A Knowledge-centric Computation Architecture and the Case of Knowledge Mining

Claus-Peter Rückemann
Westfälische Wilhelms-Universität Münster (WWU), Germany;
Knowledge in Motion, DIMF, Germany;
Leibniz Universität Hannover, Germany
Email: ruckema@uni-muenster.de

***Abstract*—This paper presents a new architecture framework, which is the research result of a series of practical problem solving implementations and further developments of the common knowledge base and integrated application components. The framework is considered knowledge-centric, based on the fundamental knowledge resources, which constitute the fundamental base and imply the core of key assets. Besides the further knowledge development, the knowledge-centric architecture flexibly allows implementations of computation components for many scenarios and the employment of available computation infrastructures. An important quality of the architecture framework is the intrinsic value to assign different roles for the professional tasks in creation and development cycles. These roles regards the major complements of knowledge, including factual, conceptual, and procedural components as well as documentation. This paper refers to a base excerpt of previous implementations and illustrates the framework for an advanced implementation case of knowledge mining. The main goal of this research is to outline the new knowledge-centric architecture and to provide a base for further long-term multi-disciplinary implementations and realisations.**

Keywords–Knowledge Mining and Mapping; Computation Architecture; Context Creation; Universal Decimal Classification; Knowledge-centric Computing.

I. INTRODUCTION

All implementations of mathematical machines, which we call 'computer systems' today, can strictly only deal with formal systems. Knowledge is a capability of a living organism and can itself not be incorporated by formal systems. Neither can intrinsic meaning, which is an essential characteristics of real knowledge and a unique stronghold of knowledge be a matter of formal systems nor can mathematical relations, the theory of sets, exlusiveness or creating completeness be applied to knowledge.

Solutions requiring a wide range of knowledge content as well as implementations of algorithms and components are often difficult to handle, the more when it comes to operating the resulting solutions for decades or even further developing content and implementations for long-term. Over time, the further developments and services are becoming more complicated without a common, holistic frame for content and implementation. When gathering a large number of independent implementations, we experienced an increasing heterogeneity in content development but also in implementations of computing components.

This background is the major motivation for the development of an advanced framework based on long-term Knowledge Resources and integrated application components providing a valuable means of tackling the challenges. Nevertheless, in complex cases even such major component groups cannot protect long-term challenges, if there is no basic framework architecture caring for knowledge and computational implementation. The practice of creating solutions, which have to deal with the complements of knowledge suggests that flexible but nevertheless methodological, systematical approaches are required. The goal of this research is to create such knowledge-centric architecture, based on a wide range of multi-disciplinary implementations and practical case studies in different disciplines and dealing with different foci, for many years. While further developing and updating the knowledge related attributes, data, implementations, and solutions, all of them had to be revisited over time, improving and where necessary recreating implementation and content.

Knowledge Resources and originary resources cover the complements of factual, conceptual, procedural, and metacognitive complements, e.g., from collections and references resources. The architecture presented here aims to seamlessly integrating separate roles of contributing parties, e.g., scientific staff creating research data, professional classification by experienced research library specialists, and developers of application components as well as services. The guideline was enabling to retain the knowledge required to resemble the intrinsic complexity of realia situations, real and material instead of abstract situations, while allowing lex parsimoniae principles of William of Ockham for problem solving. The overall outcome gained from the development of practical solutions led to the creation of a knowledge-centric architecture, which essentials are presented in the following sections.

This paper is organised as follows. Section II introduces to fundaments and previous work, Section III presents the architecture being result of this research. Sections IV and V deliver an implementation case of two major use-cases, including a discussion. Section VI summarises the results and lessons learned, conclusions, and future work.

