certain goal. The agent observes the environment at any moment in time and perceives some stimuli. Based on the stimuli, the agent behaves rationally to achieve its ultimate goal by selecting a suitable action from its repertoire of competencies. The identification of the selected action is based on reasoning. The resulting effect on the environment at the ground level is subsequently observed by the agent at the object level and the process iterates. This problem-solving cycle is shown in Fig. 4.1.

Reasoning is used in this cycle at the object level to perform better problem solving at the ground level. Therefore, we can apply the same principle to the object level by considering it as the ground level of a different (reasoning) environment: “Meta-reasoning” is the then the process of reasoning about the (lower-level) reasoning cycle. It consists of two basic components:

- The *meta-level control* of the reasoning process itself.
- The *introspective monitoring* of the reasoning process at the object level.

In other words, meta-reasoning can be understood as the perception and control of the reasoning process and forms itself a (meta-level) problem solving cycle on top of the object-level reasoning cycle. This extended problem solving process including meta-reasoning is shown in Fig. 4.2.

The goal of the meta-level control is to improve the quality of the decisions taken at the object level by spending some effort to decide what and how much reasoning to do at the object level. By doing so, balancing resources between the object level computations and ground level actions shall be achieved. In essence, meta-level control allows the agent to dynamically adapt or change the object-level computation performed (i.e., the reasoning strategy). Clearly, there is a trade-off between the (continuous) meta-level reasoning and ground level performance,

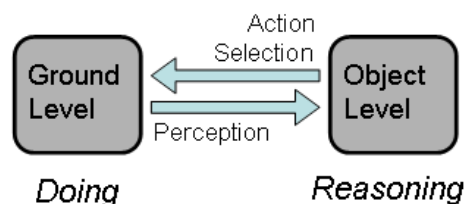


Figure 4.1: Reasoning as the Component of Problem-solving Process

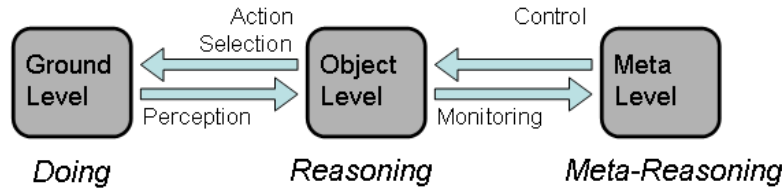


Figure 4.2: Problem-solving Process using Meta-Reasoning

since the former can interfere with the ground level actions. Therefore, the main decision points where meta-level control is important and can be useful need to be identified if we consider resource-bounded agents.

Introspective monitoring allows to collect sufficient data about the reasoning process to take informed decisions on how to adapt the it to improve or maximize performance at the ground level. This might involve recording data on the computational performance to build characteristic profiles for various reasoning strategies. Techniques from data mining and machine learning can be applied here to turn the measured observations into effective decision heuristics. Introspective monitoring might also involve generating explanations for decisions at the object level and their observed effect on the ground level performance. In case of failure to complete a reasoning task, it may involve explanations to the causes of the failure and a respective diagnosis of the object-level reasoning process.

Meta-reasoning was a very active field of research within the AI community nearly two decades ago. In recent years, it seems that research in meta-reasoning did not receive comparable attention. Interestingly, the topics seems to be revived: this year, there has been a dedicated workshop on the topic at the major forum for AI research world-wide, i.e., the AAI conference.<sup>1</sup>

A recent survey is given in [23, 25]. A foundational article on the topic is [53]. The aspect of limited rationality is discussed in depth in [54]. [55] specifically reviews approaches to the control of computation in resource-bounded agents. [71] explains why resource-bounded reasoning is a fundamental research direction within AI and presents a productive methodology to explore this field. [49] considers a classification of different form of meta-reasoning and identifies a specific form of meta-reasoning (i.e., post-meta-reasoning) which can specifically be supported by concurrent and parallel computation infrastructures.

## 4.1 Relevance to LarKC

Meta-reasoning can be seen as conceptual tool to deal with scalability issues in problem solving. Problem solving can be seen as a large search problem, in which an agent tries to find the right sequence of actions to be carried out in order to achieve a certain goal. Using meta-reasoning, instead of trying out what to do next, the agent does not only take into account the information of the environment

<sup>1</sup><http://www.aaai.org/Library/Workshops/ws08-07.php>

it acts upon, but also observations on the problem solving process performed so far (or even similar problem solving processes that have been performed in the past). The observations on the evolution of the problem solving process are used to select actions that are likely to be useful for finding or constructing a solution in a specific situation, or to stop performing actions which will probably not lead to the desired goal. Therefore, a problem solving system is likely to suffer from severe performance problems, if there are too many actions that can be tried at any moment in time.

Reasoning is a specific example of a (class of) problem solving tasks: the actions which are performed during the reasoning process are (simple) inferences. The environment that is acted upon is a knowledge space (i.e., explicitly given knowledge, or data, and knowledge derived by inferences during the reasoning process). The number of possible inferences during the construction of a proof is heavily dependent on the size of the knowledge space that is considered during reasoning. In the context of LarKC precisely the aspect of the size of the knowledge base is crucial as the platform is supposed to reason over a huge amount of data. Questions such as “What part of the data is relevant or especially useful to consider at a specific moment in time during the reasoning process? What data shall we leave out of consideration during some period? What shall we focus on? When is it a “good” moment in time to refocus our current (unsuccessful) reasoning process on different parts of the knowledge base or to use different methods to perform a certain inference (subproblem)?” are in fact meta-reasoning problems. A successful (and efficient) solution to such problems might be one of the key ingredients to enable the reasoning platform to deal with knowledge bases that are multiple orders of magnitude larger than the challenging benchmarks that inference tools have to face already today in real-world applications.

Therefore, meta-reasoning seems to be a promising and conceptually elegant tool for scaling up the reasoning capabilities of existing high-performance reasoners to Web-scale knowledge bases.

## 5 COGNITIVE ARCHITECTURES

Cognitive architectures are a means to cope with a particular problem experienced by the behavioral sciences, and psychology in particular. In these disciplines it is common to divide the object of study (mind) into a potentially large number of specialized sub-components. This approach, followed exhaustively, leads to situations in which assembling individual contributions to an overall solution becomes nearly impossible. In other words, as stated by Newell in [44], you cannot play 20 questions with nature and win. Allen Newell’s book, “Unified Theories of Cognition” [45] advanced the physical symbol system (PSS) hypothesis: A PSS takes physical patterns (symbols), combining them into structures (expressions) and manipulating them (using processes) to produce new expressions. All fields of study are sub-classes of this PSS. But the more radical idea is that “A physical symbol system has the necessary and sufficient means for general intelligent action.” That implies that both humans and machine are substantially alike in what they do to produce intelligent behavior, and that machines can reach human-level intelligence. These are the ground assumptions of cognitive psychology and AI, respectively.

