CSC
CORPORATE SMART CONTENT

Corporate Smart Content

Requirements and Use Cases

Report I on the sub-project Smart Content Enrichment

Technical Report TR-B-14-02

Adrian Paschke, Ralph Schäfermeier, Kia Teymourian,
Alexandru Todor and Ahmad Haidar

Freie Universität Berlin
Department of Mathematics and Computer Science
Corporate Semantic Web

September 2014

Freie Universität Berlin

INNOPROFILE
UNTERNEHMEN REGION
Die BMBF-Innovationsinitiative
Neue Länder

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

Abstract

In this technical report, we present the results of the first milestone phase of the Corporate Smart Content sub-project "Smart Content Enrichment". We present analyses of the state of the art in the fields concerning the three working packages defined in the sub-project, which are aspect-oriented ontology development, complex entity recognition, and semantic event pattern mining. We compare the research approaches related to our three research subjects and outline briefly our future work plan.

Contents

Chapter 1

Introduction

This is the first report on the research efforts of the **"Smart Content Enrichment" (SCE)** sub-project of the InnoProfile-Transfer **"Corporate Smart Content (CSC)**[1] project funded by the German Federal Ministry of Education and Research (BMBF) and the BMBF Innovation Initiative for the New German Länder-Entrepreneurial Regions.

The CSC project is a follow-up project of our previous research project **"Corporate Semantic Web" (CSW)**[2] (2008-2013). The CSW initiative has made important fundamental contributions, for example the application of Semantic Web technologies in particular semantic search, the use of semantic technologies to support collaborative activities, the development of corporate-wide taxonomies and ontologies in companies and many others[3]. The practical relevance of these technologies has been exemplified in several cooperation projects with regional partners.

The term *"Smart"* in *"Corporate Smart Content"* refers to semantically enriched corporate data like text documents, web pages, images, videos, news articles, process documentation and corporate data sets that enables data processing machines to interpret the content and understand the relationships. The *"Smart Content"* builds the foundation for *"Smart Applications"* that make use of semantic information by targeted access to relevant content within a particular context.

For companies, content has become an important asset that either, in the context of corporate knowledge management, contains valuable knowledge about internal processes, people, products, markets, customer relationships and competition and thus forms the basis for strategic and operational decisions, or that represents the product with which the company operates on the market itself. The beneficial use of linking these valuable assets as well as the targeted access to the content, makes it essential for the competitiveness of today's companies.

In recent decades, the usage of business software such as Content Management Systems (CMS), Customer Relationship Management (CRM), Enterprise Resource Planning (ERP) and Corporate Wiki systems has lead to a strong increase of content generation in companies. This increase introduced two new challenges, the search problem for finding the relevant content at the right time,

[1] http://corporate-smart-content.de/

[2] http://www.corporate-semantic-web.de/

[3] http://www.corporate-semantic-web.de/technologies.html

and, on the other hand, the integration of data sets generated by different systems.

In recent years, Semantic Web research has produced new approaches, technologies and standards. One of the contributions is the Linked Data initiative which has the effects that many public institutions and enterprises published their data in semantic web compatible formats (RDF, RDF-S, OWL, RuleML/RIF) to be retrievable and used by the public. The integration of relevant data from the Linked Open Data (LOD) [4] cloud has great potential for generating added value in combination with internal corporate data sets.

However, with regard to the integration of these data into corporate data and processes, some fundamental problems still exist that require further research. The problems can be summarized in the following research questions:

- Selection of relevant data sets from the LOD cloud for the inclusion in corporate databases
- Valuation and ranking of trusted external data resources for the usage in corporations
- Semantic integration of external data sets in corporate ontologies, taxonomies and processes
- Usage of integrated data and meta data for the enrichment of corporate content (generation of smart content)
- Semantic integration of corporate live data streams and its usage in business processes

A further challenge is the steady growth of Linked Data, causing problems related to scalable storage of data and data availability for the targeted users. Furthermore, the logical expressiveness of ontology languages such as OWL is an obstacle for the usage of Semantic Web technologies in corporations.

The current state-of-the-art models for the development, extension and reuse of ontologies are still based on the direct involvement of both domain experts and modeling experts in the engineering process, which is not feasible for small and medium-sized businesses with limited resources. In most cases, the use of ontologies within an organization happens through the reuse and adaptation of existing freely available or commercially acquired ontologies. In this project, one of the research subjects that we are working on is the development of required methods for the aspect-oriented adjustment of ontologies.

Figure 1.1 shows the process chain of our project for the smart content generation in corporations. The main four abstraction layers of the content process chain are shown in the figure. The *"schema and ontologies"* layer builds the fundamentals for the generation of content in corporations and the *"Content, data and processes"* layer contains all of the activities related to the content acquisition, extraction and enrichment. The *"Aggregated Corporate Knowledge"* illustrates the complete aggregation and integration of corporate content gathered from different sources and in different formats. The highest abstraction layer are *"Use Cases"*, i.e., the usage of smart content in different corporate application domains.

[4] http://www.lod-cloud.net/

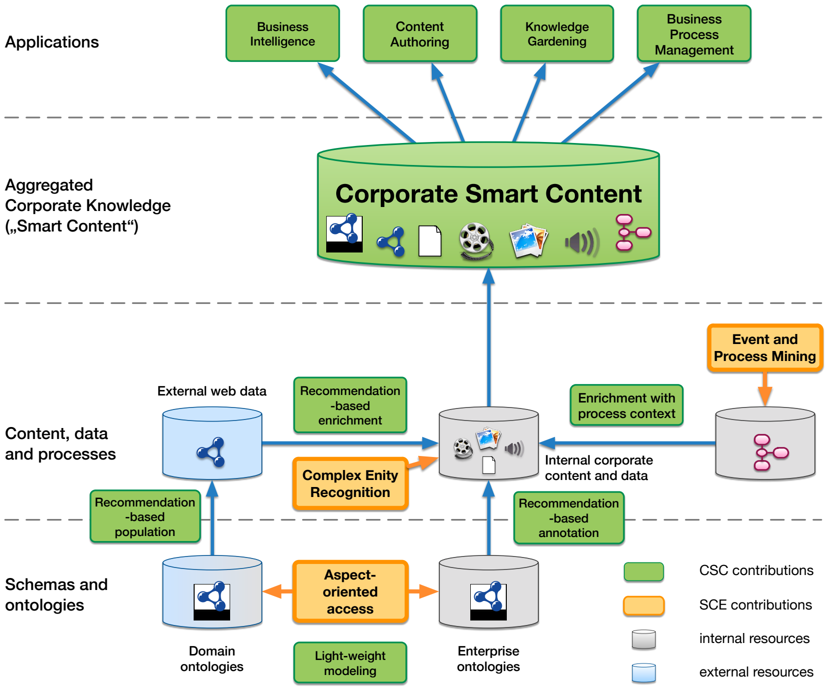

Figure 1.1: Corporate Smart Content Process Chain

In the *"Corporate Smart Content"* project we (Freie Universität Berlin) are focusing on the following three research topics within the smart content generation problem. Our research subjects are:

1. **Aspect-Oriented Ontology Development:** Ontology reuse and integration are mainly hindered by two factors: a lack of contextual information about the contents of an ontology, and a lack of means to identify, examine and reuse only the parts of an ontology that are useful in the context of the scenario at hand.

   In Chapter 2 we describe a possible solution to the two above mentioned problems in the form of a flexible and dynamic approach to ontology decomposition based on requirements from different stakeholders, inspired by the Aspect-Oriented Programming paradigm (AOP). We describe the main research problem that we address and the potential benefits of aspect-oriented ontology development.

2. **Complex Entity Recognition:** The state of the art approaches for named entities recognition can detect named entities from the text. Each of the extracted named entities represent a semantic object (aka. resource) that are in relation to other named entities. We can consider a complex entity as an object (or resource) that summarizes, represents, is composed of, or denotes a set of named entities.

   In Chapter 3 we describe our research challenge addressing the extraction of complex entities from text documents. We review related approaches and set out the benefits of the extraction of named entities from different textual documents. We also review approaches related to the annotation of named entities in text, methods of document enrichment and external knowledge sources that can be used in combination with named entity recognition.

3. **Knowledge-Based Mining of Complex Event Patterns:** Complex Event Processing (CEP) is an emerging technology to achieve actionable, situational knowledge from huge event streams close to real-time. Complex events can be detected by using event detection patterns which are specified by the business experts. The specification of event detection patterns is highly complex and requires knowledge about a wide business area. To support the specification of event detection patterns, we address the problem of automated extraction of complex events patterns.

   In Chapter 4 we describe the research problem and review the existing approaches for pattern detection which are primarily dealing with syntactical processing of event sequences to detect complex patterns only based on the sequences of events happening. We propose an extension of the existing approaches for pattern mining and describe our plans for the usage of ontological background knowledge to be able to extract complex event patterns based on the relations of events to the resources in the background knowledge.

Chapter 2

Aspect-Oriented Ontology Development

2.1 Introduction

The semantic web is a constantly growing network of facts about our world, and ontologies provide the basic truth for these facts. Ontologies are a highly expressive formalism for knowledge representation based on formal logic. The act of building an ontology is a laborious and complex task and subject of the research field of Ontology Engineering, the result of which are (variously rigorous) methodological approaches and tools for this task. Accordingly, from an ontology engineer's perspective, ontologies are also artifacts that stem from an engineering process and are bound to a life-cycle. As stated by Gruber in his seminal work on Ontology Engineering [29], the purpose of ontologies is to convey *shared* knowledge. Therefore, the *reuse* of existing ontologies is an integral part of the ontology life-cycle.

As reported in the final report of the Corporate Semantic Web project [47], ontology reuse and integration are mainly hindered by two factors:

1. a lack of contextual information about the contents of an ontology, and
2. a lack of means to identify, examine and reuse only the parts of an ontology that are useful in the context of the scenario at hand.

The first problem has been tackled by different approaches for describing ontology contents by metadata which are attached to the ontology and contain information about the contents and the provenance of the ontology, e.g., authors, engineering methodology, ontology editing tool, or the knowledge representation paradigm that was used for building the ontology. A prominent work in this area is the Ontology Metadata Vocabulary (OMV) [21].

In what concerns reuse of ontology parts, a significant body of work has been accomplished in the field of Ontology Modularization, which comprises methods for either partitioning existing ontologies into smaller and easier to handle parts (top-down approaches) or methodological (bottom-up) approaches to building modular ontologies from scratch.

In this report, we describe an approach to the two above mentioned problems by a flexible and dynamic approach to ontology decomposition based on requirements from different stakeholders, inspired by the Aspect-Oriented Programming paradigm (AOP). AOP allows for modularizing software systems based on cross-cutting concerns. We argue that this concept can be applied to ontologies and that this approach will yield the possibility to convey expressive context information to arbitrary ontology parts and a unified dynamic and flexible approach to modular ontologies.

2.2 Problem Statement

While ontology metadata help identifying the right ontology for the purpose of integration and reuse of its content in another ontology, they are not sufficient if only partly reuse is desired [50]. For example, the National Center for Biotechnology Information (NCBI) Organismal Classification (NCBITAXON) ontology[1] contains 847.760 concept definitions. Importing the entire ontology, even though only a small fraction of it might be relevant in the context of a particular application, will unconditionally lead to significant difficulties with regards to reasoning and query result retrieval performance, evolution and maintenance of the ontology, complexity management, and overall understandability by the reusing party [46]. Therefore, importing only a part of the ontology that is meaningful in the context of the application is desired. However, selecting the right subset of ontological entities is even more difficult than identifying the right ontology in the first place. What is missing, from our point of view, is a facility for providing metadata not only about an ontology but, additionally, on the level of (meaningful) fractions of its contents.