II. FUNDAMENTS AND PREVIOUS WORK

With one of the best and most solid works, Aristotle outlined the fundaments of terminology and of understanding knowledge [1] being an essential part of 'Ethics' [2]. Information sciences can very much benefit from Aristotle's fundaments

 ISBN: 978-1-61208-732-0

and a knowledge-centric approach, e.g., by Anderson and Krathwohl [3], but for building holistic and sustainable solutions they need to go beyond the available technology-based approaches and hypothesis [4] as analysed in Platons' Phaidon. So far, there is no other practical advanced knowledge-centric architectural specification known, which implements these fundaments. Making a distinction and creating interfaces between methods and applications [5], the principles are based on the methodology of knowledge mapping [6]. The implementation can make use of objects and conceptual knowledge [7] and shows being able to build a base for applications scenarios like associative processing [8] and advanced knowledge discovery [9]. Based on this background, during the last decades, a number of different case solutions were created, implemented, and realised on this fundament, including: Dynamical visualisation, knowledge mining, knowledge mapping, Content Factor, phonetic algorithms, Geoscientific Information Systems (GIS), Environmental Information Systems (EIS), cartographic mapping, service design, service management, and High End Computation. All such implementations include extensive use of Knowledge Resources and computation algorithms. This paper, presenting the new architecture, does not allow to illustrate the details of any implementation. Therefore, an excerpt of practical solutions is cited, which have been reimplemented by the collaboration of the participated research groups and published, creating a base for this architecture. Representative examples are a) integrated systems and supercomputing resources used with phonetic algorithms and pattern matching [10] for knowledge mining [11], b) multi-dimensional context creation based on the methodology of Knowledge Mapping [12], and c) an exemplary resulting, widely used conceptual knowledge subset for geo-spatial scenarios [13]. The Knowledge Resources cover the factual, conceptual, procedural, and metacognitive complements in all cases, e.g., from collections and references resources.

An understanding of the essence and complexity of universal, multi-disciplinary knowledge can be achieved by taking a closer look on classification. The state-of-the-art of classifying 'universal knowledge' is the Universal Decimal Classification (UDC) [14] and its solid background, flexibility, and long history. The LX Knowledge Resources' structure and the classification references [15] based on UDC [16] are essential means for the processing workflows and evaluation. Both provide strong multi-disciplinary and multi-lingual support. For the research, all small unsorted excerpts of the Knowledge Resources objects only refer to main UDC-based classes, which for this publication are taken from the Multilingual Universal Decimal Classification Summary (UDCC Publication No. 088) [17] released by the UDC Consortium under the Creative Commons Attribution Share Alike 3.0 license [18] (first release 2009, subsequent update 2012). These components and their qualities are integrated in the resulting architecture with the methodologies and systematic use.

III. Resulting Knowledge-centric Architecture

As discussed above, the presented architecture is the result based on a series of previously implemented problem solutions and Knowledge Resources developments over the last years.

A. General Computation Architecture

The complements diagram of the implementation architecture [19][20][21] is shown in Figure 1. The major components

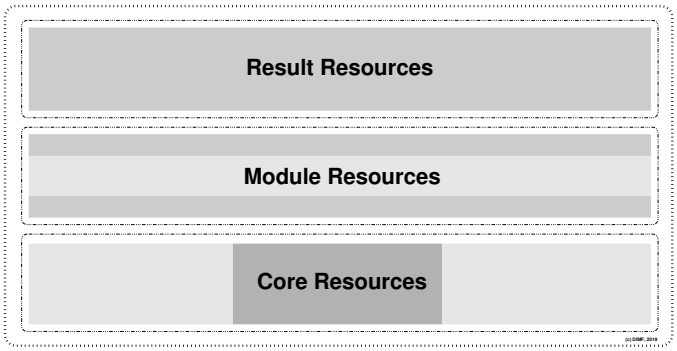

Figure 1. Complements diagram of the resources components architecture, including the three main complements of core, module, and result resources.

are core resources and module resources. The result resources include objects collections, which result from the application of core and module resources in arbitrary scenarios. The sizes of this figure and the associated complements diagrams correspond, the following figures show complementary details from this context. The core resources in this architecture comprise required resources. The complements diagram (Figure 2) shows the essential detail.

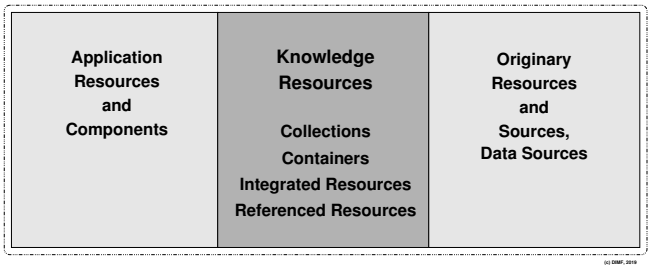

Figure 2. Computation architecture: Complements diagram of general core resources, from originary to knowledge and application resources.