The pragmatic advantage of cognitive architectures is that users can create and share models using the same language. These models are run by the interpreter of the cognitive architecture. The interpreter includes known cognitive limitations that determine the output. Examples of such limitations are the number of items one can pay attention at a time or the reduced capacity of our short-term memory. Running a model automatically produces a step-by-step simulation of human behavior. The output of the model details each individual cognitive operation (i.e., memory access, visual and auditory activities, and even motor ones). This is a working hypothesis that experimenters compare with the (inferred) steps humans perform in a controlled laboratory task.

Architectures represent knowledge in units called chunks. Each chunk has a number of slots, each of which contains a single symbol. These symbols can represent anything (including other chunks), but do not have inherent semantic value.<sup>1</sup>

### 5.1 SOAR

The first unifying theory of cognition (and architecture) was SOAR, and it was proposed by Newell [22]. Regardless of the actual success and overall adoption of SOAR in cognitive science, the seminal ideas in Newell’s book had a large impact in contemporary thinking. SOAR builds on the idea that problem solving takes place on a problem space [46]: all cognitive acts are some form of search task. SOAR originally stood for State, Operator And Result, reflecting this representation of problem solving as the application of an operator to a state to obtain a result. The basic unit of analysis is the production. Productions operate on chunks. Chunking

---

<sup>1</sup>Note the resemblance with typical Semantic Web formalisms such as RDF-schema or OWL.

is the primary mechanism for learning and represents the conversion of problem-solving acts into long-term memory. The occasion for chunking is an impasse and its resolution in the problem solving process (i.e., satisfying production rules). When SOAR reaches an impasse it uses weak methods, that is, general problem solving methods, such as hill climbing. Once it finds a solution, it stores it as a chunk so next time the agent faces the same problem it will use retrieval instead of falling into an impasse again. SOAR works well in dynamic environments, such as, for instance, controlling a fighter plane, and has some military applications. Its reasoning mechanisms are very general, but could be an interesting reference point for semantic systems using more advanced reasoning engines.

## 5.2 ACT-R

ACT-R [8, 7] divides human knowledge into two irreducible kinds of representations: declarative (“knowing that”) and procedural (“knowing how”). SOAR does not offer this distinction.

The job of the production system is to determine what production to fire, in other words, what action to carry out, at any given moment. Firing takes around 50 milliseconds, and only one production can fire at a time [61]. This enables ACT-R to make predictions about how long operations take for real humans.

ACT-R has a sub-symbolic layer; in fact, it uses a spreading activation mechanism. The (limited) amount of activation available mimics human limitations in working memory capacity. Whether a memory record’s activation exceeds the retrieval criterion is determined by a noisy process. The sources of noise include momentary fluctuations in a record’s estimated gain, estimated cost, and the activation it receives from the current constellation of context elements.

ACT-R’s equations compile the “best of breed” in the last 50 years of findings in different areas of cognition such as attention, memory, judgment etc. It is a good example of how one can put together previously independent constraints and make new ones emerge. In the recent years, Anderson’s research program includes capturing constraints from neuropsychology. ACT-R maps its modules to brain areas and makes predictions about, for example, oxygen consumption when a brain is performing each activity.

## 5.3 Detractors

It is important to point out that the symbol systems view is only one paradigm in cognitive science. Mainstream cognitive science does not work under any unified theory for several reasons:

1. For many researchers the time investment is too high to become proficient on one architecture.
2. It is not clear how (or whether) a brain can implement symbol systems, even though Anderson proposes how ACT-R maps into brain regions.

3. Some scientists do not believe that a symbol system is the “be-all” solution to all problems. In fact, some argue that symbol systems are too powerful [22].
4. Grounded cognition: a symbol system that is not grounded (as in, related to the external world) may turn to be the wrong metaphor. Some propose that abstraction may happen by using perceptual symbol systems instead of a-modal ones [9].

An alternative to symbolic systems is the sub-symbolic view.<sup>2</sup> The basic concept of the neural network approach is the parallel nature of neural processing, and the distributed nature of neural representations [52].

The goal of cognitive modeling with cognitive architectures is to explain as much of cognition as possible using only a small set of explanatory mechanisms. However, one can pursue this goal without a cognitive architecture. Another alternative is to search for general laws of cognition outside a particular architecture. The works of Chater [14, 17, 18, 19, 15] and Tenenbaum [65] fall in this category. Chater presents information distance as the pervasive tool the mind uses in its search for simplicity. Tenenbaum uses Bayesian inference as the common denominator to a wide range of cognitive activities.

## 5.4 Relevance to LarKC

Cognitive architectures aim to explain how the human mind works. It could be the case that some of the solutions the mind uses to, for instance, represent knowledge are effective for open, distributed settings such as in LarKC. There is an obvious problem with scale. The cognitive architectures in question have never been used with amounts of knowledge anywhere close to what LarKC will use. One could tweak the assumptions of the architectures to work with RDF data and investigate how far one can get or which changes or adaptations are required in order to deal with scalability issues.

---

<sup>2</sup>(This alternative is popular in West US coast research communities, while most symbolic research is East-coast based.)

## 6 SIMILARITY THEORIES

The human mind has been “designed” to evaluate similarity fast and efficiently. In the context of LarKC the question is whether the way data is represented within the human mind can be a source of inspiration for building or using a data format to make Web content more machine-friendly.

### 6.1 The Trouble with Similarity

Similarity is very labile and idiosyncratic. Goodman criticized the central role of similarity as an explanatory concept [36]. What does it mean to say that two objects  $a$  and  $b$  are similar? One intuitive answer is to say that they have many properties in common. But this intuition does not take one very far, because all objects have infinite sets of properties in common. For example, a plum and a lawnmower both share the properties of weighing less than 100 pounds (and less than 101 pounds, etc). That would imply that all objects are similar to all others, and vice versa, if we consider that they are different in a infinite set of features, too. Goodman proposed that similarity is thus a meaningful concept when defined with a certain “respect”. Instead of considering similarity as a binary relation  $s(a, b)$ , we should think of it as a ternary relation  $s(a, b, r)$ . However, once “respects” are introduced, the similarity itself has no explanatory value, the respects have. There are four psychological theories that tried to solve the problem.

### 6.2 Metric Models

Metric models (such as the vector space model or Latent Semantic Analysis (LSA)) were the first-comers and still hold important advantages. Although recent developments have addressed some implementation issues (e.g., the Singular Value Decomposition (SVD) can now be run in parallel) the direct application of LSA or any other statistical methods to Semantic Web problems is still not obvious. RDF operations are logical. In LSA vectors are obtained using statistical inference. Combining the logic and statistical approaches seems to be a worthwhile goal, and some groups are pursuing it.