Ontology Modularization approaches aim at creating meaningful partitions of large ontologies. Top-down approaches are mostly algorithmic, with the criteria being used for selecting an ontology module are determined by the respective algorithm. Some of are parameterizable to a certain extent, permitting some degree of adaptation of the modularization criteria to the user's needs. However, the parameters often reflect the internal operational mode of the modularization algorithm rather than a definition of a meaningful ontology part from a user's point of view.

Bottom-up approaches, on the other hand, involve the ontology developers in the process of determining how to construct meaningful partitions of the ontology at hand. However, requirements of later potential users of the ontology might be diverse and hard to predict. Moreover, requirements concerning the modularization may change, even within the context of the same application. An application backed by a large and complex ontology might require a module that contains the full set of declarations of concepts and only their subclass/superclass relations for browsing the concept hierarchy. Another part of the application might require only a small, but fully axiomatized module for (topically restricted) complex queries and reasoning tasks.

In software engineering, this kind of multi-faceted requirements is referred to as *cross-cutting concerns* [2], since they emerge on different levels of an application

[1] http://bioportal.bioontology.org/ontologies/NCBITAXON

[2] As defined by the IEEE standard 1471 of software architecture [28], *"concerns are those interests which pertain to the systems development, its operation or any other aspects that are critical or otherwise important to one or more stakeholders".*

and reflect different points of view on and goals of a software system formulated by different stakeholders. Cross-cutting concerns are omnipresent throughout the system and cannot easily be encapsulated in separate modules.

2.3 Aspect-Orientation as a solution to multi-faceted module selection

A well-known approach tackling the existence of cross-cutting concerns in software systems is the *Aspect-Oriented Programming (AOP)* paradigm. AOP allows for moving references to external modules out of the application code into the respective modules and provides a mechanism for reconnecting them with the application code at runtime or compile time, leading to effective and flexible modularization of the entire system. The functionality encapsulated in such a module is referred to as an *aspect* so as to reflect to the different perspectives on the system by different stakeholders.

From our observations, and as explained above, cross-cutting concerns are a problem in shared ontologies as well [57] and [58]. As observed by Gruninger [30], different parties involved in the development or later usage of an ontology may have completely different assumptions about the conceptualization formalized in the ontology. We believe that introducing a formal specification of the assumptions behind a certain conceptualiization and attributing it to the relevant parts of an ontology help understanding and adaptation of ontology parts to the needs of diverse user groups.

For example, a vendor of photography gear may deploy an ontology about digital cameras. Different actors will have a completely different view on the product (see Figure 2.1). Potential customers see a digital camera as technical device and are interested in features such as chips size, pixel size, and file formats the device supports. The sales team, on the other hand, sees the camera in terms of a sales item, with features such as wholesale price, profit margin, and the number of units in stock. The fact that these properties are part of the conceptualization is based on requirements formulated from each stakeholder's point of view. Because these requirements concern the functinality of the ontologies, i.e., the concepts and relations the ontology is supposed to specify, they are referred to as *functional requirements.* Besides that, further requirements may exist that do not concern the actual functionality of the ontology but are still relevant in the context of the application. Examples for this sort of requirement are the need for provenance information, temporal attributions (e.g., validity periods for certain facts, such as temporary special offers), and reasoning efficiency (as outlined above). This type of requirement is referred to as *non-functional requirements.*

Each of these requirements or concerns affect a particular subset or module of the facts contained in the ontology. The modules might be overlapping, i.e., a certain fact might be associated with multiple concerns.

In an application, different concerns can become relevant or irrelevant at particular points in time. Using the notion of aspects from Aspect-Oriented Programming and applying it to ontology modules, it will be possible to dynamically adapt the ontology to the situational context, the user's point of view and current requirements, such as expressiveness vs. reasoning efficiency.

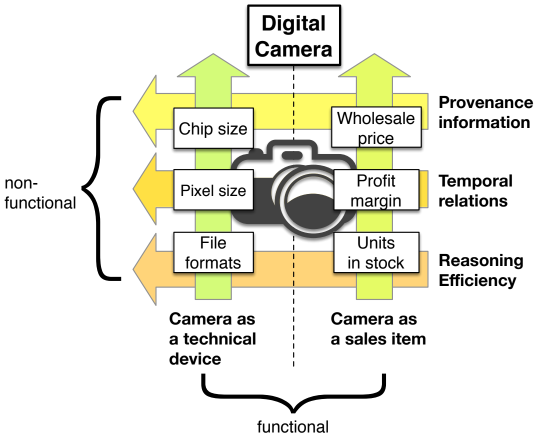

Figure 2.1: Cross-cutting concerns by the example of a digital camera. Different stakeholders are interested in different aspects of the core concept (camera). The different interests (concerns) reflect requirements formulated by each of the stakeholders. At the same time, stakeholder-independent requirements cross-cut the ontology. Each of theses requirements stems from a different level and has a different dimension.

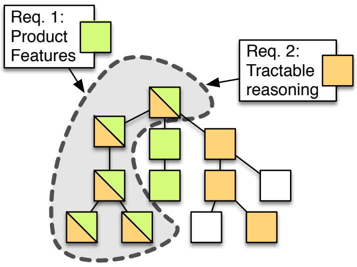

Figure 2.2: Selection of an ontology module that satisfies two cross-cutting requirements: It should only contain concepts of the subdomain "car components" of the car domain ("Engineering" aspect, dotted), and it should only contain constructs that allow for tractable reasoning ("Tractability" aspect, dashed). The resulting module (grey) only contains those constructs that are concerned by both aspects.

In the remainder of this chapter, we outline commonalities between ontology modules and software aspects and describe requirements and use cases for an aspect-oriented ontology development approach. We argue that such an approach enables (a) straightforward development of modular ontologies from scratch, and (b) flexible a-posteriori modularization, driven by user requirements.

2.4 Background and Related Work

Ontology modularization is an active research field, and there exists a rich body of related work. D'Aquin et al. [16] distinguish between different perspectives on the problem of which two different subfields have emerged.

First, there exist approaches to *ontology partitioning*, where a monolithic ontology is decomposed into smaller fractions. The motivation for ontology partitioning comes from requirements concerning maintenance and reuse, thus constituting requirements rooted in an engineering point of view. The second class of approaches is referred to as *ontology module extraction*. The motivation for module extraction is mainly selective use and reuse [16].

In [27], the authors present a partitioning approach using so called $\mathcal{E}$-Connections [15]. The criterion for the partitioning process is semantic relatedness. This is determined by checking the $\mathcal{E}$-safe property a structural constraint which avoids the separation of semantically dependent axioms in order to achieve semantically consistent modules. The relatedness of the different modules is retained by the $\mathcal{E}$-Connections. A drawback of this approach, however, is that it requires that modules are disjoint, and thus concept subsumption or the use of roles across different modules is not possible.

Schlicht et al. propose a semi-automated approach to ontology partitioning based on application-imposed requirements [59]. The method constructs a dependency graph of strongly interrelated ontology features, such as sub/super concept hierarchy, concepts using the same relations, or similarly labelled concepts. Then, the ontology is partitioned retaining strongly related groups in the same module. The method is parametrizable by specifying the features taken into account for constructing the dependency graph and the size a module should have in terms of the number of axioms.

Another class of partitioning approaches uses graph-based and social network metrics in order to determine central concepts and interrelated features which should be part of the same module [14].

Although there exist many more approaches to ontology partitioning the above examples demonstrate a shortcoming of current (semi-)automated modularization techniques. They lack flexibility, dynamicity, and a way to precisely define which parts of an ontology are effectively necessary in a particular scenario.

Approaches to ontology module extraction comprise logic-based extraction methods, for example [26], [36], [64] and [37]. These approaches are automatic and aim at producing self-contained, consistent ontology modules. They make use of logical properties such as semantic locality and inseparability.

A more unified approach to the problem of ontology module extraction and thereby similar to our work is the work of Doran et al. [25]. The authors propose using SPARQL queries in order to define ontology modules. They show that the more specialized approaches can be replicated in the form of SPARQL queries. The work of Doran et al. conforms with ours in what concerns the intensional module definition, in such a fashion that what we define as a pointcut is defined by Doran et al. in terms of SPARQL queries. It does not, however, include the possibility of extensional module definitions. In addition, SPARQL is an RDF graph based query language and neglects the semantics of DL-based ontology languages. Altogether, our vision behind this research is driven by the aim to enhance reusability of ontology modules. From our point of view, this does not only comprise the modules themselves, but also reproducibility of the modularization task. In order to achieve this, our approach makes the external module definition part of the model itself, by placing it at the same language level (as aspect-oriented languages do). In this way, ontologies can be shipped together with their aspect, i. e., module definitions. We argue that this approach improves traceability, comprehensibility, and thus reusability of ontologies and their modules.

A similar approach, using graph transformations and relying on user-defined graph-based extraction rules, has been proposed in [17].

While the latter two classes comprise approaches for *a posteriori* modularization of existing ontologies, a third arising class of methodological approaches aim at modular construction of ontologies in an *a priori* manner.

Related work in this area has been accomplished in the context the Neon project which provides a rigorous methodology for collaboratively building modular ontologies [3]

Another approach is described by [69], proposing a methodological framework for constructing modular ontologies driven by knowledge granularity. The proposed approach involves a separation into three levels: an upper ontology, modeling the theoretical framework, domain ontologies for reusable domain knowledge, and domain ontologies for application specific knowledge.

The shortcoming of existing modularization approaches is, as already mentioned in the introduction, their one-dimensionality, which is also acknowledged by [18] and [17]. The latter propose more unified approaches to the problem, however, they are restricted to the (graph-based) RDF model. Moreover, they lack formalisms of mapping modularizations to requirements, hindering relaying and reuse of module specifications.

Furthermore, as mentioned in Section 2.1, modularization requirements can be dynamic and subject to change even within the context of the same application. Therefore, we contend that a more flexible approach is needed that takes this dynamicity into account and allows for multiple, possible overlapping, modularizations of the same ontology and subsequent extraction of meaningful modules, custom-tailored to the current requirements.

2.5 Requirements for an Aspect-Oriented Ontology Development Approach

As mentioned in Section 2.1, an *aspect* in terms of aspect-oriented programming comprises a module encapsulating some particular functionality of an application and a description as to which parts of the application need to use the module.

Two central properties of AOP are *quantification* and *obliviousness* [19]. *Obliviousness* refers to the fact that all information necessary to determine the execution points where the application should make a call into an aspect module are contained within the aspect itself rather than in the application code. A developer of one module does not, and need not, have knowledge about other modules.

This information can be provided in the form of an exhaustive list of type signatures or in terms of *quantified statements* over type signatures. Expressed more formally, AOP uses quantified statements in the form

$$\forall m(p_1, \ldots, p_n) \in M : s(m(p_1, \ldots, p_n)) \rightarrow (m(p_1, \ldots, p_n) \rightarrow a(p_1, \ldots, p_n)),$$

where M is the set of all methods defined within the software system, s a predicate specifying a matching criterion, $m(p_1, \ldots, p_n) \in M$ a method adhering to the signature $m(p_1, \ldots, p_n)$, and $a(p_1, \ldots, p_n)$ the execution of the aspect with all the parameters of each method, respectively [62].

Accordingly, we define an aspect-oriented ontology module as follows:

Definition 1 (aspect-oriented ontology module) *Given an ontology $\mathcal{O}$ that consists of a finite set of axioms $\mathsf{Ax}_{\mathcal{O}}$, an aspect ontology $\mathcal{O}_A$ containing a set $\mathcal{A}$ of named aspect individuals, an aspect individual $a \in \mathcal{A}$ and a predicate*

[3] http://neon-toolkit.org/wiki/Main_Page

$\mathsf{hasAspect}$. *Then a module* $\mathcal{O}_a \subseteq \mathcal{O}$*, consisting of the set of axioms* $\mathsf{Ax}_{\mathcal{O}_a} \subseteq \mathsf{Ax}_{\mathcal{O}}$ *is an ontology module defined by the aspect* a *if* $\forall \mathsf{ax} \in \mathsf{Ax}_{\mathcal{O}_a} : \mathsf{hasAspect}(\mathsf{ax}, a)$.