The core resources can be divided into three categories: The central Knowledge Resources, originary resources, and application resources and components. The first, the Knowledge Resources, can include collections and containers as well as integrated resources and references to resources. The second, the originary resources, can include realia and original sources, which in many cases may have instances in the Knowledge Resources. The third, the application resources, can include implementations of algorithms, workflows, and procedures, which form applications and components. Instances of these components can also be employable in solutions due to their procedural nature, e.g., in module resources.

The complements diagram of general module resources is shown in Figure 3. A general set of module resources consists of input resources, modules, and output resources. The central workflow module entities are accompanied by interface module entities for input and output resources. For many architecture

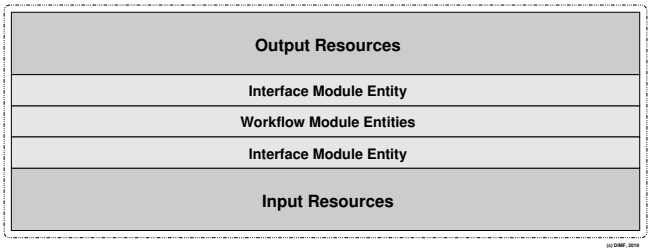

Figure 3. Computation architecture: Complements diagram of module resources, from input, interfaces and workflow entities to output.

implementations, chains of module resources can be created, which can, for example, be used in pipeline and in parallel.

B. Architecture Complements for Knowledge Mining

For the case of knowledge mining, the complements diagram of the core resources is shown in Figure 4. Application re-

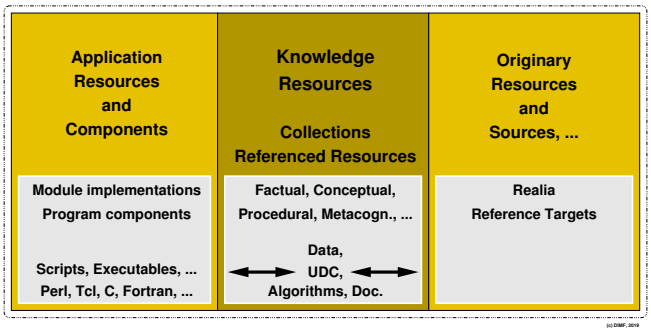

Figure 4. Knowledge Mining: Complements diagram of the core resources and examples of their contribution implementations.

sources and components are based on module implementations and program components for the knowledge mining realisation. Implementations employ scripting, high level languages, and third party components. Knowledge Resources and originary resources cover the complements of factual, conceptual, procedural, and metacognitive complements, e.g., from collections and references resources.

The respective complements diagram of a module resource for a text based knowledge mining implementation consists of several features (Figure 5). The input and output resources

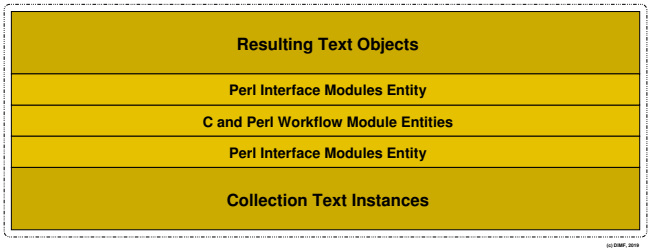

Figure 5. Knowledge Mining: Complements diagram of a module resource used for creating module chains for text based knowledge mining.

consist of text object instances in text based cases of knowledge mining. Here, the module entity implementations were implemented in C and Perl [22] for the respective implementations. The interface module entities are implemented in Perl, with the option to be on-the-fly generated within a workflow.

