Creation and Management of Community-specific Knowledge Domain Taxonomies

[1] Gerald Eichler, [2] Roland Schwaiger

Telekom Innovation Laboratories
Internet & Services: Information Relevance
[1] Deutsche-Telekom-Allee 7, D-64295 Darmstadt, Germany
gerald.eichler@telekom.de

[2] Ernst-Reuter-Platz 7, D-10587 Berlin, Germany
roland.schwaiger@telekom.de

Abstract: The semantic understanding of domain-specific terminology typical for communities, like a scientific conference, requires both, the analysis of daily spoken and written language in documents. A living taxonomy can help to cluster knowledge and classify questions against it, as well as to build up profiles of expertise. Text analysis modules support the parsing of training materials to extract a significant set of key phrases. However, the creation of a balanced categorisation tree requires the experience of community members. The *Spree* project introduces an OntoEditor™ to optimize automatic and manual processes to setup an expert finding & communication application, following well-known Web 2.0 paradigms.

Keywords: Web 2.0, semantic analysis, domain-specific taxonomy, knowledge domain design.

1 Community- and Domain-specific Terminology

When talking about a dedicated technical project, the used language follows a specific terminology. The applied vocabulary is often characteristic for a set of different projects within the same domain, called a *knowledge domain*. Clear definitions are required to create a common understanding among project partners, who form a loose *knowledge community* [Sch00]. Efficient text analysis assists the classification of new text-based materials and the identification of the best expert of an online community [MBA10].

It is also characteristic that written language, used e.g. in specifications, documentations, presentations and manuals, is different from the daily spoken language. Due to self organization processes, the latter is growing with the community and requires much more flexibility and tolerance in interpretation [Ei11].

Starting with a discourse on ontology and taxonomy, their application for knowledge domains is investigated in chapter 1. Chapter 2 illustrates the creation of a new taxonomy, taking the I^2CS as example. Basic linguistic methods are discussed in chapter 3, while chapter 4 introduces the OntoEditor™ as a Java application for knowledge domain management. Finally, in chapter 5 practical experiences are summarized.

1.1 Taxonomy vs. Ontology

There are different methods to preserve knowledge of a domain. Classical document management systems (DMS) support the structured storage of formal documents. However, a much more extensive part of information is exchanged via email, wiki, chat and blog in a highly informal manner. To combine these sources, both, linguistic and semantic tools, are required as known from machine learning [WFH11].

Linguistic modules (LM) help to normalize terminology and extract knowledge from unstructured texts. *Semantic modules* (SM) support to handle complexity, scalability, relations and object types to create a representative semantic network for a certain knowledge domain.

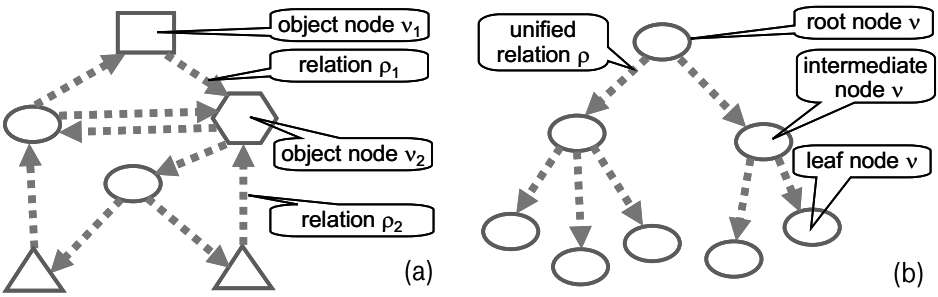

Figure 1: Ontology (a) and taxonomy (b)

As a directed graph, an *ontology* fulfils these requirements excellently, especially the representation of different object types and relations among them. However, it would be very difficult to create the one representative ontology for a certain knowledge domain. The knowledge of a domain grows evolutionary. Therefore, the metaphor of a tree offers key requirements for a hierarchical structure:

(1) New branches and leafs can be added {create}
(2) Existing branches can be split {fork}
(3) Old branches and leafs can die {delete}
(4) Branches or leafs can be bundled {merge}
(5) Situations can be compared and evaluated on similarity {match}
(6) Situations can be represented by multiple branches and leaves {classify}