Furthermore, it is possible for an aspect to be any expression in the same (or another) ontology language, allowing an aspect to be defined using arbitrary logical expressions. For example, an aspect could define a module with facts only valid during a specific period of time and consisting only of expressions within a tractable fragment of the ontology language (e.g., OWL $\mathcal{EL}$). The aspect would be the intersection of the aspect *ValidityPeriod* and another aspect *OWL_EL_Profile*.

In this manner, an ontology module can be defined either by its extension, i.e., by manually assigning ontology axioms to an aspect (and therefore a module) or intensionally by formulating a query (or several consecutive queries) specifying a set of common properties that should apply to all axioms that are supposed to be part of the module defined by an aspect.

Table 2.1 contains a list of further requirements to our approach to aspect-oriented ontology development.

2.6 Outlook

Based on the requirements described in Section 2.5, the next steps in our work toward an aspect-oriented ontology development approach will be as follows:

- As a next step, we will define the formal semantics underlying our approach and show its soundness and completeness.
- We will extend our approach to the following use cases based on the working plan of the Corporate Smart Content project:
  - Using aspects for modeling role-based views on content-based processes in enterprises.
  - Segmentation of ontologies by reasoning complexity using aspects
  - Provenance information as aspects.
  - Modelling multilingualism and intercultural perspectives using aspects
- In order to describe aspects, we will provide an aspect-ontology which defines the vocabulary and relations necessary to describe ontological aspects.
- A prototypical system for that allows aspect-oriented access to ontologies will be provided.
- We will integrate the aspect-oriented approach in the ontology life-cycle and extend the OntoMaven approach which has been a result from the Corporate Semantic Web project to allow ontology project developers to specify ontology aspects in their dependency definitions, tackling the above mentioned problem of selective reuse.

Functional Requirements		
#	**Req. ID**	**Description**
1	decomposition of cross-cutting concerns	The formalism should provide means for decomposing ontologies based on cross-cutting concerns by using aspects in order to attribute certain parts of the ontology to such concerns.
2	flexibility	The formalism must be flexible enough to express all kinds of (functional or non-functional) aspects.
3	self-descriptiveness	Aspect descriptions should be ontological entities (either in the same or a different ontology language as the base-ontology).
4	isolation	Aspects should not interfere with the semantics of the ontology they are added to. They should reside on a meta-level. They should **not** be first-class citizens.
5	combination	In order to embrace the problem of cross-cutting requirements, aspects must be combinable. It must be possible to assign to each axiom of an ontology an arbitrary number of aspects. During the ontology module selection stage, it must also be possible to select arbitrarily many aspects at the same time.
6	decidability	Should the approach be used in conjunction with an ontology language that is designed to only allow for decidable reasoning problems, then the aspect-oriented formalism should only introduce decidable reasoning problems as well.

Non-Functional Requirements		
#	**Req. ID**	**Description**
7	compatibility	The formalism must be compatible with existing knowledge modeling formalisms, i.e., ontology languages.
8	main module	The formalism must allow for aspect-oriented modularization independently of whether there exists a main module or not. If a main module exists, then it is identified by the fact that is not associated with any aspect.

Table 2.1: Requirements for aspect-oriented ontology development.

Chapter 3

Complex Entity Recognition

3.1 Introduction

Due to the advances in storage and processing capacity in the recent years, companies now have to deal with increasingly large amounts of data. These data sets can become so large and complex that traditional data processing applications can not deal with them, a phenomenon that is being called Big Data. Many internal decisions require information from all kinds of unstructured data sources such as text documents, spreadsheets, presentations and charts. Dealing with this information usually requires large amounts of manual labour and time.
The current state of the art in Information Extraction and the related field of Business Intelligence aim to address this problem by using Natural Language Processing and Machine Learning techniques in order to transform unstructured data into structured data and then try to derive useful information from the newly structured data.

3.2 Problem Statement

The issues that we address is in this research can now be substantiated in the following main problems.

- Is it possible to extract Complex Entities formed by multiple relations of different types between multiple entities of different types

- How to optimize the performance of concept learning approaches for Complex Entity Recognition

Identifying concepts or complex entities in natural language text that are composed by a series of n-array relationship patterns between different named entities.

Traditional IE approaches focus on identifying proper names in natural language text or relationships between the identified simple named entities. Such relationships however, usually involve just 2 participating entities.

Approaches that combine the relationships between multiple entities have been used for event extraction and detection. They are, however, limited in scope since they aim at discovering the presence of an event such as company default, terrorist attack, company merger etc.

We define as a complex entity that is not explicitly named in text but involves a combination of multiple relationships and indicators.

Examples of complex entities can be:

A sick person can be identified from patient reports, usually is not explicitly stated in the text but can be deduced by establishing the relationship between a specific named entity and a relationship of the type "has illness" and a specific illness. Furthermore the type of patient can then be further refined based on the illness he suffers from.

A victim of human trafficking, can be detected by mining police reports and looking for specific indicators that can be financial, interpersonal and temporal.

3.3 Ontologies for Natural Language Processing

One of the ways in which semantic technologies can improve the Information Extraction process, is to enable content editors to better annotate the information they produce by providing semantically expressive content annotation models. The use of such annotation models can improve the performance and precision of IE methods, by eliminating a large part of the preprocessing steps required and by providing better training models for supervised and semi-supervised machine learning approaches.

A further improvement can be achieved by using document enrichment approaches in the annotation step and combining annotations with existing semantic knowledge bases such as DBpedia, Freebase or Wikidata.

Automating this process proves to be a complicated task, and requires the use of NLP and Machine Learning tools as well as the efficient use of semantic models for document annotation and representation and the efficient use of internal and external knowledge bases for document enrichment.

One of the key elements for NLP applications in text documents are named entities, they give us an idea about the main individuals of interest(people, places and organisations) involved in those documents. However, early approaches either do not annotate the recognized information in the text or do so in a proprietary and non standard way. Having recognized this problem a series of approaches emerged that try to standardize the way named entities and other important elements in text documents get annotated and how those annotations are stored.

In this section we present some of the most important annotation ontologies with regards to their usage in natural language processing applications. We then proceed to compare these ontologies based on preselected criteria.

3.3.1 OAC: The Open Annotation Collaboration

The Open Annotation Collaboration(OAC)[56] is a W3C initiative that aims to standardise web annotations based on linked data principles. Annotations can be considered as an important part of the current World Wide Web as they make up a large part of it. One can consider all meta data about a resource to be an annotation such as comments, reviews, discussion threads etc. The OAC Data Model tries to provide a framework in order to express all these different types of annotations into a common data model, thereby enabling different platforms to share and reuse annotations.

In [55] the authors enumerate the main goals of the OAC data model, namely:

- provide a single consistent model that covers all types of annotations.
- reuse existing ontologies and standards in order to ensure interoperability with existing systems.
- reduce the implementation costs for all users of this standard in order to encourage adoption.
- provide an abstract model that does not have specific storage or modeling requirements.
- keep the triple-count of the serialized model low in order to enable an efficient communication between systems.

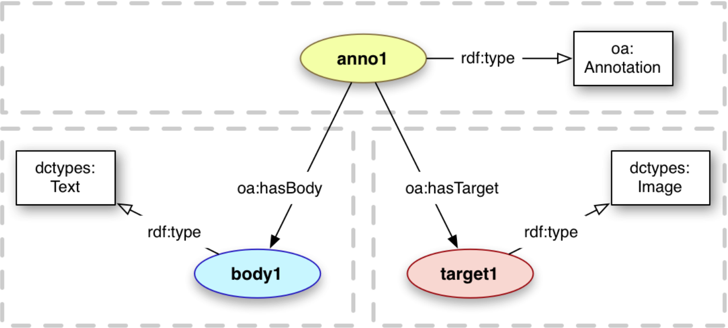

Figure 3.1: Open Annotation Data Model. (figure from [56])

Figure 3.1 shows the three main classes of resources that compose the OAC data model. These classes are Annotation, which describes the entire annotation that includes the two other classes Body and Target and creates the association between itself and the other classes. The next resource class Body, contains the actual content of an annotation, in the case of a blog comment it would be the comment text, in the case of a forum post the text of the post and so on. The last class is called Target and denotes the resource that the annotation is about such as a blog or a forum URL or even an image or a video file. The RDF properties oa:hasBody and oa:hasTarget associate an instance of an annotation class with it's respective body and target. An important distinction of the OA model is that an annotation can have multiple targets.

3.3.2 NIF: The NLP Interchange Format

The NLP Interchange Format [33], also called NIF, is a RDF-based format developed with the purpose of integrating the input and output of multiple NLP tools in order to enable "distributed and loosely coupled" NLP applications. Typical NLP tasks require the combination of multiple tools in order to achieve a particular goal, this task can however prove difficult to the incompatibility between the different APIs. Furthermore NIF tries to ease the way background knowledge can be integrated into various NLP tasks such as named entity recognition by providing an uniform layer by which the background knowledge data sources can be accessed.

The NIF standard describes two URI assignment strategies, an offset based scheme and a hash based URI strategy. The URI schemes are important since they allow linked data principles to be used when annotation different pieces of text for NLP processing. NIF also introduces a String Ontology in order to assign properties to the identified string elements and extends it with the Structures Sentence Ontology in order to enable the representation of sentences, phrases and words.

3.3.3 The Stanbol Enhancement Structure

Apache Stanbol[1] is a software framework that enables content management systems to benefit from semantic technologies by offering a series of web services that traditional CMS systems can make use of. Its main features are a set of modules that are contained in 4 main categories: Content Enhancement, Reasoning, Knowledge Models and Persistence. The Content

Enhancement module is of special interest to us since it offers an own custom NLP ontology, called the Stanbol Enhancement Structure [2] This ontology was developed in order to semantically describe the interaction between the different components of the Apache Stanbol NER pipeline. Apache Stanbol calls the NLP pipeline an EnhancementChain and its modules Enhancement Engines.

3.3.4 The NERD Ontology

The Nerd Ontology[3] was developed as part of the NERD evaluation framework[54]. This ontology is a mapping ontology consisting of manual mappings between a series of NLP related ontologies and taxonomies. The main focus in the NERD ontology is to integrate online NER services such as DBpedia Spotlight, AlchemiAPI, OpenCalais, Zemanta and other similar services into one common meta taxonomy that can be then used to evaluate this Web Services with regards to precision and recall.

The data integration in the NERD ontology is done by finding the least common denominator amongst the mapped ontology, or by defining a superclass for them. This mapping effort results in a series of 85 ontology classes which are categorized in the Core Ontology and Inferred Classes. The core of the ontology is constructed of high level concept classes such as: Person,Location,

[1] https://stanbol.apache.org/index.html

[2] https://stanbol.apache.org/docs/trunk/components/enhancer/enhancementstructure

[3] http://nerd.eurecom.fr/ontology

Organization, Time, Event, Animal, Thing, Function, Product while the Inferred classes contain more specific subclasses concepts such as: SoccerPlayer, Politician, Restaurant, Aircraft, Hospital etc.

This ontology, although not very complex, can prove useful in the evaluation of NER tools.

3.3.5 Comparison of NLP Ontologies

Ontology	OAC	NIF	Stanbol ES	Nerd Ont.
Serialisation Overhead	high	low	low	n/a
Integrates with	n/a	OAC, Uima, Gate, Stanbol etc.	n/a	NIF
Use Case	text annotation	NLP annotation	NLP annotation	NLP comparison
Standard	W3C draft	no	no	no
Tool Support	low	high	Apache Stanbol	n/a

Table 3.1: Comparison of Ontologies for Natural Language Processing

Table 3.1 shows a comparison of the ontologies we have presented before based on a series of criteria that are important for NLP tasks. The first criteria, serialisation overhead refers to the number of triples produced by a document annotated with the specific ontology. Ontologies such as OAC are data integration ontologies and not specifically tailored for NLP tasks, therefore a document encoded in such a model will be very verbose and have a high serialisation overhead. NIF or Stanbol Es are ontologies designed to be very simple and used for specific NLP tasks and therefore have a very low overhead. Another important criteria here is data integration. Although OAC has been designed with data integration and reuse in mind, it doesn't offer ready to use mappings to NLP ontologies, NIF on the other hand can integrate a lot of existing ontologies out of the box. The use-case aspect is also important since it shows what to expect from a specific ontology as well as the standardisation aspect since it shows us how large of a community support an ontology has. The tools support aspect is important for us since we want to use these ontologies in the implementation of our system.