The knowledge-centric architecture does focus on resources and application scenarios, one of the most important is computation cases. Large computation workflow chains can be built with the architecture as was demonstrated with the reimplemented solutions for different cases, which were above referred to. An implementation sequence of module resources can be considered an intermediate step in building a workflow. Results within an implementation sequence can be considered intermediate results and instances, e.g., from the integrated mining of collection and container resources.

IV. Implementation Case and Discussion

The goal was to create a knowledge-centric computation architecture, which allows a close integration of Knowledge Resources with wide spectra of complementary knowledge and flexible, efficient computational solutions, while being able to specify practically required roles for creation and long-term development. The knowledge-centric architecture can provide a base for an arbitrary range of use-cases and associated components. Two major groups of use-cases are

- resources creation and development and
- knowledge mining and selected associated methods.

A. Major use-case groups

Figure 6 shows an use-case diagram of the knowledge-centric computation architecture. The excerpt illustrates an

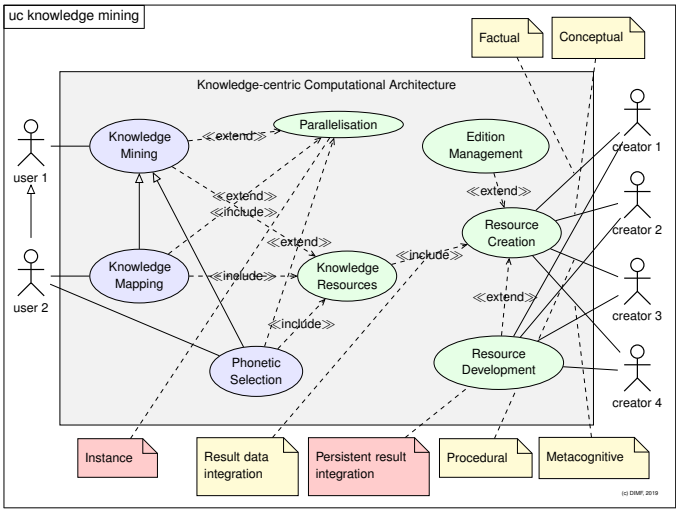

Figure 6. Use-case diagram of the knowledge-centric computation architecture: Two major use-case groups with four creator roles.

integrated view on the two groups of knowledge mining (blueish), which was implemented spanning knowledge mining use-cases and knowledge creation and development use-cases (greenish). In this widely deployed scenario, the implementation does have two main types of actors, namely creators and users. The use-cases have different actors, two ‘user’ roles and four ‘creator’ roles.

The selected system context is given by the grey box. The selected use-cases (ellipses) can be distinguished in use-cases for creators (greenish: resource creation, resource development, edition management, parallelisation) and use-cases for users (bluish: knowledge mining, knowledge mapping, phonetic selection).

Knowledge mining is supported by and using the cases of knowledge mapping and phonetic selection, which inherit to the knowledge mining instance the implemented methods and algorithms contributed by other user groups. For clearness, the creator and other roles for these two cases are not included in this diagram. Knowledge mining, mapping, and phonetic selection include the use-case of Knowledge Resources. The cases of this group are extended by parallelisation, respective computation, here instance based workflow parallelisation, which enables the computation-relevant optimisation for individual implementations and infrastructures.

The Knowledge Resources include the use-case of resource creation, which allows to integrate persistently added results. The use-case of resource creation itself is extended by the use-cases of resource development and edition management, which allows to define editions of resources for consistency in advanced complex application scenarios.

The use-case scenario reflects the professional practice of having separate roles for creating and developing factual, conceptual, procedural, and documentation, respective metacognitive knowledge. In most cases, different specialists are employed for creating and developing

- factual knowledge, e.g., research data and its documentation,
- conceptual knowledge, e.g., classification of knowledge objects,
- procedural knowledge, e.g., procedures, workflows, programs, and their respective documentation,
- metacognitive knowledge, e.g., documentation of experiences.

In practice, the creators are commonly represented by different groups of experts, e.g., scientists, classification experts in scientific libraries, and designers of scientific algorithms and workflows.