Requirements (1) to (5) are perfectly met by a *taxonomy*. All edges represent the same relation ρ e.g., "consists of" or "is derived from", depending on the direction of view. All nodes, respectively root node, intermediate node and leaf node, are of the same object type ν, see Figure 1(b). By adding the multi-classification option (6), taxonomies are a simple but powerful means to describe a knowledge domain. Starting from a root, which represents the knowledge domain itself, thematic first level nodes are designed. In semantic context such nodes are called *concepts* and single words and phrases which represent the node, are called *terms*. A concept $\mathfrak{C}$ is represented by:

(A) a **unique identifier** as self-describing and compact node name[1] [mandatory]
(B) an **implicit list** of a set of weighted, machine extracted terms [mandatory]
(C) an **explicit list** of manually added terms as keywords/-phrases [optional]
(D) **sub-concepts**, representing derived lower level (child) nodes [optional]
(E) an **explicit description**, to detail the node context[2] [optional]

The implicit description (B) is also known as *term statistics*. Terms are sorted by its term frequency. The size of a taxonomy is not limited, neither in number of levels nor in number of nodes per level. It is recommended to design the main branches orthogonally in their context to minimize tree overlapping and to maximize independency.

1.2 Exploitation of Taxonomies

The initial node structure of a taxonomy is usually generated by a knowledge domain expert manually. The simplest way is to setup a file system tree, where folders represent the concept identifier (A) and files contain comma or line separated terms (B, C). Once a taxonomy has been created, it is be trained by expanding and re-calculating the term statistics and extended by adding new concepts (1, 2) or modifying them (3, 4).

The knowledge specific taxonomy is exploited in several cases by multi-classification:

Text classification. Any kind of text document including web pages, wikis, chats, emails and simple questions is classified against an existing taxonomy by identifying the nearest concepts.

Profile storage. A community member indicates his or her expertise by selecting the most appropriate concepts from the given taxonomy. Such a profile is represented as an instance of the taxonomy.

Object matching. For a given text fragment, similar text fragments or associated profiles are identified. The degree of similarity can be quantified by an average numeric parameter, calculated over all classified nodes.

Node balancing. By normalization of node weights, a well-adjusted topic assignment is reached. This is required to overcome asymmetries of the availability of training materials.

2 The I²CS Taxonomy

There is a wide range of domain specific knowledge areas. Beside scientific communities, even an event can frame such a knowledge domain. Here, as a domain

[1] The identifier is considered implicitly to be keyword.
[2] This text supports GUI implementations i.e., can be displayed on the mouse-over event.

specific example, the annual International Conference of Innovative Internet Community Systems[3] (I^2CS) is selected.

2.1 Main Branches

Following the history of I^2CS from the very beginning in 2001, three main topic areas were announced. Taking the official Call for papers (CfP) of 2012, they are good candidates for the first level concepts of the I^2CS taxonomy:

- Foundations – Theories, models, algorithms for communities
- Technology – Distributed architectures and frameworks
- Applications and socialization – Communities on the move

The concept identifier (A) should be compact, self-describing and of the same grammar type, singular nouns are preferred. To avoid losing sub-titles, they can be either moved as keywords to the explicit terms (B) or placed as second level nodes. Part (B) is enriched by general applicable descriptors and used synonyms, indicated in italic in Table 1.

To reach a unique and balanced appearance, it is good practise to divide the third bullet into two main branches. So far, there are four main branches with regard to contents: "Foundation", "Technology", "Application" and "Socialization" in Figure 2.