### 6.3 Feature Models

Feature models use set theory. Similarity is a function of common and unique features of the objects compared. Advances in Bayesian methods, such as the topics model, propose that representation might be a language of discrete features and generative Bayesian models instead of continuous spaces. This bottom-up approach has the advantage of generating “topics” instead of un-labeled dimensions, leading to “explainable” representations. Topics can also justify asymmetries

in similarities, as conditional probabilities are indeed asymmetrical ( $P(A|B) \neq P(B|A)$ ). It is possible to build hierarchical topics models.

## 6.4 Structural Mapping Models

These models resort to structural commonalities between objects to infer similarity. Structured representations gain some of their power from the ability to create increasingly complex representations of a situation by embedding relations in other relations and creating higher-order relational structures. These higher-order structures can encode important psychological elements such as causal relations and implications. In fact, RDF as a data structure has this property (termed reification, or called compositionality, in the RDF context). So far compositionality has turned to be difficult to implement for metric and feature models. The suitability of structure-matching models originating from psychology for semantic systems still requires additional investigations, the principal question to be answered being related to the differences between these models and the similarity models used in existing Semantic Web applications. The psychological models use very simple and artificial materials. Most published papers contain a few examples for which the model works (i.e., the solar system mapped to Rutherford's model of the atom), but do not enumerate or discuss those cases in which the respective model is likely to fail. There is no published study on how general a model is (i.e., using a large selection of objects) or about its boundary conditions. More thorough testing, analysis and comparisons are needed to be able to apply psychological models to the Semantic Web in an effective manner. The overall impression is that fine-tuning a model to the examples in the associated paper took a considerable amount of time for the experimenter, so doing the same for a large representative sample of structures may be time consuming. Second, psychological similarity models stress the importance of working memory capacity limitations, which have no relevance for machine learning and general usage in applications. Working memory limitations may help the model explain human patterns such as common errors, but do not contribute to better applications. Third, scaling may be an issue. The Rutherford example requires 42 and 33 nodes to represent the solar system and atom, respectively, and it is one of the largest mappings published. Semantic Web applications can easily deal with knowledge bases several orders of magnitude larger. Last, all these theories use hand-built representations. Information extraction is a type of information retrieval whose goal is to automatically extract structured information, i.e., categorized and contextually and semantically well-defined data from a certain domain, from unstructured machine-readable documents. To date, no psychological theories of the structured kind do information extraction or propose an alternative solution to avoid hand-built representations.

## 6.5 Transformation Distance Models

Transformational distance models reduce similarity to information transmission. For transformational distance theories similarity of two entities is inversely proportional to the number of operations required to transform an entity so as to be identical to another [27, 47, 48, 15, 16].

The most well-developed model of cognition based on string edit is the syntagmatic paradigmatic (SP) model. SP proposes that people use large amounts of verbal knowledge in the form of constraints derived from the occurrences of words in different slots. The constraints are categorized in two types:

1. Syntagmatic associations that are thought to exist between words that often occur together, as in “run” and “fast”.
2. Paradigmatic associations that exist between words that may not appear together but can appear in the same sentence context, such as “run” and “walk”.

The SP model proposes that verbal cognition is the retrieval of sets of syntagmatic and paradigmatic constraints from sequential and relational long-term memory and the resolution of these constraints in working memory. When trying to interpret a new sentence, people retrieve similar sentences from memory and align these with the new sentence. The set of alignments is an interpretation of the sentence. For instance, to build an interpretation of the sentence “Mary is loved by John” they might retrieve from memory “Ellen is adored by George”, “Sue who wears army fatigues is loved by Michael”, and “Pat was cherished by Big Joe”, leading to the following interpretation:

```
Mary is loved by John
Ellen is adored by George
Sue who wears army fatigues is loved by Michael
Pat was cherished by Big Joe
```

The set of words that aligns with each word from the target sentence represents the role that the word plays in the sentence. So, in the example [*Ellen, Sue, Pat*] represents the love role and [*George, Michael, Joe*] the lover role. The model assumes that any two sentences convey similar factual content to the extent that they contain similar words aligned with similar sets of words. Note that SP does not assume any previous knowledge (i.e., syntax). The model can solve basic question-answering tasks such as which tennis player won a match when trained on a specific plain text corpus of such news. There are, of course, models using graph distance, too.

## 6.6 Relevance to LarKC

The standard data model and representation framework underlying the Semantic Web is RDF. In RDF, the fundamental concepts are resources, properties and

statements. Resources are objects, such as books, people or events. Resources have properties such as chapters, proper names, or physical locations. Properties are a special type of resources that describe the relation between two resources. Finally, a statement just asserts the properties of resources. In a sense, psychologists and Semantic Web practitioners are playing the same game: trying to model the world through a formalism. Knowing what humans, and thus, end users, consider similar would ultimately help build better semantic systems. Both XML and RDF are representation languages that can describe labeled trees, thus the applications of the tree edit distance, a subclass of string edit theory, seems to be the obvious choice. There are several algorithms proposed to match such structures efficiently<sup>1</sup>. Bertino et al ([11]), for example, propose a way to match an XML tree to a set of trees (DTDs) in polynomial time. Thus, once the starting knowledge base is in a structured form, there are algorithms to do similarity operations either efficiently or in a cognitively plausible way, but not both. The remaining step is to get from a flat form to a structure that satisfies the requirements of the algorithms, which has proven to be challenging. This step is not necessary for models such as SP, since they work on plain text. In this sense this is a promising venue. Contrary to the Semantic Web idea to create domain-specific data languages by agreement and enforce that conceptual structure onto existing text in the wild, SP proposes no structure a priori. In fact, SP captures meaning as sentence exemplars. The difficult task of either defining or inducing semantic categories is avoided. Topics and SP models do not require pre-existing classes, but still have a long way to go; the need of automatically generating structure is less pressing when one of the driving forces of the Semantic Web is the creation of ontologies.

---

<sup>1</sup>The more general case of measuring the similarity of graphs can be dealt with similarly, e.g. as reported in [26]

## 7 BRAIN SCIENCE/NEUROSCIENCE

Neuroscience is the scientific study of the brain and nervous system. It brings together studies on brain-structure/function, biochemistry, physiology, pharmacology, informatics, computational neuroscience and pathology. Although usually thought of as a branch of biology, neuroscience is attracting interest from many other disciplines, including cognitive and neuro-psychology, computer science, statistics, physics, and medicine. Indeed, many recent theoretical advances in neuroscience have been aided by the use of computational modeling.

The scientific study of the nervous system has advanced in recent times with revolutions in molecular biology, electrophysiology and computational neuroscience. For instance, the processes internal to a single neuron can be relatively well explained. However, understanding of how networks of neurons produce intellectual behavior, cognition, emotion and physiological responses is still poorly understood.