B. Main Components

The core components of a basic knowledge mining implementation with the Knowledge Resources, based on the knowledge-centric computation architecture, can be summarised with a block diagram (Figure 7). The block diagram shows the Knowledge Resources and two types of knowledge object groups, namely object collections and containers. Each type and implementation can have individual and specialised interfaces. Knowledge mining is provided by an interface with the Knowledge Resources. The diagram also contains the interface block, due to the importance of the resource creation use-case. The individual groups have ports, interaction points, which can be used via interfaces, e.g., Knowledge Resources Creation (KRC) port and Knowledge Resources Mining (KRM) port. Components can have further interfaces, with and without delegating ports. It is common that independent resources are in many cases not necessarily orchestrated.

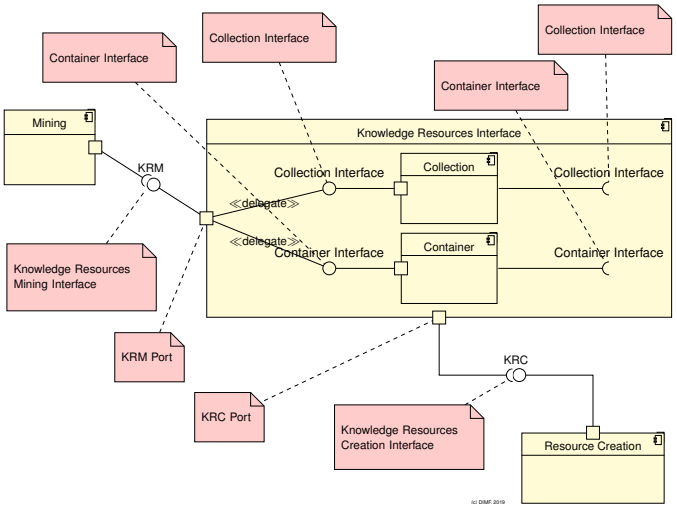

Figure 7. Block diagram of respective knowledge-centric computation architecture components: Excerpt of Knowledge Resources and interfaces.

C. Activity groups

A number of activities are associated with different components. An important activity regarding the resource creation is the creation-update (Figure 8). The resource creation component has to provide creation and update activities for the different creator groups. A simple but important example for resource creation and development is the creation of an object instance and respective updating an existing object with a new instance.

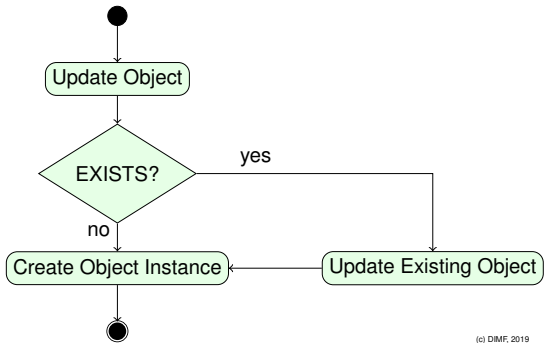

Figure 8. Activity diagram illustrating the essential object creation-update activities in the knowledge-centric computation architecture.

Start state is any state of the Knowledge Resources. End state is a new state of the Knowledge Resources. Regarding resource creation and development use-cases the start and end states should be considered intermediate states. As shown in the use-case diagram, professional practice affords the implementation of according activities for all required, specialised creator roles. A fundamental mining activity with Knowledge Resources is a resource request targeting to create an intermediate result (Figure 9). Start state is any state in a knowledge mining workflow chain. End state is a new state in a knowledge mining workflow chain. Regarding knowledge mining use-cases the start and end states should be considered intermediate states. If a resource is not available then an

 ISBN: 978-1-61208-732-0

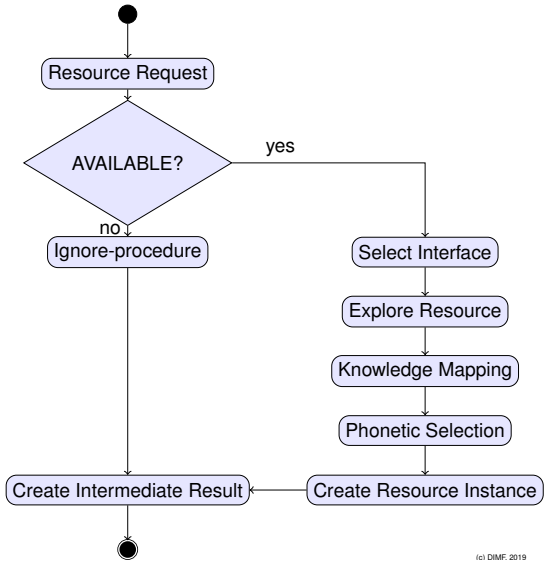

Figure 9. Activity diagram of a basic resource request for creating an intermediate knowledge mining result.

