This is not domain analysis

Albrechtsen, Hanne

Published in:
Knowledge Organization

Publication date:
2015

Citation for published version (APA):
Albrechtsen, H. (2015). This is not domain analysis. *Knowledge Organization*, *42*(8), 557-561.

This is Not Domain Analysis†

Hanne Albrechtsen

Copenhagen University, Royal School of Library and Information Science,
Birketinget 6 Copenhagen dk2300 S, Denmark
<hanne.albrechtsen@hum.ku.dk>

Hanne Albrechtsen is Director of the Institute of Knowledge Sharing (IKS), Copenhagen. She is a part-time lecturer at Copenhagen University and senior scientist at IKS. She holds a doctorate in computer science from Aalborg University, and an MLIS from the Royal School of Librarianship, Copenhagen, Denmark. She is past president of the International Society for Knowledge Organization. Current research areas include knowledge management, domain analysis, entrepreneurship, cognitive work analysis and HCI.

Albrechtsen, Hanne. **This is Not Domain Analysis.** *Knowledge Organization. 42(8)*, 557-561. 15 references.

Abstract: The article is about the origins of domain analysis for knowledge organisation. Domain analysis has become an established notion in information science and it covers the field quite broadly. Yet from the outset, the aim was to present a methodological alternative to the information processing paradigm in information science, which was especially focused on developing models for information retrieval and automatic indexing. Despite, or perhaps, because of the wide-spread use of the term "domain analysis" in information science education and research, there are varied understandings of its history and meaning. The aim of this contribution is to bring domain analysis back to the original roots of classifying in the world, and to bring a first definition of a modern of domain analysis as a method of inclusion, heterogeneity and wholeness.

Received: 4 November 2015; Accepted 11 November 2015

Keywords: domain analysis, software development, classification research

† I am indebted to the late professor of social science Susan Leigh Star for inspiring the title and perspectives on the origin of domain analysis. I would like to thank Professor John Law, Open University London, for supervising my research on "method" and discourse analysis during 1998-2000. I am grateful to my colleague Professor Birger Hjørland for joint further development of domain analysis as a theory for information science. I would also like to thank my fellow researchers in the PRACTITIONER for initiating me into the reality of software development, and in particular Ms. Lene Olsen, project manager at CRI. Finally, I am grateful to Professor María J. López-Huertas for commenting on early drafts of this article, and to doctoral student Matthew Kelly, for valuable comments and language editing.

1.0 Introduction: The journey begins

In June 1992, I prepared the thesis "Domain Analysis for Classification of Software," in connection with the information science program at the Royal School of Library and Information Science, Denmark. Since the spring of 1987, I had been working as a research engineer at Computer Resources International (CRI), a medium-sized engineering company based in Birkeroed, Denmark. CRI was involved in international research, in particular for the European Strategic Program on Research in Information Technology (ESPRIT). Quite a few of the ESPRIT projects involved problems that are at the core of information science, including knowledge organization in particular. During my affiliation with CRI, I contributed to several ESPRIT projects, in particular to the PRACTITIONER project (P1094, 1987-1991).

PRACTITIONER was concerned with new approaches to software development, based on software reuse. The research on software reuse was carried out in collaboration teams of researchers from many fields and many European countries, for instance computer science, computational linguistics, anthropology and software engineering. The research was not constrained by any particular methodology. The floor was open to many ways of doing things. Knowledge sharing amongst the European project teams, and in the local teams was essential.

The research manager at CRI formulated a strategy to create sixth generation computing. There was no fixed idea of what that would be—except that the sixth genera-

tion would be conceptually more advanced than the fifth generation computing, which focused heavily on artificial intelligence (AI) and intelligent information technology (IT) systems. The group was expected to explore that vision through project investigations of software design and applications for various domains. In order to follow the idea of working from the domains towards the system design, researchers from many professional fields were involved. Likewise, the groups reflected an even gender representation. That was quite an innovative style of thinking considering the breakneck pace that research was conducted in during the late 1980s; engineering and computer science were moving ahead quickly with the conceptual developments in the artificial intelligence (AI) and expert systems. The vision was that new AI developments and concepts would migrate into industrial developments like intelligent banking, medical informatics, automatic translation, and automatic indexing. The main fields driving the development were mathematics, computer science and engineering. The research was highly specialized. The development of tools and prototypes emerged in closed laboratory environments. It was a world of secrets, funded by billions of US dollars or European Currency Units (ECUs). It was also a world of causal reasoning and intense planning. Yet the research manager at CRI painted a different path of development through unconventional entrepreneurial thinking (Sarasvaty 2009). Because of confidentiality provisions there were intense restrictions on communicating details about the work in CRI's IT research laboratory during the first few years.