3.4 Semantic Document Enrichment

In this section we present a series of approaches for document enrichment such as context enrichment, document linking and entity linking. We then proceed to identify some of the most widely used knowledge bases that can be used in combination with document enrichment and compare them based on a series of criteria that are relevant to semantic web applications.

3.4.1 Document Enrichment Methods

Document enrichment implies the annotation of named entities in existing documents and the interlinking of the annotated entities with internal or external knowledge bases. This allows for a better classification of documents, enables semantic search engines to work effectively and makes it possible for users to better understand the information in those documents by blending in added knowledge.

The authors of [44] describe three major ways of enriching documents:

Context Enrichment

Context enrichment is a method used in document enrichment that captures information about the context in which a document was created. Examples of context information ca be the people involved, business processes and scopes of the document. This method analyzed the activity logs in document management systems extensively and makes use of the methods we discuss in Chapter 4.

Document linking

Document linking adds another step to the Context Enrichment method by analyzing the relationships between documents when they are created. If for example, a document is created in the same business process as another document, those documents will be related to each other by a predefined relationship, and that relationship will be added to the document context. Such relationships can be: owner, creationProcess, creationTime etc. Other types of relationships can be determined statistically based on document similarity measures [38].

Entity linking

Entity linking is an extension of the Document Linking approach. It involves linking documents not only to related documents but also related entities in external knowledge bases. Most research papers usually only describe this method since the other methods require extensive knowledge about the creation context of the documents as well as the existence of significant document corpus.

Other methods

More recent research focuses on using document enrichment for the establishment of connections between different media types [6] [5] . These approaches can be considered as stand alone methods since they combine the previous approaches with speech recognition and computer vision methods.

3.4.2 Data Sources for Document Enrichment

Documents are enriched by adding links to information that is not directly expressed in the document itself. As a consequence, the quality of the enrichment process is directly established by the quality of the knowledge base the documents are enriched with. In order to make this process to produce good results we need to select existing semantic knowledge bases of consistent data quality and that provide an expressive ontology. Finding such semantic knowledge bases can prove difficult since it requires a good understanding of the contents of the knowledge base as well as the ontology and the knowledge base population process. In the following section we present on overview of the most widely used knowledge bases for document enrichment:

DBpedia

DBpedia[39] is a community effort that aims to extract structured data from Wikipedia. It consists of 2 main components: the extraction framework and the mappings Wiki.

The DBpedia extraction framework is a Scala based software framework that processes the Wikipedia dump files and extracts structured data from the Wikipedia articles. It focuses mainly on the properties from the Wikipedia info boxes but also has custom built extractors for labels, geo coordinates, category information, redirects, abstracts, images, page links and other kinds of information.

The Mappings Wiki is a crowd sourced approach that maps the Wikipedia info box properties to ontology properties and also allows users to create a data-driven ontology that maps the entire world view contained in Wikipedia. This approach has proved to be very successful, and has made DBpedia into the most used Linked Data source. Another valuable aspect of the Mappings Wiki is that it allows not only for the mapping of the English Wikipedia, but also for the mapping and creation of localized sub-ontologies, resulting in the creation of the DBpedia Internationalization Effort and the DBpedia Country Chapters.

DBpedia can be accessed over SPARQL through the DBpedia SPARQL Endpoint or downloaded in various RDF serializations for local processing.

Freebase

Freebase is a similar approach to DBpedia, the main difference being that Freebase, being owned by Google, has the financial backing to employ thousands of people in order to curate the data coming from Wikipedia and remove most inconsistencies and errors in it. Freebase also adds data from a series of otheropen data sources but, as a result of its data-driven ontology development approach, models its ontology closely to the Wikipedia category model.

Freebase offers no open SPARQL endpoint, but provides its own MQL endpoint. This knowledge base can also be downloaded in an RDF dump or in its own proprietary format.

Wikidata

Inspired by the efforts of the DBpedia and Semantic Mediawiki communities, Wikimedia has decided to embed structured data in the core of Wikipedia. This approach requires a total rethinking in the way Wikipedia Infoboxes and even Articles are created and maintained, thereby heavily impacting the Wikipedia community. In order to test this approach first before integrating it into Wikipedia, Wikidata[71] was created. Wikidata allows users to create info boxes in a structured way, with predefined properties that are checked for consistency on introduction against the info box definition.

Wikidata can be accessed over its own query interface, but does currently not enable SPARQL Queries. Dumps can be downloaded in a proprietary format but are also recently available as RDF. Furthermore, the DBpedia project is currently extracting the information from Wikidata, mapping it to the DBpedia ontology, and integrating it into DBpedia.

Comparison of Data Sources for Document Enrichment

	DBpedia	Freebase	Wikidata
Wikipedia Coverage	all	all	partial
SPARQL	yes	no	no
RDF dumps	yes	yes	yes
External Datasources	yes	yes	yes
Separate Ontology	yes	no	no
Financing	Open Source	Google	Wikimedia
Multilingual	yes / high	yes / low	yes / low
LOD Support	yes	partial	no

Table 3.2: Comparison of Data Sources for Document Enrichment

Table 3.2 shows a comparison of the most largest and most used knowledge bases that can be used for document enrichment. The criteria we chose for this comparison are those that we considered useful for choosing the best suited knowledge base for a specific task. Such criteria include the availability of a Sparql endpoint, existence of RDF dumps, integration external data sources, ontology support, multilingual and linked data capabilities.

3.5 Overview of Named Entity Recognition

Current information systems have to deal with large amounts of unstructured data: text documents, audio recordings or video files. Due to the nature of this data, computers cannot directly access and process it they way they would with relational databases. Filtering and organizing unstructured text documents for example, involves high amounts of manual labour. In order to make unstructured documents easier to process for computer systems, Information Extraction (IE) aims to transform unstructured data into semi-structured or structured data via automatic means.

One of the first steps in IE approaches is to recognize and extract all proper names and to classify them into a series of predefined categories of interest such as names of people, places and organisations. The research field that has developed around the task of entity extraction is called Named Entity Recognition (NER).

After named entities have been detected and extracted from the text, another important part in IE systems is to extract the relationships between these entities, this task is known under the term Relation Extraction (RE). Typical relations are person to person relationships such as John “is married with”Jane, person to organisation relationships: John “is the CEO of” Examplecorp, and location relationships , which relate the location of a person or an organisation:

John "resides in" the USA or IBM "is headquartered in" the USA.

While relation extraction deals with extracting relationships between two entities, the task of of extracting and classifying relationships between an arbitrary number of entities is applied in Event Detection. Examples of events can be: company mergers, monetary transactions, terrorist attacks etc.

In the last years a series of approaches have been developed that can recognize simple named entities in text documents, and disambiguate this entities from similar named entities that may appear in the same or different documents.

The most common NER approaches can be classified in three major categories based on the learning methods they use [45], the oldest method being hand-crafted rules and newer methods are based on machine learning approaches such as supervised, semi-supervised and unsupervised learning.

3.5.1 Hand-crafted Rules

The first NER systems used handcrafted rules in order to recognize named entities in text. These types of rules can be very simple, but are highly dependent on the characteristics of each language. In English texts for example, recognizing proper names can achieve high accuracy just by using a rule that looks for capitalized words, this type of rule however would not work for languages such as Korean or Japanese, where other types of rules need to be used. For specific applications domains such as chemistry, or biology the rules will become very complex and specific and adapting the rule-base to other domains than the one it was designed for is very difficult or not possible at all.

3.5.2 Supervised Learning Approaches

Supervised learning approaches usually work by using a large corpus of annotated data where the entity mentions have been labeled by hand. This corpus then gets divided into a test corpus and a training corpus. The system is then trained to recognize named entities similar to those in the training corpus and the recognition is evaluated against the training corpus. Techniques used for supervised learning based NER include Hidden Markov Models, Decision Trees, Maximum Entropy Models, Support Vector Machines and Conditional Random Fields.

3.5.3 Semi-supervised Learning Approaches

Building a training corpus for supervised learning approaches can be costly and time consuming, and it represents one of the biggest drawbacks of supervised learning. In order to address this drawback, so called semi-supervised or weekly supervised approaches use a method called bootstrapping. This approach allows them to start a set of seed entities (a small number of annotated named entities), it then detects the local context and the surrounding identifying features in which these entities appear and tries to find similar entities based on these contextual clues.

One of the most influential approaches in semi-supervised learning is called mutual bootstrapping [53]. It extends the bootstrapping approach by adding the types of entities to the initial seeds. It then gathers the patterns found around these entities and also ranks the contexts in which they were found.

3.5.4 Unsupervised Learning Approaches

Unsupervised learning works by trying to identify preexisting patterns in a text corpus by using clustering approaches. The main characteristic of unsupervised NER approaches is trying to overcome the need for an annotated corpus by using resources such as external knowledge bases or dictionaries (DBpedia, Freebase, Wordnet). The main difference consists in the absence of a training step by using similarity methods to compute the likelihood of a given named entity belonging to a specific entity type. The literature is very unclear on this topic, a lot of the methods described as unsupervised learning are actually semi-supervised methods, however one can classify them by the usage of topical methods for unsupervised learning such as: Clustering or Neural Networks.

3.5.5 Comparison of Learning Methods for Named Entity Recognition

Approach	Advantages	Disadvantages
Handcrafted Rules	ease of use good performance in specific cases	hard to adapt to other domains require experts to create rules can become very large and complex
Supervised Learning	does not require domain experts directly good scalability and performance	requires large amounts of labeled training data adapting to new domains requires new trainings data
Semi-supervised Learning	requires less trainings data lower cost in adapting to new domains	very sensitive to inconsistencies in labeled data out-of-domain data has only limited uses as trainings data
Unsupervised Learning	requires even less seed data than semi-supervised learning, or none at all	limited application domains the disadvantages of semi-supervised learning are more pronounced with unsupervised learning approaches

Table 3.3: Comparison of Learning Methods for Named Entity Recognition

Table 3.3 shows a comparison of the most common approaches to Named Entity Recognition based on the learning methods with the added inclusion of rule-based methods, since they are important for the understanding of the other approaches. We did not choose any specific criteria for this comparison but rather presented the most common advantages and disadvantages presented in the scientific literature. The scope of this table is to present a fast overview of advantages and disadvantages when choosing an appropriate learning method for NER, based on the specific task at hand. For example if someone wants to recognise names of specific organisations and already has a list of them, a rule-based approach would be the most cost-effective method to use. On the other hand for an advanced Named Entity Recognition systems one would need to use semi-supervised methods and variations thereof.

3.6 Conclusion

In this chapter we described our research challenge which is about the extraction of complex entities from text documents (i.e., unstructured data). We started with describing the necessary technologies for our approach such as semantic data models for natural language processing, document enrichment approaches and external semantic knowledge bases. We presented the current state of the art learning approaches for named entity recognition. Building upon the previously described technologies we aim at developing new conceptual approaches, that can go beyond the detection of proper names and detect what we call "complex entities" in text documents.

Chapter 4

Knowledge-Based Mining of Complex Event Patterns

4.1 Introduction

Detection, prediction and mastery of complex situations are crucial to the competitiveness of networked businesses, the efficiency of Internet of Services and dynamic distributed infrastructures in manifold domains such as finance/banking, logistics, automotive, telecommunication, e-health and life sciences. Complex Event Processing (CEP) is an emerging technology to achieve actionable, situational knowledge from huge event streams in real-time or almost close to real-time.