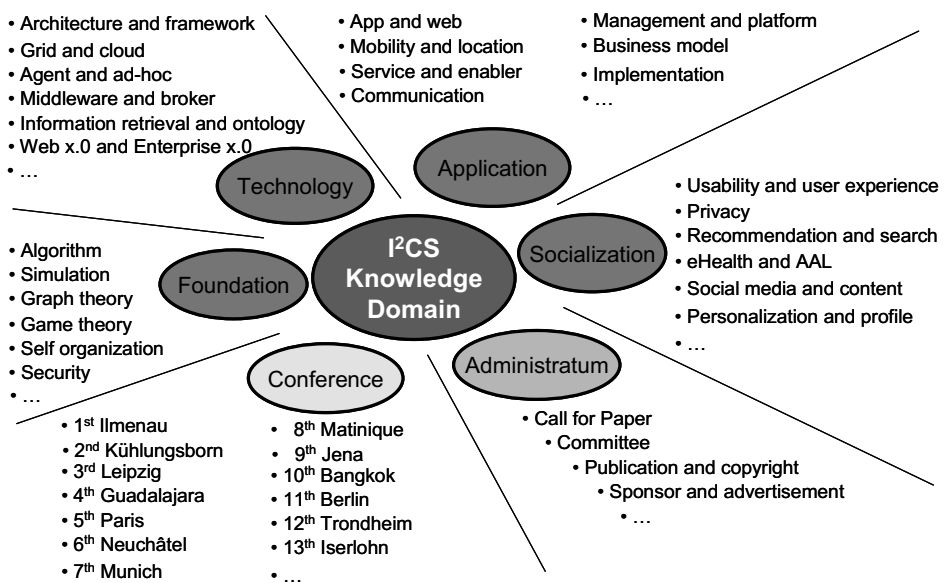

Figure 2: Initial I^2CS knowledge domain taxonomy

[3] URL: http://www.i2cs-conference.org/

The second important source is the event itself, in our case the "Conference". Relevant descriptors here are years and locations, countries and companies as well as the names of participants and chairs.

Semi-permanent information e.g., committees, organisation support go a third type of orthogonal main branch. Let's call it "Administratum".

Table 1: Initial result of the manual I²CS taxonomy approach

1st level node ID	Explicit terms keywords/-phrases	2nd level node ID candidates	Training materials
(A)	(C)	(D), see Figure 2	Source for (B)
Foundation	theory, model, algorithm, *basics, simulation, graph, security*	cfp sub-bullets	related abstracts, authors, submissions, talks, keynotes
Technology	architecture, framework, *middleware, cloud, grid, agent, SOA*	cfp sub-bullets	related abstracts, authors, submissions, talks, keynotes
Application	*service, implementation, search, applet, app, SaaS*	cfp sub-bullets	related abstracts, authors, submissions, talks, keynotes
Socialization	community, *wiki, blog, social media, mobility, privacy, society, profile*	cfp sub-bullets	related abstracts, authors, submissions, talks, keynotes
Conference	*I2CS, IICS, event, venue, program, best paper, panel, session, presentation, talk*	1st, 2nd, 3rd, …, 12th	annual documents (web page, cfp, flyer, poster, program, proceedings foreword)
Administratum	*information, contact, FAQ, call, proceedings, chair, submission, org, registration*	committee, publication	general documents, names, affiliations

2.2 Training Materials

Training materials are any kind of electronically stored documents which contain concept relevant text information for both, the scientific branches and the administrative branches. I²CS related examples are given in Table 1.

There is no need to provide structured materials, as the Language Analysis Module (LAM) will be in charge of extracting text and calculate the term statistics. Sources can be either text files (.txt, .doc, .pdf), or web based URLs (.html, .url, .lnk). Unfortunately, the quality of training materials varies. Therefore, it is preferred, to identify a huge quantity of documents from different sources per node. Missing materials on public topics can be filled by public glossaries e.g., Wikipedia[4], Wikibooks[5], Wiktionaries[6].

[4] URL: http://en.wikipedia.org/
[5] URL: http://en.wikibooks.org/
[6] URL: http://en.wiktionary.org/

3 Automatic and Manual Classification

In the linguistic context, *classification* means, to discover the closest concepts from a provided taxonomy for a given text in a given language.

3.1 Language Analysis Module (LAM)

The Language Analysis Module (LAM) is the key component for the linguistics. It is dictionary-based and therefore language-specific. Dictionaries for English and German were taken from the Weka toolset [We12].

Document type specific converters are responsible for extracting any context relevant text as unstructured stream from the source documents. Beside text documents, any type of Microsoft Office™ documents are supported by specialised converters. By stemming and n-gram building, the document is given a unique footprint with the creation of its term statistics. Figure 3 shows this step-by-step process.