Special areas of interest at CRI concerned how classification can support software development and reuse. The research investigations led to contacts and collaboration with experts in classification and indexing and related fields. In 1990, Dr. Ingetraut Dahlberg founded a new professional society for the field: the International Society for Knowledge Organization (ISKO). ISKO's first international conference took place in Darmstadt in 1990. The conference provided a first opportunity to present the work done in the PRACTITIONER project and to build community with colleagues in the field (Albrechtsen 1990). Quite a few presentations of PRACTIONER have been given at software engineering conferences and in associated journals (eg., Mili et al. 1994), or in technical reports, such as those delivered to the European Economic Community (EEC) (e.g., Sedwell et al. 1988) and later to the European Union. The work done prior to the development of "domain analysis" was at the same time highly multi-disciplinary and collaborative.

2.0 Software concepts and domains

PRACTITIONER (P1094, 1988-1992) developed the notion of "software concepts" as a main unit of analysis for developing tools for software reuse. "Software concepts" covered a range of standard representational models for system development, including data models and object modeling, as well as descriptions and maps of software components for medium-sized or large software systems. The research in the software reuse field was and is quite comprehensive (for an overview, see Albrechtsen 1992; Mili et al. 1994; Frakes and Kang 2004). The main factors driving this research were the growing demands for large-scale systems in the military sector and the public sector. The large number of failed IT systems also meant that there was an obvious need for system development; in addition to this there was a need to rein in development costs—the programmers, to a large degree, ruled the land. The code (software programs) was often largely incomprehensible to teams outside of the development teams, and despite standardization and documentation, there were still problems getting big and complex systems built. The overall dream was, metaphorically, an industrial one: a software factory, where existing software and models could be linked and put to life instead of being taken away to a software cemetery and laid to rest. This was, regrettably, the state of affairs at the time PRACTITIONER set out with ideas for solutions and prototype tools.

Software reuse research started out as a highly competitive field. It unfolded within various disciplinary settings, typically within software engineering and computer science (Frakes and Kang 2004; Prieto-Diaz 1991). From the point of view of classification research, Prieto-Diaz's approach is especially interesting because its main theoretical basis is Ranganathan's theory of faceted classification. Prieto-Diaz developed a faceted scheme for classification of software components and introduced the term "domain analysis" for the analytic-synthetic approach that he suggested.

Despite the rich choices of knowledge organization methods and systems that we explored in PRACTITIONER, a comprehensive theory for classification of software did not exist. The multiplicity in the field during the 1980s and 1980s is typical of a new research field, with many different disciplines and interests contributing. By adding to that the influence of *multidisciplinarity*, with all of the attendant methodological, paradigmatic and classificatory concerns that influence and cross boundaries, it is not difficult to see how chaos may well be imminent. Fortunately, close collaboration, and continuous knowledge sharing amongst the researchers and developers, helped to obviate such a situation occurring. Despite this, situations of uncertainty prevailed. I recall once instance

where our project manager at CRI discovered that two separate and very different prototypes were under development, contrary to the contract and agreement regarding prototype development in PRACTITIONER. The project groups and managers discussed the situation, and agreed that two prototypes would be better than one. The situation was, at the outset, a competitive one with the question being asked: "which prototype would qualify as the "real" project outcome?" The result was, however, a significant innovation in terms of new tools for software development and reuse (Mili et al. 1994).

Overall, the project groups at CRI experimented with many approaches to knowledge organization, from faceted classification to computational linguistics and automatic indexing and term extraction for thesauri.

Yet the proposition at this point is: We did not do domain analysis.

While the projects were domain-specific: software development for diverse work domains ranging from libraries to the space industry and medical informatics and robotics, we also developed knowledge organization systems (KOSs) for specific domains like software. Still, it needs to be highlighted that the development of knowledge organization systems for specific domains is not, in and of itself, a domain analysis.

3.0 Where are the domains?

My second proposition is that there is no ready-to-hand domain "out there" to be discovered and colonized by a domain analyst for the design of a knowledge organization system, which will then make it possible for the knowledge organizer to "rule the land." There is no all-knowing analyst. The domain is not a kingdom, nor is it a republic. On the contrary, for the design of a KOS, the domain is, in the terminology of Schmidt and Wagner (2004, #) a "field of work" for several interested parties (for instance, professional organizations, libraries or KOS researchers) that are concerned with the development of specific knowledge fields or tools. As a "field of work" the domain is constructed in and through the process of planning, design and construction of a particular KOS. I am aware of the apparent constructivist agenda underlying this statement, but in alignment with Schmidt and Wagner, this is a pragmatic view of design.