In many business organizations some of the important complex events cannot be used in process management because they are not detected from the workflows and decision makers cannot be informed about them. Detection of events is one of the critical factors for event-driven systems and business process management.

The current successes in business process management (BPM) and enterprise application integration (EAI) makes it possible that many organizations know a lot about their own activities. Almost all of the business activities are logged in different log and audit systems so that all they can be used to monitor the business processes. However, the huge amounts of event information cannot be used completely in the decision making and process controlling, because the specification of event detection patterns have to done manually by humans and are highly complex.

The permanent stream of low level events in business organizations needs an intelligent real-time event processor. The detection of occurrences of complex events in the organization can be used to optimize the management of business processes. The existing event processing approaches are dealing primarily with the syntactical processing of low-level signals, constructive event database views, streams, and primitive actions. Our research on semantic complex event processing [66, 65, 68, 67] provided solutions for the fusion of background knowledge with the event streams. We provided solutions (within our research project "Corporate Semantic Web" [1]) for the detection of complex events based on the

[1] http://www.corporate-semantic-web.de/semantic-complex-event-processing.html

background knowledge

In this research, we address the problem of automated extraction of patterns for detection of complex events. The existing approaches for the pattern detection are primarily dealing with syntactical processing of event sequences to detect complex patterns only based on the sequences of event happening. As an extension of the existing approaches for pattern mining, we investigate the usage of ontological background knowledge to be able to extract complex event patterns based on the relations of patterns to the resources in the background knowledge.

In the following, we setup the a model for the event processing and define the concepts of events and event streams (Section 4.2). We specify our research question based on the given event model and define the main research challenge (Section 4.3). We review the existing relevant approaches for pattern detection from sequence of data items and list the state of the art approaches for the pattern mining for complex event processing (4.4).

4.2 Event Processing Model

In this section, we specify our model for the problem that we address in this research. We introduce our model for events and event streams.

Definition 4.1 ***(Event)*** *An Event object is a tuple of $\langle \bar{a}, \bar{t} \rangle$ where $\bar{a}$ is a multiset of fields $\bar{a} = (a_1, ..., a_n)$, and is defined by the schema $\mathbb{S}$. The $\bar{t} = (t_s, ..., t_e)$ s a sequence of timestamps representing the different happening times of the event, the start t_s and end timestamps t_e of the event.*

For example an event in stock market applications has the fields *(name, price, volume, timestamps)*, like *(IBM, 80, 2400, 10:15, 10:15)* the start and end time of this event is, because this is an instantaneous event. An *Event* can also be considered as a set of attribute values $\langle \bar{av}, t_s, t_e \rangle$ where $\bar{av}$ is a multiset of attribute value tuples $\bar{av} = ((a_1, v_1), \ldots, (a_n, v_n))$, for the above example we will have $(((name, IBM), (price, 80), (volume, 2400)), 10:15, 10:15)$.

Definition 4.2 ***(Event Stream)*** *An Event Stream is an infinite sequence of events with the same schema $\mathbb{S}$.*

Definition 4.3 ***(Event Type)*** *An Event Type is a event stream with a data schema and can uniquely be identified. (A general definition, we refer to it also as "syntactic event type")*

An event instance is a single event. Composite events can be detected based on the temporal relationship of events, e.g., two events happens after each other in stream, or they happen at the same time. An event processing engine can detect events based on their temporal relationships or based on the syntactic matching of their attributes. The detection is defined based on the formal semantics which are defined event operation algebra, like operations defined in Snoop [8] (described in Section 4.4). Different event processing systems have already extended these operations and specified different event detection operations.

For modeling of background knowledge about events, we adopted a knowledge model from description logic [3].

Definition 4.4 ***(Knowledge Base)*** *A knowledge base (KB) is a pair $(\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox. An interpretation $\mathcal{I}$ is a model of a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{I}$ is a model of $\mathcal{A}$.*

A knowledge base KB can also be considered as a set of logical axioms K_S representing a logical theory in which a set of entailed logical axioms K_E are true so that $K_S \subset K_E$.

A new set of knowledge are driven from the existing logical axioms K_S. The entailment is specified based on the entailment rules, e.g., in description logic different expressiveness levels are defined, which can be mapped to different entailment rule sets.

After the specification of two models for the two worlds, event processing, and knowledge representation. We can assume that some of the fields of events (e.g., attribute/values) are in relation to some of the resources in KB (such as individuals, concepts, roles and sentences). It is possible to ask the knowledge based and retrieve background knowledge about the attributes of events.

Definition 4.5 ***Event2KB Link*** *– An Event2KB Link is a logical axiom which is embodied by one of the attribute/value pairs of an raw event instance and links it to an individual in the knowledge base.*

4.3 Research Problem

The problems that we address is in this research can now be substantiated in the following main problems.

- Is it possible to enrich the event stream history to extract complex event patterns based on the relations of events in the background knowledge?
- Is it possible to extract event detection patterns very close to the real-time and not as post-processing (which is usual in the most of pattern detection approaches)?
- How to optimize the separation of the event stream to data sets (stream slices) so that the pattern extraction can be optimized?

4.4 Related Work

In this section, we review the related work to our research problem. We start by introducing basic definitions of complex event processing concentrating on the element that will be probably employed directly in our solution. We review some of the approaches for pattern recognition. Machine learning provides very essential concepts about causality and anomalies. Many algorithms were proposed for mining data streams that can be adapted in other fields, like in CEP. We also review some algorithms that was designed for mining sequential databases. Such databases consist of data that can be ordered according to a timestamp, which makes it similar to events. Finally, we take a look at a recent paper that draws a bigger picture by proposing a comprehensive system for mining event patterns.

4.4.1 Complex Event Processing

Complex Event Processing, or CEP, is the "Computing that performs operations on complex events, including reading, creating, transforming, or abstracting them" [40]. A CEP-System allows defining complex events[2], receives simple event notifications and reports any occurrences of the complex ones in a real time fashion [49, 48].

Complex events are defined using event operators that bind simple and complex events together. Given an event source that fires the simple event types $\{A, B, C \ldots\}$, examples of complex events are $\{A \wedge B\}$, $\{(A \vee \neg B) \wedge C\}$, $\{A; B\}$ as we will see in detail later in this section.

4.4.2 Event Specification Language

There have been many attempts to formally describe a language for expressing complex events[3] and their relationships in a way similar to Boolean operators, especially in the context of active databases, like in [34] and [23].

Snoop Event Algebra Operators

Chakravarthy et al. provide an operation semantic for Snoop [7]. This operational semantic is build based in the event specification operators defined in **ODE** [22]. Snoop[4] provides an event specification language along with the semantics of composite events over a global event-history In Snoop an event E (primitive or composite) is a function from the time domain onto the boolean values, True and False.

$$E : T \rightarrow \{True, False\}$$

If an event of type E happens at time point t, then the function is True, otherwise is false.

The precise semantics of composite event detection are specified by the Snoop event operators as follows:

1. **OR ($\bigtriangledown$) Operator:** $(E_1 \bigtriangledown E_2)(t) = E_1(t) \vee E_2(t)$

   The OR ($\bigtriangledown$) operation matches the events when at least one of E_1 or E_2 can be matched.

2. **AND ($\bigtriangleup$) Operator:** $(E_1 \bigtriangleup E_2)(t) = (\exists t^1)(E_1(t^1) \wedge E_2(t)) \vee (E_2(t^1) \wedge E_1(t)) \;\; t^1 \leq t)$

   The Snoop $\bigtriangleup$ (similar to AND) operation matches the event when an instance of E_1 occurs and an instance of E_2 was already occurred in earlier or the same time point, or vice verca (E_1 occurred before E_2).

[2] http://www.slideshare.net/isvana/ruleml2011-cep-standards-reference-model http://www.slideshare.net/isvana/epts-debs2011-event-processing-reference-architecture-and-patterns-tutorial-v1-2 http://www.slideshare.net/isvana/epts-debs2012-event-processing-reference-architecture-design-patterns-v204b

[3] http://www.slideshare.net/opher.etzion/debs2009-event-processing-languages-tutorial

[4] We describe Snoop in more details because Snoop is one of the high impact research efforts in event processing field.

3. **ANY Operator:** The ANY operator matches, when exactly m match of events happens out of n events in time, ignoring the relative order of their occurrence.

$$ANY(m, E_1, E_2, \dots, E_n)(t) = \exists t^1 \exists t^2 \dots \exists t^{m-1}$$
$$(E_i(t^1) \wedge E_j(t^2) \wedge \dots \wedge E_k(t^{m-1})) \wedge (t^1 \leq t^2 \dots \leq t^{m-1} \leq t) \ \wedge$$
$$(1 \leq i \dots k \leq p) \wedge (i \neq j \neq \dots \neq k \neq p) \wedge m \leq n$$

4. **SEQUENCE Operator:** $(E_1; E_2)(t) = ((\exists t^1)(E_1(t^1) \wedge E_2(t)) \wedge t^1 \leq t)$ The sequence operator matches when E_2 occurs and the E_1 has already occurred in a time before E_2.

5. **Aperiodic Operators** (A, A^*)**:** The aperiodic operation of Snoop allows the expression of an aperiodic event in a time interval marked by two events. Snoop provides two different variation of aperiodic operator, the non-cumulative and cumulative operation.

   The A operator (Aperiodic non-cumulative) is matched each time E_2 occurs between E_1 and E_3. $\sim$ symbol means on all occurrence of E_3 (aka every occurrence of E_3).

$$A(E_1, E_2, E_3)(t) = (E_1(t^1) \wedge \sim E_3(t^2) \wedge E_2(t))$$
$$\wedge \ (t^1 < t^2 \leq t) \quad \vee \quad (t^1 \leq t^2 < t)$$

   The A^* operator is the aperiodic cumulative operator. A^* signals only once inside the given interval of two marker events (E_1 *and* E_3).

$$A^*(E_1, E_2, E_3)(t) = (E_1(t^1) \wedge E_3(t)) \quad \wedge \quad (t^1 < t)$$

   The operation accumulate the zero or more occurrences of E_2 between the E_1 and E_2. The operation is done and closed with the occurrence of E_3 and not with the happening of E_2.

6. **Periodic Operators (**P**,** P^***):** The period operator $P(E_1, [T], E_3)(t)$, E_1 and E_3 are two events, T is a constant amount of time. The operation detects all of happening of E_1 to E_3 in the constant time T. Formally defined:

$$P(E_1, [T], E_3)(t) \ = \ (E_1(t^1) \wedge \sim E_3(t^2)) \ \ \wedge$$
$$(t^1 < t^2 \leq t) \ \ \wedge$$
$$t^1 + i * T = t \ \ for \ \ 0 < i < t$$

   The cumulative variation of periodic operator accumulates time of occurrences of the periodic event, formally:

$$P^*(E_1, [T], E_3)(t) \ = \ (E_1(t^1) \wedge \sim E_3(t)) \ \ \wedge$$
$$\wedge \ t^1 + T \leq t$$

7. **Not ($\neg$) Operator:** The not operator detects the non-occurrence of an event. Not operation $\neg(E_2)[E_1, E_3](t)$ denotes the non-occurrence of event E_2 in the closed interval formed by E_1 and E_3. Formally defined in Snoop:

$$\neg(E_2)[E_1, E_3](t) = (E_1(t^1) \wedge \sim E_2(t^2) \wedge E_3(t))$$
$$\wedge t^2 \leq t^2 \leq t$$

Snoop also introduce the concept of parameter contexts which influence the detection behavior of snoop operators. For the detection of a complex event multiple matches might be available. Based on the semantic context of operators different matches of primitive events are available, e.g., for the event history (a b b) during the matching of (A;B) pattern, the complex event might be matched once or twice based on the semantics of event detection system.