The nervous system is composed of a network of neurons and other supportive cells that form functional circuits, each responsible for specific tasks. Thus, neuroscience can be studied from many different perspectives: molecular, cellular, systems, cognitive:

- Molecular: Studies are aimed at discovering how neurons behave (expressing or receiving of signals) and how complex networks are constructed.
- Cellular: Studies focus on how signals are processed both physiologically and electro-chemically.
- Systems: Studies how the circuits are formed and used to produce the physiological functions, such as reflexes, sensory integration, motor coordination, circadian rhythms, emotional responses, learning and memory, etc.
- Cognitive: Addresses questions of how psychological/cognitive functions are produced by the neural circuitry. The emergence of powerful new measurement techniques combined with sophisticated experimental techniques from cognitive psychology allows neuroscientists and psychologists to address abstract questions such as how human cognition and emotion are mapped to specific neural circuitries.

Additional information about this area is available at various sites. [33] is the Web site of the Society for Neuroscience, containing information and links to introductory as well as latest news publications. [50] collects information related to brain health topics. Finally, the Wikipedia page on Neuroscience [67] provides a good, actual overview of the discipline to start with.

### 7.1 Relevance to LarKC

LarKC will explore complementary techniques that are applicable to selection, abstraction and reasoning. The cognitive and systems perspectives of neuroscience are both interesting to LarKC in these contexts. Furthermore, neuroscience is also

becoming relevant in many inter-disciplinary fields including in the social sciences, including neuro-economics, decision theory and social neuroscience. These studies address some of the most complex questions involving interactions of the brain with environment. Of particular relevance is the brain's ability to draw useful conclusions from incomplete and seemingly unrelated information. Research inspired by LarKC will include techniques for how to identify this unrelated, but useful information and the development or adaptation of algorithms to use such information.

## 8 ECONOMICS

In this chapter we give a brief overview of some of the most important aspects of economy theory and motivate their relevance to LarKC.

### 8.1 Supply-Chain Management

Supply Chain Management (SCM) integrates supply and demand management within and across companies. It focuses on:

- Distribution networks - locations and units between which the products are moving and which handle it along the way such as production facilities, suppliers, distribution centers, warehouses, customers.
- Distribution strategy - how is operation controlled (central vs. distributed), delivery schema (direct, pooling, closed loop), mode of transportation, replenishment strategy (pull, push or hybrid).
- Information - demand signals, forecasts, inventory.
- Inventory management - quantity and location of inventory.
- Cash-Flow - how is money exchanged between the entities in the supply chain.

### 8.2 Cost-Benefit Analysis

Cost-benefit analysis (CBA) can be used to assess the value of a project or a proposal. It requires both the costs and benefits to be expressed in discounted monetary terms, that is the amount of cash translated from the future numbers to its present value using discount rate. One of the important indicators in CBA is the benefit-cost ratio which is computed as present value of benefits relative to the present value of costs. Another indicator is NPV which is the present value of benefits minus the present value of costs. The first indicator gives the relative “return” per invested unit and the second indicator estimates the absolute gain for a project analyzed.

CBA is frequently used in the transportation sector and is used to assess new road or rail construction projects. Several analysis of such projects showed that the costs are usually underestimated, being the result of approximations, “positive thinking” and interests of the people involved to make the project happen.

### 8.3 Risk Management

Risk management (RM) is a structured approach for managing uncertainty. It usually consists of the following steps:

- Identification of potential risks - Risks can be identified from the direction of sources (external such as weather, or internal such as employees of a company) or threats (of losing money).
- Risk assessment (preparation) - This refers to the calculation of a quantitative or qualitative value of a risk related to a concrete situation and a recognized threat. Quantitative risk assessment requires calculations of two components of risk, the magnitude of the potential loss, and the probability that the loss will occur. The risk of an event happening can be measured as “expected loss” by multiplying the values of the two components. In financial decisions loss is usually measured in amount of money. In health or environmental decisions the risk is just measured as the probability of occurrence of an event.
- Developing strategies for managing risk (preparation) - This includes transferring the risk to another party (out-sourcing or insurance), avoiding the risk, reducing the negative effect of the risk, accepting some or all of the consequences of a particular risk (accept and budget).
- Mitigation of risk using available resource (execution)

In ideal world all risks should be assessed. However, investing too much time into assessing and managing unlikely risks can occupy resources which could be more profitably used elsewhere. Also, prioritizing the risks and focusing on the more probable ones or the ones with high losses is non-trivial.

The risks not covered during risk management are called “residual risks” and are usually dealt by using “business continuity planning”. This is an concept used to create and validate a practiced logistical plan for how an organization will recover and restore partially or completely interrupted critical function(s) within a predetermined time after a disaster or extended disruption.

## 8.4 Relevance to LarKC

Ideas from supply-chain management can be used to distribute and balance the load between various plug-ins required to solve a given query. For example, if the reasoning engine requires a larger amount of triples from a triple store, the result could be first cached locally on the triple store (akin a distribution center) and then streamed from there to the reasoning engine (polling distribution strategy). Another scenario would be when a meta-reasoner could forecast demand for specific information from the triple store and would issue commands for caching that information before the reasoning engine would come to ask for it. These aspects are relevant at the conceptual and integration level (work package WP1) and at platform level (work package WP5).

Ideas from risk management and cost-benefit analysis could be used when assessing whether a specific reasoning step is worth executing or maybe using some simpler heuristic could provide a good-enough answer for the overall task. This becomes important when dealing with huge distributed knowledge bases for which it is not possible or feasible to do complete reasoning or to take into account the complete

knowledge base. Risk in such cases can be heuristics providing wrong answers and such cases should be handled in some controllable and to some extent predictable way (e.g., by giving confidence estimations of the provided result). These issues are potentially relevant to all work packages in charge of developing plug-ins, notably work package WP4 (Reasoning and Deciding).

## 9 BOUNDED RATIONALITY

The Bounded Rationality theory was proposed against the traditional postulate that an “Economic Man” would possess enough knowledge to be able to define an appropriate utility function for his actions and a sufficient amount of computational resources to choose those actions which are likely to yield maximum utility in a given environment.

In relation to Web reasoning a statement Herbert Simon made decades ago is still very actual [59]:

“The point of departure is the observation that human thinking powers are very modest when compared with the complexities of the environments in which human beings live. Faced with complexity and uncertainty, lacking the wits to optimize, they [human beings] must be content to satisfy to find “good enough” solutions to their problems and “good enough” courses of action.”

These principles can be applied to Semantic Web reasoning: faced with issues of scalability, heterogeneity, dynamics, inconsistency, and time limitation, “(h)ere, as elsewhere, the best is often the enemy of the good” [60].

The mechanisms how a system of limited computational power can deal with complexity, and the stopped rule based on the satisfying criterion have also been discussed by Simon as early as 1955 [58]. In cognitive science and AI, one of the earliest applications of the notion of bounded rationality was heuristic search in problem solving. The heuristics can usually be used to effectively search the problems space but this method can not guarantee work [1].