My work on domain analysis for classification of software explores the knowledge domains of software development and reuse from different perspectives and knowledge interests in computing disciplines—in a narrow sense— as well as more broadly, in the humanities and social sciences. There was no predefined itinerary for the journey. At the outset, it was not possible to set out a detailed plan. Figure 1 represents a map of the actual journey into the domain of software reuse, as presented in Albrechtsen 1992a and Albrechtsen 1992b.

The mission took off from a bibliometric island from where I could view the publications on software reuse, in relation to the disciplines and topics involved (step 1 in figure 1). The results led to a new step: a focus on the state-of-the-art for software reuse in the most important disciplines involved: computer science and software engineering (step 2 in figure 1). In particular for computer science the study identified quite diverse paradigms or knowledge interests, from a technical interest that sought to create perfect and fail-safe tools to a more liberating interest that involved creating IT tools that would set free creativity in individuals and groups (step 3 in figure 1). That last knowledge interest was not mainstream for software development in 1992, with the exception of Scandinavian schools of software development and the developing field of computer supported cooperative work (CSCW). As my research focused on classification the question now became: "what kinds of KOS tools and theory would be relevant to developing a method for

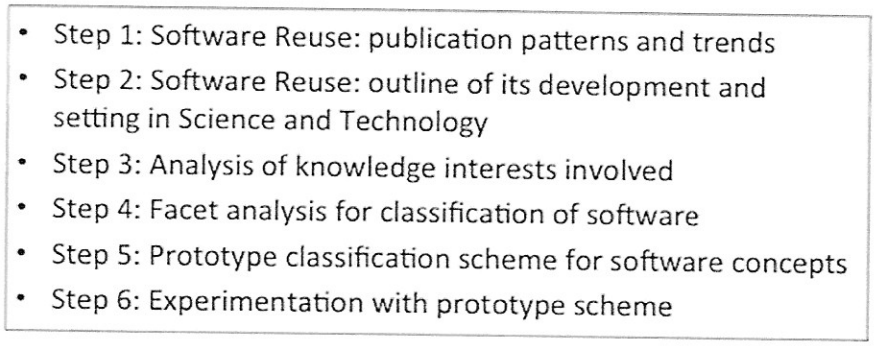

Figure 1. Main Points of the Journey into Domain Analysis for Software Reuse.

(This model was presented in Albrechtsen 1992a and 1992b, and it was subsequently used for instruction of KOS design at the Royal School of Library and Information Science, Copenhagen and Aalborg)

classification of software?" The journey continued to classification theory and to the use of faceted classification for the "domain" of software (step 4). Prieto-Diaz's work (1991) was an important inspiration. Prieto-Diaz applied and translated Ranganathan's generic model of facet analysis and classification for what he termed "domain analysis," as a systematic methodological approach for knowledge organization. The meeting with Diaz's work is the first encounter with an important computer science theory that integrates system development with an important theory developed in library and information science: Ranganathan's theory of faceted classification.

I chose the model of faceted classification for classification of software because of its flexibility and openness to concepts and with a view toward the dynamic developments then occurring in the fields of computer science and software engineering. The next key destination of discovery became therefore the concept of "facet" and understandings of "facet" in information science (step 4 in figure 1). This phase proved to be a rather complex part of the journey, presumably because facet analysis was my specialty at the time. I realized that in order to complete the journey I would have to construct a facet classification scheme for software (see Figure 2).

(For a detailed introduction to each facet, see Albrechtsen 1992a, 33-52). The resulting scheme was a prototype that included the key aspects investigated (step 5 in figure 1). The scheme was finally put to a walkthrough evaluation for two very different software concepts: The Book House tool for indexing fiction novels, and the word processor program Word Perfect's "Save" function (Step 6 in figure 1). The conclusions from these modest first tests were not that the analysis approach was proven right or wrong. The aim was to invite the new method into the reality of a prospective user.

4.0 Here are the domains: concluding remarks.

Domain analysis has become an established concept in information science and it covers the field quite broadly. Domain analysis was scaled up from the KOS journey on software reuse to a comprehensive methodological framework in information science by Hjørland and Albrechtsen (1995). From the outset, the aim was to present a methodological alternative to the information processing paradigm in information science, which was concerned with developing models for information retrieval (IR) and automatic indexing. Despite, or perhaps, because of the widespread use of the term "domain analysis" in IS education and research, there are varied understandings and accounts of its history and meaning. Domain analysis appears to have acquired interpretive flexibility (in the terminology of SCOT, Social Construction of Technology). The growing amount of variations in meanings and use might indicate a certain strength of the concept to prevail across interpretations and uses, almost like a boundary object (in the terminology of Star 2010). At the same time, that could also indicate the implicit view that "anything goes" in domain analysis—or pluralism, in a philosophical sense. Lopez-Huertas's article (2015) on interdisciplinarity and domain analysis is a highly relevant case in point: if domain analysis is restricted to scientific knowledge domains or professions like psychology and medicine, then how can we develop domain analysis for multidisciplinary fields like women's studies and IT systems development?