Snoop defines different contexts for their operators, *unrestricted, recent, chronicle, continuous, and cumulative* which are specified in [8]. These context can change the behavior of event processing. The unrestricted context is the normal case of event detection operations and might produce a lot of complex events occurrences which all might not be useful for the applications.

The details of event processing context are specified in [8]. We shortly review these context, because they are first introduced in Snoop and have effects on event detection behaviors. In the *recent context* only the most recent occurrence of the initiator of the composite event is considered and will be used fro event detection, and all other non-initiator event instances will be deleted. The recent context is useful for applications in where high throughput of raw events should be used and multiple occurrences of the same event type do not effect the event detection pattern. In *chronicle context* an occurrence of an initiator is paired with a terminator event instance and they build unique couples, the oldest initiator with the oldest terminator. This is useful in applications where different occurrences of types of events and their correspondence needs to be matched. For example detection of events between aborts, rollbacks, and other transaction operations in a database. The *continuous context* each initiator of composite event starts the detection of the event, the incoming terminators cause the detection of one or more composite event of the same type. This kind of context is interesting for trend analysis and forecasting application in which a moving window specifies the data for event detection. In the *cumulative context* all occurrences of the incoming event from the same type are accumulated until the composite event is detected. This context has useful applications in which multiple occurrences of the same event type needs to be grouped and used in a meaningful way when the event occurs.

4.4.3 Windowing and Slicing of an Event Source

It is often necessary to consider only a part of the event source, especially when dealing with an endless flow of events like in event streams. We call such a "bound portion of an event stream" [40] a *window.* A window can be the last fifty events in a stream for example or the events occurring within a time frame of five minutes.

The source is divided into multiple short windows assumed to contain enough information about its patterns. Finding the size of the window is actually no trivial task [42]. Depending on the problem domain, this size can be set manually or determined automatically based on the input.

Apart from the concrete size of the window, we consider the events that happen within the same window to be semantically related more strongly than other events, which makes the window concept in CEP similar to the concept of transaction in sequential databases and association rules.

If we use a windows with SNOOP for example, events that occur in the same window are connected by the sequence operator. If an event didn't happen during a window, we assume that its negation happened.

Another important method is to take *slices* from the stream. A slice is a longer representative of the whole stream. The concept is especially helpful in contexts where pattern data has to be held in memory, but saving the whole stream, or tracking event patterns of the whole stream, is impractical.

Types of Windows

The *sliding window* is the most applied method for scanning event streams. It guarantees that all associations between occurring events would be considered based on distances and not on positions.

Yet other forms exist. As we will see later in this section, some early algorithms [41] [73] just split the stream into *buckets* associating events based on the position at which they occur and not only on the relative time distance among their occurrences.

In [11] Calders et al. proposed the concept of *flexible window*, which also considers the history of an item when computing its frequency in the current window.

4.4.4 An Example on Complex Event Patterns

To get a grip on event patterns, we present here an example of an abstract event source and show the patterns that can be extracted from it.

Assuming the set of event types $E = \{A, B, C, D\}$, and considering the following sample source of instances:

$$a, b, c, d, a, b, a, c, a, b, c, d$$

Each occurrence of an event type is a pattern. From the above sample stream we can extract the following *primitive patterns*:

$$P_p = \{A, B, C, D\}$$

Those are simply the types of all instances that occurred in the stream. To look closer, we can divide our source into windows of a given size w. Assuming a window size $w = 3$, we get the initial window $w_0 = \{a, b, c\}$:

$$\overbrace{a, b, c}^{w_0}, d, a, b, a, c, a, b, c, d$$

In addition to primitive patterns, we can easily find some *complex* event patterns, i.e., patterns consisting of multiple primitive events related to each other by event operators. We observe for example that b occurs after a, c after a etc. So the above example contains the following *explicit patterns*:

$$EP_0 = \{\{A\}, \{B\}, \{C\}, \{A; B\}, \{A; C\}, \{B; C\}, \{A; B; C\}\}$$

Since no instances of D occur in this window, we can induce the set of *negation patterns*:

$$NP_0 = \{\{\neg D\}\}$$

Moreover, these patterns *occur* in the same window as the first set EP_0, so we can generate further implicit patterns using the AND operator $\wedge$ by matching each negation in NP_0 with each pattern in EP_0 to get the set of *implicit patterns* IP_0:

$$\begin{aligned} IP_0 = \{&\{\neg D \wedge A\}, \\ &\{\neg D \wedge B\}, \\ &\{\neg D \wedge C\}, \\ &\{\neg D \wedge \{A; B\}\}, \\ &\{\neg D \wedge \{A; C\}\}, \\ &\{\neg D \wedge \{B; C\}\}, \\ &\{\neg D \wedge \{A; B; C\}\} \\ &\} \end{aligned} \tag{4.1}$$

If we continue scanning the stream through a sliding window, we can extract further interesting patterns. We can slide the window one step to get $w_1 = \{b, c, d\}$:

$$a, \overbrace{b, c, d}^{w_1}, a, b, a, c, a, b, c, d$$

As we did with w_0, we can extract the following explicit patterns from w_1:

$$EP_1 = \{\{B\}, \{C\}, \{D\}, \{B; C\}, \{B; D\}, \{C; D\}, \{B; C; D\}\}$$

The set of negation patterns for w_1 is $NP_1 = \{\neg A\}$, which has to be matched with all the patterns contained in w_1 to generate the set of implicit patterns:

$$\begin{aligned} IP_1 = \{&\{\neg A \wedge B\}, \\ &\{\neg A \wedge C\}\} \\ &\{\neg A \wedge D\}\} \\ &\{\neg A \wedge \{B; C\}\} \\ &\{\neg A \wedge \{B; D\}\} \\ &\{\neg A \wedge \{C; D\}\} \\ &\{\neg A \wedge \{B; C; D\}\} \\ \} \end{aligned} \tag{4.2}$$

Moreover, if we consider the two sets of explicit patterns EP_0 and EP_1 extracted from two successive windows, we can induce further implicit patterns using disjointed events from both windows. If we take the patterns $\{B;C\}$ from EP_1 and $\{B;D\}$ from EP_1, we can induce a shared pattern between the two windows indicating the occurrence of B with one of the events C and D:

$$\{\{B;C\} \wedge \{B;D\}\}; \{\{B;(C \vee D)\}\}$$

4.4.5 Pattern Detection Algorithms

In the context of this research , we are interested in machine learning solutions for the problems of association and causality between occurring events. We review some basic algorithms that we consider relevant to our work.

We start with APriori algorithm that evaluates the strength of association among jointly occurring items, then we explain the notion of anomaly detection that aims at detecting outliers in a flow of frequent elements.

Association Rule Learning

An association rule is a relationship that binds a set of items. This relationship exists when these items occur together in some context. Early researches about association rules concentrated on the problem of *market basket*, but mining these rules in a database can be applied in other fields too, like in mining association rules between events.

Simply put, an association rule exists between two products, say beer and chips, if they appear together in shopping bills more often than other products do. So if beer and chips appear often together, one can say that a costumer who buy beer is more likely to buy chips too than other customers.

Apriori:Fast Algorithm for Mining Association Rules

A solution for this problem was introduced by Agrawal et al. in [1] where they present a formal model and an algorithm called APriori that finds all significant association rules in a database.

The model works on a set $\mathcal{I} = I_1 \dots I_n$ of n items and represents a transaction as a binary vector t, where $t[k] = 1$ if I_k appears in the transaction and $t[k] = 0$ otherwise.

A transaction t is said to *satisfy* a set of items $X = I_k$ where $k \in [1..n]$ if $t[k] = 1 \forall I_k \in X$.

Thus, an association rule is an implication $X \Rightarrow I_j$, where X is a subset of $\mathcal{I}$ and $I_j \in \mathcal{I}$.

The association rule $X \Rightarrow I_j$ is satisfied in the set of transactions T with confidence factor $0 \leq c \leq 1$ if, and only if, $c\%$ of the transactions that satisfy X also contains the element I_j, which we denote $X \Rightarrow I_j|c$.

APriori is one of the most known algorithms for mining association rules in a database [4]. The goal of the algorithm [2] is to generate *large* item sets. It starts by scanning the database and counting the occurrences of individual items, i.e., large 1-item sets. Only item sets with high frequencies are kept for the next pass. The algorithm continues by scanning the database again and computing frequencies for large k-item sets.

Enhanced versions has been proposed for several applications in various fields. Examples are GSP [32] and SPADE [74] algorithms developed for mining sequential data as we will see.

Anomaly Detection

An anomaly is an unexpected pattern being recognized at a time point at which another, more frequent, pattern is expected.

The problem of anomaly detection goes back to the works of the statistics community in the 19^{th} century. Many techniques has been developed since then [9].

In the context of complex event processing, an anomaly, or an outlier, is an event occurring with unusual attributes, or a series of events differing partially from a frequent pattern.

The event of withdrawing a big amount from a bank account from which only little amounts used to be drawn is an unusual event that might indicate that a bank card has been stolen.

On the other hand, if the notification sequence {On, Temperature(50), Temperature(90), Off} in an automatic kettle represents a normal behaviour, then the series { On, Temperature(50), Temperature(90), Temperature(95)}, i.e., the absence of the usually occurring Off-event, is an anomaly that indicates a failure in the automatic switch-off mechanism.

Furthermore, the series { Temperature(50), Temperature(90), off}, i.e., the kettle starts without pressing the on-button, might indicate a problem with the power switch causing the kettle to start without the switch being pressed.

4.4.6 Mining Algorithms for Sequential Databases

Sequential pattern mining is detecting frequent items in a set of ordered items in a sequential database, i.e., a database whose items can be ordered based on a timestamp [61].

Several techniques have been suggested for mining sequential data using machine learning algorithms. Available algorithms, mostly derived from Apriori [2] and PrefixScan [51], extract all frequent patterns from a sequential database or stream.

Recent algorithms adopt a more compact representation of the patterns by considering only closed patterns, i.e., patterns that are not a part of other patterns with the same frequency.

Algorithms in this field can be divided into two categories [32]:

- *Apriori-Based*: Optimized versions of Apriori algorithm. We will consider GSP and SPADE from this category.
- *Pattern-Growth-based*: Discovering frequent sequences without generating large item-sets. We will take a look at PrefixSpan and simillar algorithms form this category.

Generalized Sequential Pattern Mining Algorithm

Generalized Sequential Pattern Mining Algorithm, or GSP, is an enhanced version of Apriori [32] that starts by generating candidate patterns for k-frequency first and continues by pruning infrequent and repeated ones.

GSP still produces a huge set of candidate sequences and requires multiple scans of the database.

Sequential Pattern Discovery using Equivalent Class

Sequential PAttern Discovery using Equivalent class, or SPADE, uses efficient lattice search techniques and scans the database only three times [74].

SPADE suffers from the same deficiencies of the former algorithm like the huge set of candidates and the need to scan the database multiple times.

Prefix-Projected Sequential Pattern Mining Algorithm

Since frequent subsequence can always be found by growing a frequent prefix, Prefix-projected sequential pattern mining algorithm, or PrefixSpan, generates frequent sequences based only on frequent prefixes [51].

Closed Pattern Mining: CloSpan and ClaSP

In [72] Yan et al. proposed CloSpan, an algorithm for mining closed sequential patterns, i.e., patterns not included in other patterns that are more frequent. Closed patterns include all the information about frequent patterns in the database and can be used to minimize the search space and memory usage during the mining process.

ClaSP [24] was an enhanced version of CloSpan that applies additional pruning techniques and a vertical database layout.

4.4.7 Data Stream Mining

In mining data streams, the aim is to extract "knowledge structures represented in models and patterns in non stopping streams of information" [20].

In this section we review the most important algorithms that make the landmarks of this field.