In recent years the idea of bounded rationality can be found in various dual-process theories which share the common hypothesis that there are two kinds of information processes in human cognition. One, namely system 1, is fast, automatic and unconscious, and the other one, namely, system 2, is slow, effortful and conscious. Many scientists argue that system 1 is related to intuition and heuristic reasoning, and system 2 to rule-based reasoning. Putting aside this common hypothesis underlying current dual-process theories, the individual theories vary significantly [30]. For example, Kahneman and Tversky paid much attention to the biases of intuition [40], but Gigerenzer and colleagues proposed fast and frugal heuristics programs [35].

### 9.1 Relevance to LarKC

The mechanism and the neural basis of these two systems and the processes of the cooperation between them in our daily life need to be further investigated before being ready to be effectively applied to Semantic Web reasoning. In work package WP4 (Reasoning and Deciding), we will employ both heuristic reasoning in the style of Gigerenzer and colleagues [35] and rule-based reasoning, among other methods, and integrate between different reasoning paradigms. We will also

run functional Magnetic Resonance Imaging (fMRI) experiments<sup>1</sup> to explore the neural basis of human heuristic problem solving.

---

<sup>1</sup>Functional MRI or functional Magnetic Resonance Imaging (fMRI) is a type of specialized MRI scan. It measures the haemodynamic response related to neural activity in the brain or spinal cord of humans or other animals. It is one of the most recently developed forms of neuroimaging.

## 10 PATTERNS AND PATTERN LANGUAGES

### 10.1 Core Concepts

A pattern describes a proven and tested solution to a common problem within a specific context. To be useful a pattern should clearly specify the problem(s) which it can solve and the context in which such problems arise and in which the solution is recommended. In addition, it should contain best practices and guidelines to document the most important aspects which ensure a high quality of the artifact (or system) to be built. A good pattern must achieve the optimal balance between generality and applicability; the idea expressed in a pattern should be general enough to be applied in various settings, but still specific enough to provide effective support.

A pattern language is a method to capture patterns in a structured way. It defines the terminology to be used to name concepts in the field of interest, describes key characteristics of effective solutions for meeting a stated goal, and assists the designer in the application of the patterns through the design process. Elements of a pattern language include the pattern name, pictures describing key aspects of the problem or solution, the context, the problem statement, the factors affecting the story and their trade-offs, the solution, or examples.

The architect Christopher Alexander coined the term "pattern language" [5]. A pattern language provides a means to describe common problems of architectural design, at scales varying from how cities should be built to where to place windows and doors in a corridor. The theory of patterns and pattern languages has been introduced in [5] and elaborated in [6]

According to Alexander [5] patterns can vary in how far they are proven in the real world, in their level of abstraction, and in their level of scale. With respect to the first aspect, Alexander associates a rating to each pattern to indicate how well they are proven in real-world examples. The level of abstraction refers to the type of problem and solution expressed in a pattern. A very generic pattern typically contains examples that are concrete and specific. Finally, the level of scale is related to the scope of the patterns. Low-level patterns can be interconnected to form large-scale systems.

Collections of patterns are available for various disciplines. Putting aside architectural design, patterns are an established instrument in software engineering, social sciences and pedagogy.

An overview of the pattern language and basic patterns introduced by Christopher Alexander et al. in "A Pattern Language" is available at [41]. Yahoo! is publishing a collection of design patterns for Web applications and social networking in [43]. A good summary of patterns and associated languages in the HCI community is available at [32]. Patterns of reasoning are addressed for example in information or cognitive architectures.

## 10.2 Relevance to LarKC

Patterns are important in LarKC at the conceptual and engineering levels. The engineering of the LarKC platform can make use of various types of patterns at analysis, design, implementation and testing time (WP5). As part of the conceptual framework of LarKC (WP1) we survey a wide range of related disciplines, from cognition to economy and biology in the search for ideas which can lead to new paradigms and novel methods of reasoning at Web scale. Integration patterns provide a means to organize and combine these multidisciplinary ideas and to use them in a systematic manner within the LarKC platform. These integration patterns will result into an integration methodology, which will provide a process-oriented view upon a new approach to problem solving based on various techniques: inductive and deductive, Boolean and probabilistic, complete reasoning and limited rationality etc.

## 11 DISTRIBUTED AND PARALLEL COMPUTING

### 11.1 Distributed Systems and Distributed Computing

There are many definitions of what a distributed system is. In [64], a distributed system is defined as a collection of independent computers that appear to the users as a single computer. [24] defines a distributed system as a computer system whose components are located at networked locations and communicate and coordinate tasks using the message passing paradigm. Different types of distributed systems exist ranging from systems with centralized control and shared knowledge to fully decentralized systems where both data (knowledge) and control is distributed. However, independently of the definition adopted or the type of distributed system being considered, every distributed system shares a series of basic characteristics like reliability, transparency (of location, of replication, of migration, of access and concurrency), autonomy of components and support for heterogeneity and scalability. Some of the benefits of using distributed systems in comparison to other approaches like for example the more classical client-server paradigm are scalability, reliability, resource sharing and a higher performance/cost ratio.

Related to the concept of distributed systems is the concept of *Distributed Computing*, which refers to a form of parallel computing where a task is divided into smaller subtasks that run concurrently on different networked computers.

Different types of distributed systems and applications have been developed. For example, computer clusters (Cluster Computing), computer grids (Grid Computing [39]) and Peer-to-Peer systems are well-known types of distributed systems. Examples of distributed applications are distributed query engines, distributed storage engines, distributed file systems, content sharing applications (e.g., Napster, Gnutella, Freenet), collaborative applications (e.g. Jabber, Groove) and instant messaging applications (e.g., Skype).

- Grid Computing - Grid computing is a type of distributed computing that enables coordinated resource sharing and problem solving in large, multi-institutional virtual organizations. Typically, Grid computing is concerned with the execution of computationally-intensive applications such as market forecasting in the financial sector, drug discovery and seismic analysis and simulation among others.
- Cluster Computing - A Computer Cluster, on the other hand, is a group of computers usually connected through high-speed network links that appears to the user as a single super computer. As with Grids, they are typically used for performing computationally-intensive calculations such as the ones found in the areas of e-Business and e-Science.

The design and implementation of distributed systems require a combination of both hardware and software architectures. Typical network architectures include Client-Server, 3-tier architecture (a special case of the N-tier architecture) and Peer-to-Peer computer networks. A well-known software architecture (or middle-

ware) for development of distributed applications is BOINC <sup>1</sup>, which has been used in diverse application areas such as medicine, astrophysics, molecular biology and mathematics. One of the most well-known uses of BOINC is in the SETI@home. Other software platforms include JXTA <sup>2</sup> and .NET <sup>3</sup>, which support implementation of P2P systems. Another approach to implement distributed systems is through the use of a Service Oriented Architecture (SOA) and Web services.