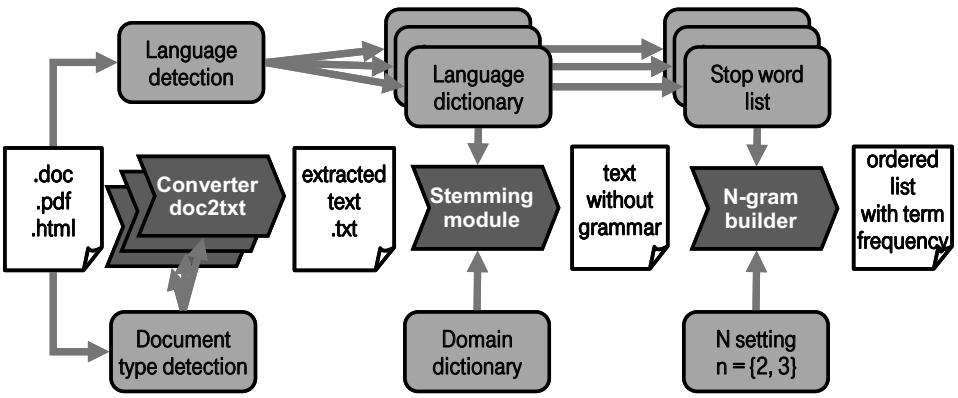

Figure 3: Text analysis process by LAM

Stemming compasses the replacement of nouns by its nominative singular, of verbs by its infinitive present and of adjectives by its neutral singular form. Any capitalisation is ignored and replaced by lower case letters. *N-gram building* comprises the splitting of sentences into semantic phrases of n words. Useful splitting indicators are stop words e.g. conjunctions and prepositions. Meaningless words e.g., articles are dropped. A typical n-gram is the combination of adjective plus noun.

3.2 Design Methodology

The creation phase of a new domain specific taxonomy (see section 1.2) is followed by the training phase (see section 2.2) and applied in the operation phase for matching. Figure 4 illustrates this process.

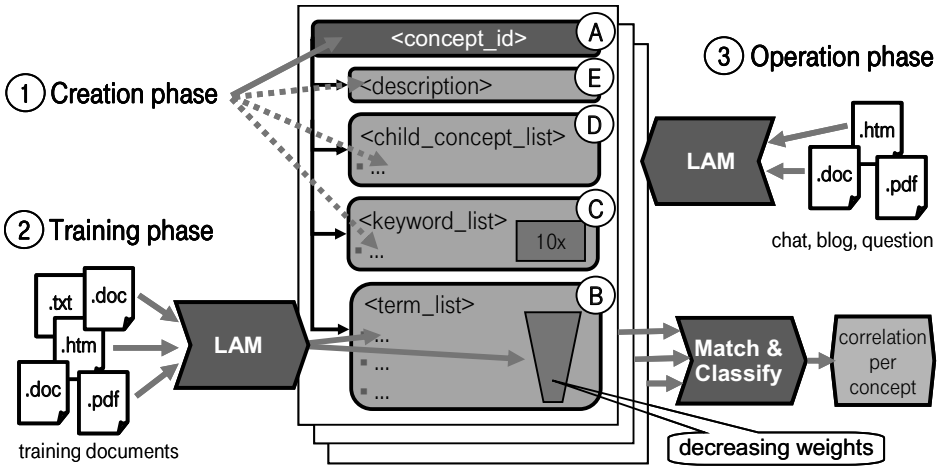

Figure 4: Taxonomy design methodology

Four steps lead to the starting point for the application of a taxonomy to reach the goals of section 1.2:

1. initial structuring of knowledge by a domain expert
   ⇒ basic tree skeleton
2. development of a common understanding by a well-prepared experts' workshop
   ⇒ rich keyword collection
3. manual document classification by single experts
   ⇒ balanced text to node assignment
4. automatic document analysis by machine learning algorithms
   ⇒ ordered, weighted term statistics

4 The Spree™ OntoEditor™

Within the Spree™ project (see section 5.3) an intuitive taxonomy management tool was developed. Even though it handles knowledge taxonomies, in honour to the initial approach it is still called OntoEditor™.

4.1 OntoEditor™ Concept

To allow non-IT community experts to manage and maintain the initial taxonomy, a GUI-based taxonomy editor, the OntoEditor™, was designed. To meet the KISS[7] paradigm, an explorer-like Java application was implemented.

[7] KISS … keep it short and simple

The OntoEditor™ can access any running Spree™ instance on the web by specifying its IP address and providing valid credentials for the role of an “ontology administrator” for that instance. Chances are invoked after an export of the new taxonomy by the “ontology administrator” followed by a restart of the application by the “system administrator”.

Figure 5: Layout of the OntoEditor™

The left column shows the concept tree and allows adding, moving or deleting concepts (nodes) from the taxonomy. Name, description (mouse over event) and explicit keywords for the selected concept are displayed on the upper right. They main window is tab-controlled with tabs for document association, term list, term frequency and the term frequency diagram as shown in Figure 5.