As we will see, some algorithms take a probabilistic approach finding estimated frequencies of itemsets, while other algorithms try to find the exact frequencies deterministically. Furthermore, some algorithms concentrate on recent items and consider them more relevant than the old ones.

The reviewed algorithms differ in the way they handle the stream. Some of them divide it into successive non-overlapping buckets, while others use some kind of sliding window that takes into account any neighbourhood between incoming items regardless of the time point at which the event occur.

Clocking method also differ. While old algorithms considered the arrival of an item to be a trigger for updating the data structure, newer algorithms use time points to estimate the age of observed items.

Lossy Counting

In [41] Manku and Motwani present two algorithms for computing frequencies of arriving items in a data stream with a configurable error rate. Using a support threshold $s \in [0..1]$ and a tolerance factor $\epsilon \in [0..1]$, where $\epsilon \ll s$, and assuming the count of already seen items to be N, they assure that the output of their algorithms would maintain the following guarantees:

- No false negatives: All items with frequency exceeding sN are output.
- Restricted error rate for false positives: No items with a frequency under $(s - \epsilon)N$ is output.
- The difference between the estimated frequencies and the true ones is at most ϵN.

The algorithm divides the observed stream into *buckets*. The computed frequency of each incoming item is updated and infrequent items are pruned from the data structure when the end of a bucket is reached. For each potentially frequent item, the algorithm remembers the estimated frequency. If the newly inserted item has been previously pruned in some stage, the number of times the item could have occurred before can be computed as the number of elapsed buckets minus one. This number represents the maximum error rate in any estimated frequency.

Remembering the maximum error of current frequencies guarantees not pruning any frequent item since the maximum value of the real frequency can be computed, thus the guarantee not to get any false negatives.

On the other hand, some items might remain in the set of frequent items although they are not frequent if the summation of their estimated frequency and the possible misses in the past exceeds the sought threshold. Such items represent the false positive cases.

Frequenct Datastream Pattern Mining

In [73], Yu et al. argue that even the little count of false positives allowed in false-positive oriented algorithms, like Lossy Counting, can lead to a huge number of false positives in the final results.

Considering the same argument for the purpose of this thesis, we have to notice that computing frequencies of complex event patterns based on already erroneous data about primitive events will lead to much greater error rates in the final results which involve complex events.

An important feature of FDPM, or Frequent Datastream Pattern Mining, is the use of Chernoff bound to estimate the error rate of currently estimated frequencies instead of, as in Lossy Counting, depending on a fixed value.

The algorithm receives items and keeps track of infrequent ones. Using Chernoff bound, the number of observations required to achieve the target confidence is updated at runtime.

Assuming a support level θ and a probability control variable δ given by the user, the algorithm ensures an output with no item having frequency less than θ, thus no false positives. On the other hand, the probability for any frequent item to appear in the results is at least $1 - \delta$ [35].

Moment

Moment [13] is an algorithm for maintaining *closed* frequent itemsets over a stream using sliding window. An itemset is said to be closed if it has a higher frequency than of its super itemset.

In a sliding-window configuration, the algorithm depends on the heuristic that the set of frequent itemsets changes relatively slowly over successive windows [35], so the problem can be efficiently solved by concentrating on the boundaries between frequent and infrequent itemsets.

The algorithm uses a compact in-memory data structure called closed enumeration tree, or CET, for tracing closed frequent itemsets as well as candidate itemsets that form the boundary between frequent and infrequent itemsets.

Given a minimum support s and a database $\mathcal{D}$ containing itemsets, where an itemset consists of members of a set of items σ, the algorithm tries to find itemsets with frequency $s|\mathcal{D}|$.

In the tree of closed itemsets, the algorithm keeps track of the following types of nodes:

- *Infrequent gateway nodes*: Infrequent nodes whose parent or one of its siblings is frequent.
- *Unpromising gateway nodes*: Frequent itemsets contained in closed itemsets with the same frequency.
- *Intermediate nodes*: Itemsets containing sub-itemsets with the same frequency.
- *Closed nodes*: Closed itemsets in the current window.

The task is now to update this structure while items arrive. At any moment, the set of closed itemsets can be reported as the output of the algorithm.

4.4.8 Mining Recent Frequent Items

In their work [10], Chang and Lee emphasize the relevance of *recent* frequent items and propose an algorithm that adaptively detect such patterns over an online data stream.

The algorithm utilizes a *damped window* model [35] where itemsets have a weight that decay over time, which gives the most recent itemsets higher weight compared to the old ones. Damping weights are maintained using a decay factor d. The weight of an itemset is reduced by $1d$ at the arrival of each new itemset.

The data stream is processed transaction for transaction. The arrival time point of transactions is registered so that the weights don't need to be updated, they can be simply computed on demand. When a new itemset arrives, only the counts of its super-sets are updated. The count of those is reduced by the decay factor then increased by one.

Time Fading Model

Similarly to the damped window model, time fading model emphasizes the importance of recently incoming items by reducing the weight of old items. Algorithms of this model, like $\lambda - HCount$ in [12], introduce a fading factor $\lambda \in [0..1]$ and grants an item that has arrived n time points ago the weight λ^n which needs to be updated only when the item occurs .

4.4.9 Process Mining

Process mining starts by analyzing an event log of an existing information system. Logged events represent *activities*, i.e., a well-defined step in some process, performed in the system. An event belongs to a specific case and has its order among other events within this case [70].

There are three main types of process mining:

- *Discovery*: Where an underlying process model is extracted from an event log without any prior knowledge.
- *Conformance*: Where an event log is examined to check whether it conforms to a given model or not.
- *Enhancement*: In which the information extracted from an event log is used to enhance an existing process model.

We will concentrate on discovery in our review since it is the closest to our task. Discovering a model hidden in an event log is actually the same as detecting a frequent event pattern. But the context that binds items in process mining is the case, while CEP consider windows to be the container of events.

4.4.10 Fuzzy Mining

Gnther et al. [31] complain about process mining methods that result in incomprehensible "spaghetti-like" models and propose a fuzzy approach inspired by the readability and comprehensiveness of modern maps visualization systems.

Modern road-maps tend to *aggregate* low-level data and display summarized visual information. They also apply *abstraction* to hide insignificant information and *emphasize* relevant information using highlighting. Finally, specialized maps offer great opportunities for *customization*.

The authors suggest two metrics for evaluating detected models and applying features of road-maps on them. These two metrics are:

- *Significance*: Indicates the importance of an event or an event sequence. This metric can be measured by the frequency of the event for example.
- *Correlation*: Shows the strength of a relationship between two successive events. To measure correlations, the authors suggest checking overlapping attributes among such events or the similarity in their names. We notice here an obvious analogy to the measures of association rules.

Fuzzy mining can now be achieved to reach a simplified model by including:

- Events of high significance.
- Less significant but highly correlated events. Those events have to be aggregated in clusters.

Events with both low significance and low correlation are not considered in the resulted model. The result is a simplified, but comprehensive, model that summarize the main features of the process traced in the event log.

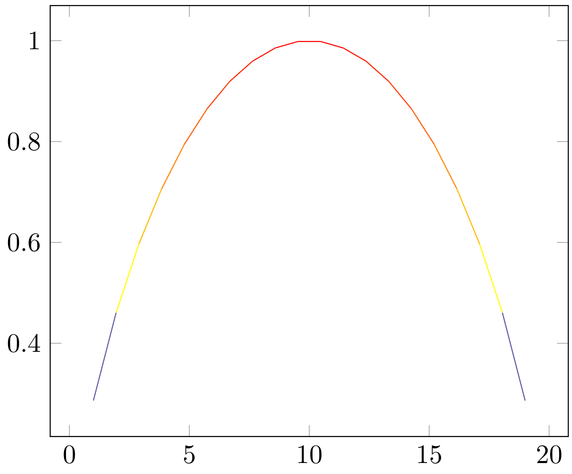

Figure 4.1: IGR for two classes as a function of the size of one class assuming a total size of 20 for the whole set.

4.4.11 Information Gain Ratio

Information Gain Ratio, or IGR, is a measure used with decision trees to determine the amount of information gained when a decision is made in some node of the tree [52].

In the case of classification, this ratio can be used to detect the attributes that mostly characterize a specific class. This is achieved by studying a learning set of objects whose classes are known and using their attributes as branching conditions in a decision tree [42].

Let's consider a set of images N with n images. Out of the n images, p images belong to a class P and r images belong to another class R. The probability for an object O to be classified as member of class P is given by $\frac{p}{n}$ and of class R by $\frac{r}{n}$.

According to Shannon [60, p.20], the amount of informational entropy contained in the question about the class of each of the images in the set is given by the following equation:

$$I_N = -\frac{p}{n}log_2(\frac{p}{n}) - \frac{r}{n}log_2(\frac{r}{n}) \tag{4.3}$$

The value of this function depends on the relative size of the classes. Figure 4.4.11 demonstrates the value of this function depending on the size of one class. The figure shows that the gain is at its most when the objects are equally distributed on the classes. This means that an attribute that can distinguish such classes is of most significance.

To generalize the concept, we can consider the case of splitting the set N into subsets $N_i : i \in [0..l]$ based on an attribute A that takes its values from $\{a_1, a_2..a_l\}$. I.e., we classify an object o in class N_i if $o.A = a_i$. Figure 4.4.11 visually demonstrates this branching.

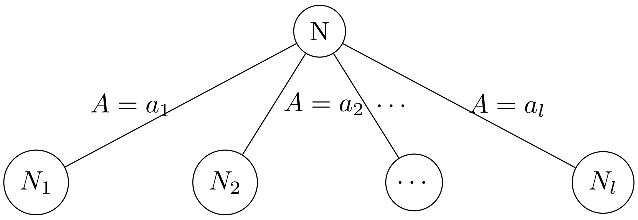

Figure 4.2: Branching in a decision tree based on a the value of a property A

The amount of entropy in this case is given by [52]:

$$E_A = \sum_{i=1}^{l} \frac{|N_i|}{|N|}.I_{N_i} \tag{4.4}$$

Considering that a class N_i contains p_i objects of class P and r_i of class R, we find that:

$$E_A = \sum_{i=1}^{l} \frac{|N_i|}{|N|} \cdot [-\frac{p_i}{n_i} log_2(\frac{p_i}{n_i}) - \frac{r_i}{n_i} log_2(\frac{r_i}{n_i})] \tag{4.5}$$

Where the probability for an object o to belong to a class N_i is the size of N_i, denoted here as $|N_i|$, divided by the size of N denoted $|N|$. And the amount of information we gain by classifying based on the attribute A is the reduction we achieve in the entropy, which can be measured by:

$$IGR(A) = I_N - E_A \tag{4.6}$$

If we have a set of 100 objects 60 of them are of class P and the rest of class R, i.e., $I_N = 0.97$, and the attribute $A \in a1, a2$. The information gain depends on the ability of the attribute to distinguish between the two classes.

If the values if this attribute leads to similar distribution as in the whole set, the information gain would be close to zero because $E_A = IN$.

On the contrary, if each value of the attribute charachterizes one of the classes, i.g. all objects O with $O.A = a_1$ belong to class R and those with $O.A = a_2$ belong to class P, then the attribute is very characteristic and leads to the highest information gain with $E_A = 0 \implies IGR_A = I_N$, i.e., the whole amount of information is gained and we can perfectly distinguish between the two classes based on this attribute.

4.4.12 State of the Art on Event Pattern Mining

In this section, we review a work that tries to solve a very similar problem. We consider iCEP framework presented in [42, 43] to be the state of the art in this field because it is, at the time of writing these lines, the most recent work that tries to extract event patterns from an event stream, which is exactly the task we aim at addressing.

In [42], Margara et al. presented a comprehensive solution called iCEP that finds meaningful event patterns by analyzing historical traces of events, extracting event types and attributes and applying machine learning ad-hoc algorithms on those patterns to detect hidden causalities between primitive and composite events.

iCEP depends heavily on Information Gain Ratio, to measure the influence of events on one another.