ignore-procedure continues for creating an intermediate result. The ignore-procedure can contribute its status to the workflow chain. If the resource is available then an interface is selected for the resource exploration. The exploration can use available activities, e.g., knowledge mapping and phonetic selection, in order to create a resource instance, which contributed to creating an intermediate result for the workflow chain. Examples of activities are multi-dimensional context creation by knowledge mapping [6] and phonetic association, e.g., using Soundex [23][24][25]. Sample Soundex codes developed [11] are used for names in various textual, contextual, and linguistical situations, implemented in order to be integrated in a large number of situations.

V. Discussion

The computation architecture provides the flexibility that workflow chain modules and whole workflow chains can be employed sequential or parallel. The components in general are not limited by the architecture to be implemented for synchronous or asynchronous accesses if required for arbitrary algorithms and workflows.

The implementations for practical case studies built upon this architecture span different disciplines and deal with different foci. The excerpted cases include general, simplified cases of knowledge mining and practical knowledge development scenarios from realisations, which were implemented for large practical solutions. These cases are relevant because of the professional background and practice required to deal with development of resources and application components for long-term tasks.

In complex scenarios, different disciplines contribute fulfilling different tasks. In case of knowledge creation and development and its valorisation different specialised expertise is required. In general, content and applications are created by different disciplines. Even different aspects of content may require different specialists groups, different roles, e.g., natural sciences research data and conceptual valorisation are often done by scientists from a respective discipline and information scientists. Many components have to be revisited and improved over time as the results and facilities should be continued and preserved and be available for long-term. In the implementation cases, factual, conceptual, procedural, and metacognitive knowledge is cared for by different experts. The architecture allows flexible and efficient separation of roles. For example, research data can be created by a role and can at any time be amended with classification and procedural documentation by experienced research library specialist and researcher roles.

The Knowledge Resources are containing a lot more content and references than can be used at present time in most cases. The architecture allowed to support retaining the associated knowledge required to resemble the intrinsic complexity of realia situations while implementing selected solutions for isolated as well as complex situations. The development of knowledge mining and the provisioning of services based on these tasks can continuously be done by application developers, accessing the continuously extendable Knowledge Resources.

VI. Conclusion

The paper presented the research results of creating a knowledge-centric computation architecture. The resulting architecture was developed in continuous cross development of multi-disciplinary, multi-lingual Knowledge Resources and practical knowledge-centric solutions. This paper presented the major qualities of the computation architecture. The practical employment of the architecture was illustrated for advanced knowledge mining and practical development uses-cases.

The contributing research collaboration achieved to create a practical approach for a knowledge-centric computation architecture, which allows the methodological and systematical development and employment of components, including Knowledge Resources. The architecture covers the creation of flexible solutions, which allow to most widely employ the complements of knowledge.

Computation architecture and use-cases proved in practice to support both the seamless separation and integration of roles for different disciplines and tasks while implementing and realising solutions. In addition to the implemented and referred case studies, we showed that major use-cases can be efficiently be managed. Especially, on the one hand, knowledge creation and development can be professionally dealt with by groups from responsible disciplines. On the other hand, knowledge mining relying on the resources can be based on the work of these disciplines while service based use and implementation can be given different roles.

Future research will concentrate on further extending and developing Knowledge Resources in order to provide long-term capacities and creating new advanced algorithms and mining workflows for enabling fundaments for new insight, participating different institutions and roles, based on the knowledge-centric computation architecture.