4.2 OntoEditor™ Implementation and Application

If not yet locally available, the Java application can be downloaded from a web site by invocation of a Java start script, including certification. It comes as a zipped set (17 MB) of .jar files.

For the storage format of a taxonomy, several ideas were discussed. The Resource Description Frameworks (RDF)[8] is a common method to define and store ontologies. The Web Ontology Language (OWL and OWL2)[9] by W3C is even more powerful. With Protégé an excellent editing tool is available [KnMu04]. However, to follow KISS again, a standard file system was preferred for the implementation. There are three types of objects:

- Hierarchical folders: representing the concepts
- Document files: containing the trainings materials of the folder above
- Index files: containing IDs, keywords and extracted term statistics

5 Lessons Learnt and Future Dimensions

Both, the generation of knowledge domain specific taxonomies and the development of the Spree™ community application, generated many interesting insights and experiences. This chapter addresses very compact: the community view, the ontology administrator view and the application view.

5.1 People and their Needs for Information

Enterprise and web communities. They have different requirements. Therefore on the one hand, the GUI of an expert search and communication application needs specific adaptation (Figure 7) [REK+11] while the matching core will be the same. On the other hand, the introduction of a community application requires different methodology [Ei11].

Time and duration horizon. The support of a single event requires different organisational and expert competencies than to a long living expert forum. While a single event focuses on management and administrative stuff (How to/FAQ), an expert forum concentrates on contents (a stable knowledge base). The I²CS taxonomy will fill a gap to support a regular event series.

Simplicity. Enterprise communities tend to reflect their administration overhead. However, communities have none or flat hierarchies by its nature. Therefore, the number of roles in an expert system should be strictly limited. As depicted in Figure 6, two roles, the questioner and the expert, are sufficient. This is simplified even more by a single rule: a questioner becomes an expert by creating a personal profile.

Language tolerance. The correctness of language in paper work is decreasing. Moreover, the style of chat and emails is even worse regarding linguistics. Therefore, spelling, grammar and semantic tolerant LAMs are required.

---

[8] URL: http://www.w3.org/TR/rdf-concepts/ (2004)
[9] URL: http://www.w3.org/TR/owl-semantics/ (2004) and http://www.w3.org/TR/owl2-overview/ (2009)

Language redundancy. Firstly, the terminology of experts is different from the language of questioners, as examples in medical domains illustrate. Secondly, the preciseness differs in the comparison of scientific world vs. daily topics.

Re-classification. Community members are asked to re-classify already automatically indexed materials. This will sharpen the selected classifiers.

5.2 Parameters to play with

To optimize the exploitation of a taxonomy, there are many tuning options. However, most of them are only applicable as the size increases. It is important to keep the balance of the size of the taxonomy and the size of the community. The following bullet items summarize shortly practical findings on issues and their potential solution:

- Creation of new nodes: differentiate only if different experts are to be addressed.
- Number of taxonomy levels: follow the Rule of Seven[10].
- Poor training documents: compensate by increased weight of keywords[11].
- Unbalanced set of training documents: taxonomy needs weight normalization.
- Profile fine tuning: introduce skill levels only in large systems.
- Term inheritance: decrease weights of terms derived from child to parent node.

5.3 LAM Implementation in Spree™

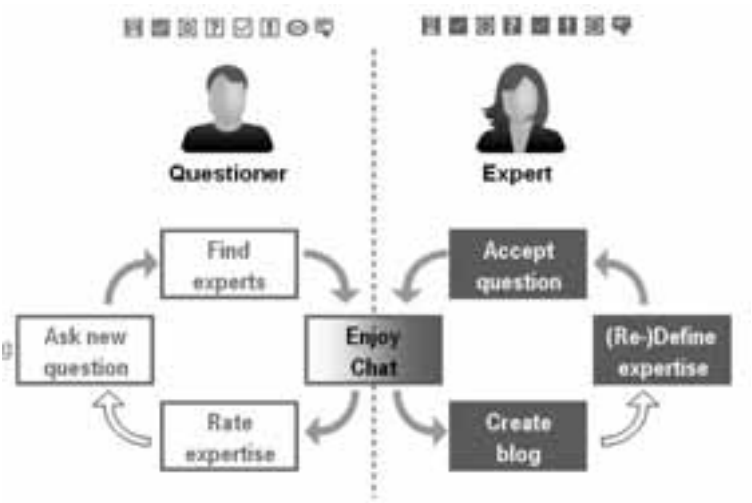

Figure 6: The Spree roles, basic principle and control loops

[10] The Rule of Seven says: limit any human-readable enumeration to a maximum of seven bullet items.
[11] To assign them a term frequency of ten, has been a good practise.