The approach to domain analysis for classification of software was presented and discussed at international conferences and in journals during 1992-1993 (e.g., Albrechtsen 1992b; 1993). In hindsight, I realize that the first research on domain analysis for classification of software built on an implicit thesis that software devel-

- Facet 1. Scientific paradigm
- Facet 2: Method/technical paradigm
- Facet 3: Scope or ambition of software concept
- Facet 4. Type of application
- Facet 5. Form
- Facet 6. Technical environment in terms of operating systems and hardware
- Facet 7: Application area
- Facet 8. Task area
- Facet 9. Development stage

Figure 2. Main Facets of Prototype KOS for Software Reuse.

opment is, or can be, a kind of knowledge creation. That system development is fundamentally a translation amongst diverse interests and parties.

The journey in the early days of domain analysis was a modest one. In Haraway's terms I was a modest witness and participant of KOS research in the light of AI and expert systems. My colleagues and I never developed 6th generation computing. However, we developed new ways of working with science and development, in multidisciplinary teams and groups, based on entrepreneurship and knowledge sharing. I never found the promised land of "domain." Domain analysis is a method. Following Law (2004), I believe it is important to point to how a method not only describes realities, but also is intimately involved in creating them. "Domains" are not terrains out there, waiting to be described and analysed by the initiated few. Fundamentally, we may all create them.

References

Albrechtsen, Hanne. 1993. "Subject Analysis and Indexing: from Automated Indexing to Domain Analysis." *The Indexer* 8 no. 4: 219-24.

Albrechtsen, Hanne. 1992a. "Domain Analysis for Classification of Software." Thesis. Royal School of Library and Information Science. Available at Researchgate or from the author (e-mail: hanne.albrechtsen@hum.ku.dk).

Albrechtsen, Hanne. 1992b. "Domain Analysis for Classification of Software." In *ASIS '92: Proceedings of the 55th ASIS annual meeting, 29.* Medford (New Jersey): American Society for Information Science and Learned Information, pp. 2948-63.

Albrechtsen, Hanne. 1991. "PRESS: a Thesaurus Based Information System for Software Reuse." In *Classification Research for Knowledge Representation and Organisation: Proceedings of the 5th International Study Conference on Classification Resea (FID)*, ed. Nancy J. Williamson and Michelle Hudon. Amsterdam: Elsevier, pp. 137-49.

Albrechtsen, Hanne. 1990. "Software concepts: Knowledge Organisation and the Human Interface." In *Tools for Knowledge Organization and the Human Interface: Proceedings, 1st International ISKO-Conference, Darmstadt, 14-17 August 1990*, ed. Robert Fugmann. Advances in knowledge organization 2. Frankfurt: INDEKS Verlag, pp. 48-63.

Frakes, William B. and Kyo Kang. 2005. "Software Reuse Research: Status and Future." *IEEE Transactions on Software Engineering* 31: 529-36.

Hjorland, Birger and Hanne Albrechtsen. 1995. "Toward a New Horizon in Information Science: Domain Analysis." *Journal of the American Society for Information Science* 46: 400-426

Law, John. 2004 *After Method: Mess in Social Science Research.* 1st ed. International Library of sociology. London: Routledge.

López-Huertas, María J. 2015. "Domain Analysis of Interdisciplinary Knowledge Domains." *Knowledge Organization* 42: 570-80.

Mili, Hafed et al. 1994. "Practitioner and SoftClass: A Comparative Study of Two Software Reuse Research Projects." *Journal of Systems Software* 25:147-70.

Prieto-Diaz, Ruben. 1991. "Implementing Faceted Classification for Software Reuse." *Communications of the ACM* 34 no. 5: 88-97.

Sarasvathy, Sara.D. 2009. *Effectuation: Elements of Entrepreneurial Expertise.* Northampton (MA): Edward Elgar Publishing.

Schmidt, Kjeld and Ina Wagner. 2004. "Ordering Systems: Coordinative Practices and Artifacts in Architectural Design and Planning." *Computer Supported Cooperative Work (CSCW)* 13: 349-408.

Sedwell, Ian, Ulla Kaaber and Hanne Albrechtsen. 1988. The Linguistic Analysis of Unix On-Line Documentation, Unrestricted, P1094-BrU-WPC4-Working Paper-8814 (Technical Report).

Star, Susan Leigh. 2010. "This is not a Boundary Object." *Science, Technology & Human Values* 35: 601-17.