 ISBN: 978-1-61208-732-0

ACKNOWLEDGEMENTS

We are grateful to the "Knowledge in Motion" (KiM) long-term project, Unabhängiges Deutsches Institut für Multidisziplinäre Forschung (DIMF), for partially funding implementations, case studies, and publications under grants D2016F5P04648, D2017F1P04812, and D2018F6P04938 and to its senior scientific members and members of the permanent commission of the science council and the board of trustees, especially to Dr. Friedrich Hülsmann, Gottfried Wilhelm Leibniz Bibliothek (GWLB) Hannover, to Dipl.-Biol. Birgit Gersbeck-Schierholz, Leibniz Universität Hannover, to Dipl.-Ing. Martin Hofmeister, Hannover, and to Olaf Lau, Hannover, Germany, for collaboration, practical multi-disciplinary case studies, and the analysis and implementation of advanced concepts. We are grateful to Dipl.-Ing. Hans-Günther Müller, Cray, Germany, for his excellent contributions and assistance, providing practical private cloud and storage solutions. We are grateful to all national and international partners in the Geo Exploration and Information cooperations for their constructive and trans-disciplinary support. We are grateful to the Science and High Performance Supercomputing Centre (SHPSC) for long-term support and The International ARS Science and History Network for providing multi-disciplinary application scenarios and assistance. / DIMF-PIID-DF98_007.

REFERENCES

[1] Aristotle, Nicomachean Ethics, 2008, (Written 350 B.C.E.), Translated by W. D. Ross, Provided by The Internet Classics Archive, URL: http://classics.mit.edu/Aristotle/nicomachaen.html [accessed: 2019-03-24].

[2] Aristotle, The Ethics of Aristotle, 2005, Project Gutenberg, eBook, EBook-No.: 8438, Release Date: July, 2005, Digitised Version of the Original Publication, Produced by Ted Garvin, David Widger, and the DP Team, Edition 10, URL: http://www.gutenberg.org/ebooks/8438 [accessed: 2018-01-01].

[3] L. W. Anderson and D. R. Krathwohl, Eds., A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. Allyn & Bacon, Boston, MA (Pearson Education Group), USA, 2001, ISBN-13: 978-0801319037.

[4] Plato, Phaedo, 2008, (Written 360 B.C.E.), Translated by Benjamin Jowett, Provided by The Internet Classics Archive, URL: http://classics.mit.edu/Plato/phaedo.html [accessed: 2019-03-24].

[5] C.-P. Rückemann and F. Hülsmann, "Significant Differences: Methodologies and Applications," KiMrise, Knowledge in Motion Meeting, November 27, 2017, Knowledge in Motion, Hannover, Germany, 2017.

[6] C.-P. Rückemann, "Methodology of Knowledge Mapping for Arbitrary Objects and Entities: Knowledge Mining and Spatial Representations – Objects in Multi-dimensional Context," in Proceedings of The Tenth International Conference on Advanced Geographic Information Systems, Applications, and Services (GEOProcessing 2018), March 25 – 29, 2018, Rome, Italy. XPS Press, Wilmington, Delaware, USA, 2018, pp. 40–45, ISSN: 2308-393X, ISBN-13: 978-1-61208-617-0, URL: http://www.thinkmind.org/index.php?view=article&articleid=geoprocessing_2018_3_20_30078 [accessed: 2019-03-24].

[7] C.-P. Rückemann, "Creation of Objects and Concordances for Knowledge Processing and Advanced Computing," in Proceedings of The Fifth International Conference on Advanced Communications and Computation (INFOCOMP 2015), June 21–26, 2015, Brussels, Belgium. XPS Press, 2015, ISSN: 2308-3484, ISBN-13: 978-1-61208-416-9, URL: http://www.thinkmind.org/index.php?view=article&articleid=infocomp_2015_4_30_60038 [accessed: 2019-03-24].

[8] C.-P. Rückemann, "Advanced Association Processing and Computation Facilities for Geoscientific and Archaeological Knowledge Resources Components," in Proceedings of The Eighth International Conference on Advanced Geographic Information Systems, Applications, and Services (GEOProcessing 2016), April 24 – 28, 2016, Venice, Italy. XPS Press, 2016, pages 69–75, ISSN: 2308-393X, ISBN-13: 978-1-61208-469-5.