## 11.2 Parallel Computing

Parallel Computing is a form of computation that is based on the idea of splitting a large, compute-extensive task into smaller subtasks that can then be run in parallel (concurrently) over multiple processors or systems. Two of the most well-know forms of parallelism are data-parallelism and task-parallelism. The first focuses on distributing data across multiple processing nodes while the second one focuses on the distribution of tasks or jobs. Other forms of parallelism include bit-level parallelism and instruction-level parallelism.

## 11.3 Relevance to LarKC

The LarKC platform is a modular and distributed platform. These properties together with the distributed nature of the Web make the research in the area of distributed systems highly relevant for the project.

Ideas, techniques and principles from distributed computing can be used in the development of different types of plug-ins such as distributed reasoning and information retrieval plug-ins. Similar approaches in this lines include the works in [57, 31, 62] In the same way, distributed computing techniques can be applied in the area of data storage. The work in [3] for example proposes a distributed RDF storage and retrieval engine. Technologies such as MapReduce, which enables processing of large data sets, and BigTable, a distributed storage system for structured data, could also be explored.

Peer-to-Peer systems are another type of distributed system that can be used in the development of the LarKC platform. Such systems can be used in the implementation of distributed reasoning architectures and plug-ins. Similar approaches include the work done in [4].

Parallelization techniques can be used in the implementation of decision plug-ins. A decision plug-in could split reasoning tasks into smaller ones and dispatch them to a group of plug-ins running in parallel. In the same way, reasoning plug-ins can make use of data-parallelization strategies to divide large amount of data into smaller pieces to be handle by other reasoning plug-ins. Each reasoner would perform a certain task on its local data and then submit the results to a coordinating reasoner plug-in.

---

<sup>1</sup><http://boinc.berkeley.edu/>

<sup>2</sup><https://jxta.dev.java.net/>

<sup>3</sup><http://www.microsoft.com/.NET/>

## 12 SOFTWARE ENGINEERING

The IEEE Computer Society defines software engineering as [2]:

“(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. (2) The study of approaches as in (1).”

More precisely this definition encompasses a number of aspects, including software requirements, software design, software construction, software testing, software maintenance, software engineering management etc.

As software engineering is still an evolving field this is by no means the only possible definition and neither is it generally accepted. Furthermore many different software development methodologies exist which also assign each of the identified subareas of software engineering very distinct priorities or only focus on one of them at all.

Software engineering and programming in general are challenging because of two reasons [28]:

1. The raw amount of computing power and its rapid increase pose a significant novelty.
2. The very deeply hierarchical problem domain, from a high-level architectural view on software down to fine-grained implementation details, that, in general, can not be decomposed into more manageable components in a straightforward manner.

In this sense software engineering is a prime example of human problem solving in regard to very complex systems. Thus basic principles from software engineering are potentially not only applicable in order to produce the LarKC platform, but can also serve as input for the design of the platform and its algorithmic framework in general by observing different software engineering methodologies as problem solving approaches on their own.

This is further reinforced if a program is regarded from a theoretical point of view, as a formula that is derived by the engineer to solve a specific task. In this regard programming is a reasoning task performed by a human in a specific formal language and software engineering methodologies are methods to guide and simplify this process. This highly theoretical approach to software engineering of “program derivation” is described in-depth in [29].

### 12.1 Waterfall Model

The waterfall model, originally proposed by Royce in [51], follows a sequential development approach to the design and implementation of a software system. It identifies several phases of the software engineering process: requirements analysis, design, implementation, validation, integration and maintenance. Each of

these phases follows its predecessor in a pre-defined sequence and has a specific purpose and an expected outcome. Based on this purely sequential ordering it is also important to complete each phase before the next one in the process can start.

The benefits of the waterfall model are mainly its clear and intuitive structure. Its shortcomings were pointed out by Royce himself in the seminal paper from 1987: performing precise planning at the beginning of a project is very difficult and requires a high degree of expertise from all involved parties. Adjustments in later stages of a running project, both from a technical and from a management point of view are costly to accommodate. In this regard the rigid planing of the methodology is also its biggest problem and parallels form the waterfall model can be drawn to traditional sound and complete reasoning in the scope of LarKC.

## 12.2 Spiral Model

The spiral model (as introduced by Boehm in [13]) is a risk-driven methodology that includes iterations of phases such as repeated analysis, design, evaluation and implementation which gradually evolve into a final system. Each iteration also involves an assessment of the further risks (in terms of cost) and possibilities (in terms of software reuse).

The spiral model was one of the first methodologies (albeit not *the* first) to include iterative components in order to explicitly address the limitations of the waterfall model. It has the noteworthy benefit that each cycle can be independently planned and assessed in terms of risks and benefits. Thus it allows to flexibly react depending on the current project status. As such it has been applied successfully in long-term, large software projects. In the context of LarKC it is worthwhile to point out its resemblance to the very general algorithmic pipeline of LarKC, which foresees a repeated execution of retrieval, abstraction, selection, reasoning and deciding tasks.

## 12.3 Agile Software Development

Rather than being a methodology itself, agile software development is a set of principles [34] for building efficient and effective software. It is applied in various methodologies for software engineering such as Pragmatic Programming [38], Extreme Programming [10], Scrum [56] and others. At the core of agile software development we can mention aspects such as customer satisfaction, frequent, incremental releases, close cooperation with customers, starting from the very specification, as well a a continuous adaption of requirements after releases resulting in incrementally improving piece of software.

Each of the methodologies previously mentioned concentrates on particular aspects which are assumed of value for the software development process: test-driven development, specific coding standards and styles, design fundamentals, and so on. The main theme in all these models remains, however, the same, that is, that

in specific contexts very lightweight and simple building blocks are likely to produce suitable results fastest and without much overhead and that these solutions can then be gradually improved. This credo is also shared by the LarKC consortium in its software development strategy and is reflected in our associated work plan and milestones schedule.

## 12.4 Relevance to LarKC

Software engineering is of twofold interest to the LarKC project. First and foremost, one of the main results of the project will be software obviously, which is (or at least should be) the result of a systematic engineering approach that is in line with best practices thought in software engineering. Secondly and more fundamentally software engineering deals with some of the most complex systems ever built by humans. Thus, this discipline is particularly interesting for LarKC because it clearly makes the process of abstracting a problem and formally expressing it by the human programmer explicitly visible.

## 13 CONCLUSIONS AND FUTURE WORK

This deliverable has given an overview of several areas and disciplines related through their methods and principles to the overall goals of LarKC. We have surveyed reasoning paradigms, notably meta-level reasoning, as well as theories from cognition science and brain science, followed by economics and architectural design. These areas provide interesting insights and ideas which can be applied to solve problems which are similar or analogous to the ones with which we are confronted in the LarKC project, from reasoning and deciding to selecting and transforming, and integrating various paradigms into a coherent framework.