Spree™ is a web application which supports its registered participants in establishing a public (Web 2.0) or corporate (Enterprise 2.0) community [REK+11]. Members will be able to "meet" new colleagues depending on their expert profiles, get the possibility to chat with them and thus extend their contacts. Spree combines linguistic analysis, expert matching, notification, chat, blog and rating functions to initiate and archive asynchronous online discussions.

Spree™ was developed in close corporation of Telekom Innovation Laboratories[12] and the CC IRML of DAI-Labor at Technische Universität Berlin[13]. The matching core was applied in the "Spree – the knowledge exchange network" application for several domains e.g., TechnlogyRadar, system integration testing, customer care services, media recommendation and medicine. A representative example for a scientific community is the implementation for the Lindau Nobel Laureate Meetings, see Figure 7(a) [Ei10]. Figure 7(b) shows the login page of a Spree™ instance for pain management. To optimize the user expectations, a public web survey was established[14].

(b) German skin: information & login page

(a) English skin: personal home page

Figure 7: Spree™ GUI for LNLM (a) and pain management (b)

[12] URL: http://www.laboratories.telekom.com/
[13] URL: http://www.dai-labor.de/en/irml/spree/
[14] URL: http://schmerzlotse.de/

A Spree™ instance for networking among I²CS participants over multiple years is under development and will be soon part of the permanent conference URL at http://www.i2cs-conference.org/ for the upcoming events. Therefore, the presentation of this contribution will contain a discussion (experts' workshop) on future I²CS contents. Any proposals can be sent by email to taxonomy@i2cs-conference.org.

Acknowledgements

Many thanks to Susanne Wieczorek (Foundation Lindau Nobelprizewinners, Lindau), Thomas Strecker (adesso AG, Cologne), Robert Wetzker (aklamio UG, Berlin), Winfried Umbrath (GameDuell GmbH, Berlin), Ernesto William de Luca (University of Applied Sciences Potsdam) and Andreas Heck (CEO DIAmed GmbH, Bensheim), who supported intensively the transfer activities of several Spree™ implementations after the research and development phase.

References

[Ei10] Gerald Eichler: SPREE@Lindau – The Lindau Nobel Laurate Meetings 2010 scientific community. URL: http://www.spree-lindau.de/downloads/Spree-Manual-2S.pdf, June 2010.

[Ei11] Gerald Eichler: Self Organization in Enterprise 2.0 Communities: To introduce a new Experts' Exchange Application. In Autonomous Systems: Development and Trends. Springer Studies in Computational Intelligence, Vol. 391, pp. 189-200, Heidelberg Berlin, November 2011.

[KnMu04] Holger Knublauch, Mark A. Musen: Editing Description Logic Ontologies with the Protégé OWL Plugin. In Proceedings of the 2004 International Workshop on Description Logics (DL2004), pp. 70-78, Whistler, June 2004.

[MBA10] Florian Metze, Christian Bauckhage, Tansu Alpcan: Social and expert search in online communities. In Semantic Computing, P. Sheu, H. Yu, C. V. Ramamoorthy, A. K. Joshi, and L. A. Zadeh, Eds., pp. 341-356. IEEE Press/Wiley, Hoboken, NJ, USA, May 2010.

[REK+11] Andreas Rederer, Gerald Eichler, Thomas Kury, Roland Schwaiger: Spree – Trial-based Improvements transferring an Enterprise into a Web 2.0 Expert Community Application. In proceedings of the 11th I²CS 2011, LNI P-186, pp. 93-104, Koellen, Bonn, June 2011.

[Sch00] Michael Peter Schmidt: Knowledge Communities. 1st edition. Addison-Wesley, Munich, 2000.

[We12] Open Source Sourceforge Project: Weka – Machine Learning Software in Java. URL: http://sourceforge.net/projects/weka/ Last accessed: April 2012.

[WFH11] Ian H. Witten, Eibe Frank, Mark A. Hall: Data Mining: Practical Machine Learning Tools and Techniques. 3rd edition. Morgan Kaufmann, January 2011.