[9] C.-P. Rückemann, "Advanced Knowledge Discovery and Computing based on Knowledge Resources, Concordances, and Classification," International Journal On Advances in Intelligent Systems, vol. 9, no. 1&2, 2016, pp. 27–40, ISSN: 1942-2679.

[10] "Perl Compatible Regular Expressions (PCRE)," 2019, URL: https://www.pcre.org/ [accessed: 2019-03-24].

[11] C.-P. Rückemann, "Archaeological and Geoscientific Objects used with Integrated Systems and Scientific Supercomputing Resources," International Journal on Advances in Systems and Measurements, vol. 6, no. 1&2, 2013, pp. 200–213, ISSN: 1942-261x, URL: http://www.iariajournals.org/systems_and_measurements/sysmea_v6_n12_2013_paged.pdf [accessed: 2019-03-24].

[12] C.-P. Rückemann, "Multi-dimensional Context Creation Based on the Methodology of Knowledge Mapping," International Journal on Advances in Software, vol. 11, no. 3&4, 2018, pp. 286–298, ISSN: 1942-2628, URL: http://www.iariajournals.org/software/soft_v11_n34_2018_paged.pdf [accessed: 2019-03-24].

[13] C.-P. Rückemann, "Superordinate Knowledge Based Comprehensive Subset of Conceptual Knowledge for Practical Geo-spatial Application Scenarios," in Proceedings of The Eleventh International Conference on Advanced Geographic Information Systems, Applications, and Services (GEOProcessing 2019), February 24 – 28, 2019, Athens, Greece. XPS Press, Wilmington, Delaware, USA, 2019, pp. 52–58, ISSN: 2308-393X, ISBN: 978-1-61208-687-3, URL: http://www.thinkmind.org/index.php?view=article&articleid=geoprocessing_2019_3_30_30039 [accessed: 2019-03-24].

[14] UDC, Universal Decimal Classification. British Standards Institute (BSI), 2005, complete Edition, ISBN: 0-580-45482-7, Vol. 1 and 2.

[15] C.-P. Rückemann, "Enabling Dynamical Use of Integrated Systems and Scientific Supercomputing Resources for Archaeological Information Systems," in Proc. INFOCOMP 2012, Oct. 21–26, 2012, Venice, Italy, 2012, pp. 36–41, ISBN: 978-1-61208-226-4.

[16] "UDC Online," 2018, URL: http://www.udc-hub.com [ac.: 2019-03-24].

[17] "Multilingual Universal Decimal Classification Summary," 2012, UDC Consortium, 2012, Web resource, v. 1.1. The Hague: UDC Consortium (UDCC Publication No. 088), URL: http://www.udcc.org/udcsummary/php/index.php [accessed: 2019-03-24].

[18] "Creative Commons Attribution Share Alike 3.0 license," 2012, URL: http://creativecommons.org/licenses/by-sa/3.0/ [accessed: 2019-03-24].

[19] B. Oestereich and A. Scheithauer, Analyse und Design mit der UML 2.5: Objektorientierte Softwareentwicklung, 11th ed. De Gruyter Oldenbourg, 2013, ISBN: 978-3486721409.

[20] "Unified Modeling Language (UML)," 2019, URL: http://uml.org/ [accessed: 2019-03-24].

[21] "About the Unified Modeling Language Specification 2.5.1," Dec. 2017, OMG – Object Management Group, URL: https://www.omg.org/spec/UML/ [accessed: 2019-03-24].

[22] "The Perl Programming Language," 2019, URL: https://www.perl.org/ [accessed: 2019-03-24].

[23] R. C. Russel and M. K. Odell, "U.S. patent 1261167," 1918, (Soundex algorithm), patent issued 1918-04-02.

[24] D. E. Knuth, The Art of Computer Programming: Sorting and Searching. Addison-Wesley, 1973, vol. 3, ISBN: 978-0-201-03803-3, OCLC: 39472999.

[25] National Archives and Records Administration (NARA), "The Soundex Indexing System," 2007, 2007-05-30, URL: http://www.archives.gov/research/census/soundex.html [accessed: 2019-03-24].

 ISBN: 978-1-61208-732-0