The table below summarizes the results of this first survey with respect to the relevance of the corresponding topics in LarKC. These findings will be leveraged in deliverable D1.1.2, due M12, which will elaborate on concrete means to apply ideas, techniques and principles in an inter-disciplinary way to our setting.

These initial findings and insights will be the starting point for the follow-up deliverable of task T1.1, whose aim is to provide a systematic gateway to related work in various other areas of science. Deliverable D1.1.2, due M12, will report on how some of the techniques (or integration patterns of these techniques) from areas surveyed here will be refined or customized in the context of LarKC.

Topic	Relevance to LarKC
Approximation theory	Methods, techniques and heuristics to choose and work with multiple reasoning strategies. Combination of rule-based reasoning with case-based reasoning. Relevant work packages: WP1 and WP4.
Quantum logics	Non-classical negation in the sense of an orthogonal complement as an interpretation of negation in ontologies and Description Logics. Extensional equality and Leibniz-substitutivity principle relevant to various reasoning requirements of typical semantic systems, such as closed world vs. open world assumption, unique name assumption etc. Commonalities between the semantic space model in work package WP2 and quantum logics. Relevant work packages: WP1, WP2, WP4.
Meta-reasoning	Conceptual instrument to decide upon relevant data sources at a particular stage of the reasoning process, to select among different plug-ins realizing the same task, etc. Relevant work packages: WP1, WP4, WP5.
Cognitive architectures	Mind models can be applied to design new methods to structure, represent and use data and to perform reasoning tasks. Relevant work packages: WP1, WP2, WP4.
Similarity theories	Algorithms (transformation-based models, SP model) to compute similarities between a query and records in a knowledge base in an efficient or cognitive plausible way. Relevant work packages: WP1, WP2, WP4.
Brain science	Models capturing the ability of humans to draw useful conclusions from incomplete and seemingly unrelated information could be relevant for the novel reasoning and decision techniques in WP4. Relevant work packages: WP1 and WP4.
Bounded rationality	The basic principle and methods by Herbert Simon are at the core of the LarKC vision. Dual-process theories combining heuristic and rule-based reasoning should be further investigated to implement novel reasoning algorithms. The principles underlying these theories are relevant to the LarKC conceptual framework. Relevant work packages: WP1 and WP4.
Economics	Techniques from the area of supply-chain management are potentially applicable to questions of load balancing. Another use case could be forecasting the requirements for specific information from particular repositories for caching purposes. Relevant work packages: WP1 and WP4. Cost-benefit analysis and risk management are relevant for meta-reasoning and decision, in work packages WP1 and WP4, respectively.
Design patterns	Patterns can be applied for the engineering of the LarKC platform and its underlying conceptual framework. Our project will develop a novel approach to problem solving, which builds upon various techniques. This engineering approach to “philosophical” questions may lead to new insights which we will try to systemize in this task in form of an integration methodology. Design patterns are a core part of such a methodology. Relevant work packages: WP1 and WP 5.
Distributed and parallel computing	Distributed computing methods and techniques can be used in the development of different types of plug-ins such as those for retrieval, reasoning or decision making. Parallelization is relevant for the implementation of decision plug-ins. Relevant work packages: WP2, WP4, and WP5.
Software engineering	Agile software development is applied in LarKC in all technical work packages. Relevant work packages: WP5.

---

## REFERENCES

- [1] A. A. Newell and H. Simon. *Human problem solving*. Prentice-Hall Englewood Cliffs, NJ, 1972.
- [2] A. Abran and J. W. Moore. *Guide to the Software Engineering Body of Knowledge*. IEEE, 2001.
- [3] Gergely Adamku and Heiner Stuckenschmidt. Implementation and evaluation of a distributed rdf storage and retrieval system. In *WI 2005: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 393–396. IEEE Computer Society, 2005.
- [4] P. Adjiman, P. Chatalic, F. Goasdoue, M. C. Rousset, and L. Distributed reasoning in a peer-to-peer setting: Application to the semantic web. *Journal of Artificial Intelligence Research*, 25:269–314, 2006.
- [5] C. Alexander. *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, 1977.
- [6] C. Alexander. *The Timeless Way of Building*. Oxford University Press, 1979.
- [7] J.R. Anderson. *Rules of the Mind*. Lawrence Erlbaum Associates, 1993.
- [8] J.R. Anderson and C. Lebiere. *The Atomic Components of Thought*. Lawrence Erlbaum Associates, 1998.
- [9] L.W. Barsalou. Perceptual symbol systems. *Behavioral and Brain Sciences*, 22(04):577–660, 1999.
- [10] K. Beck. *Extreme programming explained: embrace change*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1999.
- [11] E. Bertino, G. Guerrini, and M. Mesiti. Measuring the structural similarity among XML documents and DTDs. *Journal of Intelligent Information Systems*, 30(1):55–92, 2008.
- [12] G. Birkhoff and J. von Neumann. The logic of quantum mechanics. *Annals of Mathematics*, 37:823–843, 1936.
- [13] B. W. Boehm. A spiral model of software development and enhancement. *Computer*, 21:61–72, 1988.
- [14] G.D. Brown, I. Neath, and N. Chater. A temporal ratio model of memory. *Psychol Rev*, 114(3):539–76, 2007.
- [15] N. Chater. The Search for Simplicity: A Fundamental Cognitive Principle? *The Quarterly Journal of Experimental Psychology Section A*, 52(2):273–302, 1999.
- [16] N. Chater. Cognitive science: The logic of human learning. *Nature*, 407:572–573, 2000.
- [17] N. Chater and G. D. Brown. From universal laws of cognition to specific cognitive models. *Cognitive Science*, 32(1):36–67, January 2008.

- 
- [18] N. Chater and P. Vitányi. Simplicity: a unifying principle in cognitive science? *Trends in Cognitive Sciences*, 7(1):19–22, 2003.
- [19] N. Chater and P.M.B. Vitányi. The generalized universal law of generalization. *Journal of Mathematical Psychology*, 47(3):346–369, 2003.
- [20] M. L. Dalla Chiara and R. Giuntini. Quantum logics. *Handbooks of Philosophical Logic*, 6:129–228, 2002.
- [21] P. Compton and R. Jansen. Knowledge in context: a strategy for expert system maintenance. *Proceedings of the second Australian joint conference on Artificial intelligence table of contents*, pages 292–306, 1990.
- [22] R. Cooper and T. Shallice. Soar and the case for unified theories of cognition. *Cognition*, 55(2):115–149, 1995.
- [23] S. Costantini. Meta-reasoning: A survey. In *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part II*, pages 253–288, London, UK, 2002. Springer-Verlag.
- [24] G. Couloris, J. Dollimore, and T. Kinberg. *Distributed Systems - Concepts and Design*. Addison-Wesley, Pearson Education, 2001.
- [25] M. T. Cox and A. Raja. Metareasoning: A manifesto. In *Proceedings of the International Workshop on Metareasoning: Thinking about thinking, co-located with AAAI-2008*. AAAI Press, 2008.
- [26] Matthias Dehmer, Frank Emmert-Streib, and Jurgen Kilian. A similarity measure for graphs with low computational complexity. *Applied Mathematics and Computation*, 182(1):447–459, November 2006.
- [27] S. Dennis. A Memory-Based Theory of Verbal Cognition. *Cognitive Science*, 29(2):145–193, 2005.
- [28] E. W. Dijkstra. On the cruelty of really teaching computing science. *Communications of the ACM*, 32:1398–1404, 1989.
- [29] E.W. Dijkstra and WH Feijen. *A Method of Programming*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1988.
- [30] B. T. Evans. Dual-processing accounts of reasoning, judgment, and social cognition. *Annual Review of Psychology*, 59, 2008.
- [31] C. Feier. A framework for distributed reasoning on the semantic web based on open answer set programming. In *KWEPSY*, 2007.
- [32] S. Fincher. Hci pattern-form gallery. <http://www.cs.kent.ac.uk/people/staff/saf/patterns/gallery>. last visited September, 2008.
- [33] Society for Neuroscience. Website. <http://www.sfn.org/>, last visited September, 2008.
- [34] M. Fowler and J. Highsmith. The agile manifesto. *Software Development*, 9:28–32, 2001.
- [35] G. Gigerenzer and D. G. Goldstein. Reasoning the fast and frugal way: Models of bounded rationality. *Psychological Review*, 103(4):650–669, 1996.
-

- [36] N. Goodman. Seven strictures on similarity. *How Classification Works: Nelson Goodman Among the Social Sciences*, 1992.
- [37] E. Hüllermeier. Case-Based Approximate Reasoning (Theory and Decision Library B). 2007.
- [38] A. Hunt and D. Thomas. *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley Professional, 2000.
- [39] C. Kesselman I. Foster, editor. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, 1998.
- [40] D. Kahneman. *Maps of bounded rationality: A perspective on intuitive judgment and choice*. Nobel Prize Lecture, 2002.
- [41] Etext library. A pattern language. <http://downlode.org/Etext/Patterns/>, last visited September, 2008.
- [42] RS MICHALSKI and PH WINSTON. Variable precision logic. *Artificial intelligence*, 29(2):121–146, 1986.
- [43] Yahoo! Developer Network. Design pattern library. <http://developer.yahoo.com/ypatterns/>, last visited September, 2008.
- [44] A. Newell. You can't play 20 questions with nature and win: Projective comments on the papers of this symposium. *Visual information processing*, pages 283–308, 1973.
- [45] A. Newell. *Unified theories of cognition*. Cambridge, Massachusetts: Harvard University Press, 1990.
- [46] A. Newell and H.A. Simon. *Human problem solving*. Prentice-Hall, Englewood Cliffs, N.J.: Prentice-Hall, 1972.
- [47] E.M. Pothos and N. Chater. Categorization by simplicity: A minimum description length approach to unsupervised clustering. *Similarity and Categorization*, pages 51–72, 2001.
- [48] E.M. Pothos and N. Chater. A simplicity principle in unsupervised human categorization. *Cognitive Science*, 26(3):303–343, 2002.
- [49] S. Pourazin and A. A. Barforoush. Concurrent metareasoning. *J. Supercomput.*, 35(1):51–64, 2006.
- [50] Brain Research and Information Network. Website. <http://cerebralhealth.com/neuroscienceresearch.php>, last visited September, 2008.
- [51] W. W. Royce. Managing the development of large software systems: concepts and techniques. In *Proceedings of the 9th international conference on Software Engineering*, pages 328–338, Monterey, California, United States, 1987. IEEE Computer Society Press.
- [52] D.E. Rumelhart and J.L. McClelland. *Parallel distributed processing*. MIT Press, 1986.

- 
- [53] S. Russell and E. Wefald. Principles of metareasoning. In *Proceedings of the first international conference on Principles of knowledge representation and reasoning*, pages 400–411. Morgan Kaufmann Publishers Inc., 1989.
- [54] S. Russell and E. Wefald. *Do the right thing: studies in limited rationality*. MIT Press, 1991.
- [55] M. Schut and M. Wooldridge. The control of reasoning in resource-bounded agents. *Knowledge Engineering Review*, 16(3):215–240, 2001.
- [56] K. Schwaber and M. Beedle. *Agile Software Development with Scrum*. Prentice Hall PTR Upper Saddle River, NJ, USA, 2001.
- [57] L. Serafini and A. Tamilin. Drago: Distributed reasoning architecture for the semantic web. In *The Semantic Web: Research and Applications*, volume 3532 of *Lecture Notes in Computer Science*, pages 361–376, 2005.
- [58] H. A. Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, 69, 1955.
- [59] H. A. Simon. *Models of Thought*. New Haven and London: Yale University Press, 1979.
- [60] H. A. Simon. *Models of Bounded Rationality*. Cambridge, MA: MIT Press, 1997.
- [61] T.C. Stewart and R.L. West. Deconstructing ACT-R. *Proceedings of the Seventh International Conference on Cognitive Modeling*, pages 298–303, 2006.
- [62] U. Straccia and R. Troncy. R.: Towards distributed information retrieval in the semantic web: Query reformulation using the omap framework. In *In: Proc. of the 3rd European Semantic Web Conference*, pages 378–392. Springer, 2006.
- [63] G. Takeuti. Quantum set theory. *Current Issues in Quantum Logics*, 8:303–322, 1981.
- [64] A. S. Tanenbaum and M. van Steen. *Distributed Systems: Principles and Paradigms*. Prentice Hall, Pearson Education, 2002.
- [65] J.B. Tenenbaum and T.L. Griffiths. Generalization, similarity, and Bayesian inference. *Behavioral and Brain Sciences*, 24(04):629–640, 2002.
- [66] L.G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
- [67] Wikipedia. Neuroscience. <http://en.wikipedia.org/wiki/Neuroscience>, last visited September, 2008.
- [68] Y.Y. Yao. Three perspectives of granular computing. *Journal of Nanchang Institute of Technology*, 25(2):16–21, 2006.
- [69] Y.Y. Yao. The Art of Granular Computing. *Rough Sets and Intelligent Systems Paradigms International Conference, RSEISP 2007, Warsaw, Poland, June 28-30, 2007: Proceedings*, 2007.

- [70] Y.Y. Yao and N. Zhong. Granular Computing Using Information Tables. *STUDIES IN FUZZINESS AND SOFT COMPUTING*, 95:102–124, 2002.
- [71] S. Zilberstein. Resource-bounded reasoning in intelligent systems. *ACM Computing Surveys*, page 15.