The authors propose an event model in which an event has a type and attributes. The event type defines the attributes characterizing instances of this type, and an event instance has values assigned to its attributes. Additionally, each event is marked with a time stamp referring to the time point at which the event has happened.

The following example suggested by the authors represents an event:

Temp@10(room=123, value=24.5)

This is an event of type Temp, for temperature, that occurred at the time point 10 and has two attributes: room, whose value is 123, and value, equalling 24.5. Naturally formulated: The temperature in room 123 was 24.5 at time point 10.

Composite events are built using five operators:

- *Selection*: allows selecting relevant event notifications based on the values of their attributes.
- *Conjunction*: retrieves patterns of multiple events happening together.
- *Sequence*: captures two events happening in a specific order.
- *Window*: defines the maximum time frame of a given pattern.
- *Negation*: expresses the absence of an event.

The authors suggested a syntax for event patterns shown in the following example:

$\underline{Pattern\, P_3}$
within 5m. Smoke() and Temp(value >50) and **not** Rain(mm >2)
where { Temp - >Smoke }

Which refers to capturing a smoke event and a temperature event with the "value" attribute exceeding 50 and no *Rain* event within a maximum time distance of 5 minutes between them. Additionally, the *Temp* event has to proceed the *Smoke* event, as to be inferred from the statement $Temp- > Smoke$.

Figure 4.3 shows the different modules composing iCEP.

Event Types Learner

By examining event traces of primitive events, i.g. $A, B..$, and composite ones, i.g. $CE_1, CE_2 \ldots$, the event types learner tries to recognize primitive events that led to the occurrence of the complex ones.

To achieve this goal, a set of variables $v_1, v_2 \ldots v_n$ is constructed including primitive events that occur in the same time window and functions of them.

For each variable v_i the information gain ratio IGR_v is computed. A variable is considered relevant if its IGR exceeds a specific threshold.

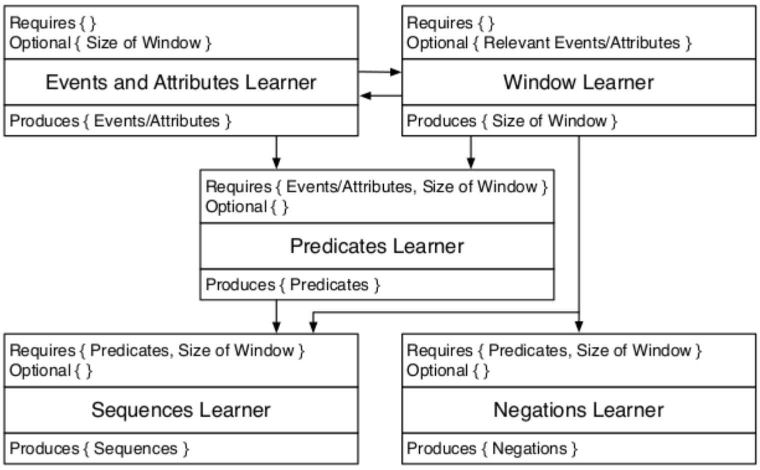
Requires { }
Optional { Size of Window }
Events and Attributes Learner
Produces { Events/Attributes }
Requires { }
Optional { Relevant Events/Attributes }
Window Learner
Produces { Size of Window }
Requires { Events/Attributes, Size of Window }
Optional { }
Predicates Learner
Produces { Predicates }
Requires { Predicates, Size of Window }
Optional { }
Sequences Learner
Produces { Sequences }
Requires { Predicates, Size of Window }
Optional { }
Negations Learner
Produces { Negations }

Figure 4.3: Overall Architecture of iCEP[42]

Window Learner

Similarly, and having relevant events and attributes, the window learner tries to find the window length that maximizes information gain from the relevant variables. The information gain of the window for a set S of variables is measured by:

$$IGR_A = \sum_{s \in S}^{I} GR_s \tag{4.7}$$

Predicates, Negations and Sequence Learners

The predicates learner Tries to find the values of a relevant attribute that lead to the occurrence of a composite event and builds a decision tree to model these relationships. The task of the *negation learner* is to detect relevant negations, i.e., events whose absence led to the occurrence of a composite event. Finally, the sequence learner learns the temporal relationships between primitive events in the context of a composite one.

4.5 Semantic Event Pattern Mining

In the previous section, we reviewed the relevant approaches for the extraction of patterns from the stream of data. These approaches are designed to extract complex patterns based on the temporal sequence and attributes of events. We aim to extend the existing approaches for the extraction of event detection patterns based on the relations of events in the background knowledge.

Our approach consist of a preprocessing step which enriches the event stream with the relevant background knowledge. In the preprocessing step we extract the relevant background knowledge for each of the event objects based on their attributes and enrich these background knowledge to stream. In the following pattern mining step complex event patterns should be detected based on the enriched knowledge.

We plan to investigate the usage of background knowledge for the pattern mining. For the patten mining approach we consider a part of the event data as stream slice (a relatively large data stream window). The event pattern mining system has to processed the enriched event stream data within the given stream slice and extract frequent and infrequent patterns.

In our approach the main target is to extract complex event patterns so that they can be used to control the business processes. This fact is the main difference between our research challenges and the research area of business process mining[5].

4.5.1 Data Sources for Event Pattern Extraction

Our approach requires to work on event stream data sources so that it can extract complex event patterns from the history or flowing data stream.

The most important event stream data sources in the business processes are different logging systems of the business process management systems. Almost all of the business activities like staff activities and customer relations are logged in the systems log files. The changes on the business documents by the staff members are also high promising event stream data sources because every updates of the document by the staff members are logged in the audit reports.

Corporate internal collaborative software systems like Wiki systems and corporate document management systems like Microsoft SharePoint[6] provide large amounts of log files and audit reports that can be analysed by our event pattern mining approach. The log files are provided in different formats, like Apache standard log format, MXML[7] (Mining eXtensible Markup Language) and XES[8] (eXtensible Event Stream).

As background knowledge for the analyse of event stream different background knowledge sources can be used, like internal databases (database schemes and data objects) and open resources of background knowledge like available data on Linked Open Data (LOD) sources.

4.6 Comparison of Algorithms

Table 4.1 shows the main properties of the algorithms we reviewed above. We can see that some algorithms try to detect all patterns, while others concentrate on closed or recent ones. The results of the algorithms can be probabilistic with a specific error rate, or deterministic containing the exact frequencies of the detected patterns. While most algorithms count frequencies for the whole stream, some algorithms limit their scope on the current window.

Early algorithms divided the stream into non-overlapping buckets, while more recent ones use more advanced windowing methods. The $\lambda HCount$ al-

[5] http://www.processmining.org/ , retrieved April 2014

[6] http://office.microsoft.com/en-001/sharepoint/ , retrieved April 2014

[7] http://www.processmining.org/logs/mxml

[8] http://www.xes-standard.org/ , retrieved April 2014

Algorithm	**Patterns**	**Probabilistic**	**Scope**	**Segmentation**
Lossy Counting [41]	All	Yes	Stream	Buckets
FPDM [73, 35]	All	Yes	Stream	Buckets
Moment [13]	Closed	No	Sliding	Sliding
estDec [10]	Recent	No	Stream	Buckets
λ-HCount [12]	Recent	Yes	Stream	Continuous

Table 4.1: Comparison of Algorithms for Data Stream Mining

gorithm doesn't divide the stream at all and regards it as a continuous flow of transactions.

To our best of knowledge, the only research work that target on pattern mining for complex event patterns is the iCEP framework [42, 43]. Our work extend this work for more abstract and higher-level event detection patterns.

4.7 Conclusion

In this chapter, we described our research challenge about the extraction of complex event patterns for event streams. We reviewed the most relevant existing approaches for the pattern mining from the data streams and described our initial plans for the investigation of event enrichment prior to the event patten mining. In the next report, we will report on the detail concepts for the knowledge-based event pattern mining.

Chapter 5

Conclusion

Success of content distributors and consumers is dependent on the delivery of relevant content tailored to the needs of the recipients. In this project, we envision a process chain for the creation of Corporate Smart Content, which we define as content enriched with the necessary corporate knowledge that enables applications for needs-based content delivery.

The process includes the in-house creation of content and/or importation of content from external sources, such as (linked) data repositories, audio/video archives, or news feeds as well as the creation of own or imported external ontologies, semi-automated, recommendation-based annotation of content and population of ontologies, and recommendation based enrichment with conceptual knowledge from the ontologies, as well as process knowledge mined from activity and event patterns.

Corporate Smart Content as the outcome of this process will enable the construction of smart applications, that allow for situation-aware and context-sensitive access to corporate content, that help employees or end-customers finding the content they need in order to get their work done, and that fits the current project they work in, the role they assume in the project, the current step in their process, and the information needs resulting from this situative context.

The sub-project of Freie Universität Berlin *Smart Content Enrichment* tackles three activities in this process:

Aspect-oriented ontologies: Enrichment of corporate content with ontological knowledge enables applications and users to make sense of the content and provide more relevant search results. Current ontologies lack means to provide contextual meta knowledge about the facts they model, such as situational relevance or intention. Aspects will provide this kind of meta knowledge and allow for context-aware access to aspects of ontological knowledge that are relevant in a particular situation. In the following project phases, we will provide a formal definition of such aspects in ontologies and extensions to existing ontology development methods and tools with means for modeling ontological aspects as well as APIs to access them.

Complex Entity Recognition: One of the key elements in corporate documents are named entities that give an idea about the main topics of those documents. Complex entities are classes or instances that have dependencies to other classes or instances. Identifying these dependence relations and deriving composite entities from them will, e.g., allow making concepts and topics in documents explicit that are only implicitly referred to, and will allow establishing connections between documents of corresponding topics which are not explicitly mentioned. Based on the state-of-the-art analysis in the first project phase, the next steps will be to adapt existing NER methods and develop new methods for recognizing complex entities, annotating them and linking them to them to entries in a background knowledge base. We will implement the previously mentioned methods in a prototype Complex Entity Recognition system.

Knowledge-Based Mining of Complex Event Patterns: In this report, we described the research challenge of the extraction of complex event patterns for event streams. We reviewed the most relevant existing approaches for pattern mining from data streams and described our initial plans for the investigation of event enrichment prior to event patten mining. In the next project phase, we will develop detailed concepts for knowledge-based event pattern mining.

We are going to identify which kind of event detection knowledge-based patterns can be extracted from the enriched event stream and how the event enrichment can be optimized for the pattern detection. Furthermore, we plan to illustrate the usage of the extracted pattern in business processes.

Appendix A

Work Packages

Work package FU 1	**Aspect-Oriented Ontology Development (AOOD)**	
Work package FU 1.1	**Analysis of relevant aspects for the access to corporate knowledge**	09/13-02/14
WP 1.1 Task 1.1.1	Requirements analysis of a formal approach to aspect-oriented access to ontologies	09/13-02/14
WP 1.1 Task 1.1.2	Development of a prototypical formalism for the specification of temporal aspects (validity periods) and integration with semantic search	09/13-02/14
WP 1.1 Task 1.1.3	Classification of identified aspects and definition of a strategy for the implementation of a generic formalism and a generic technical method for the access to corporate knowledge using the identified aspects	09/13-02/14
WP 1.1 Milestone 1.1.1	Validation of the strategy with the industrial partners	09/13-02/14
Work package FU 2	**Semantic Complex Entity Recognition and Annotation in corporate data**	
Work package FU 2.1	**Analysis and study of entities in corporate data**	09/13-02/14
WP 2.1 Task 2.1.1	State of the art analysis of semantic entity representation models	09/13-02/14
WP 2.1 Task 2.1.2	State of the art analysis of approaches to complex entity recognition in heterogeneous data	09/13-02/14
WP 2.1 Task 2.1.3	State of the art analysis of approaches to semantic entity enrichment	09/13-02/14
WP 2.1 Milestone 2.1.1	Validation of the studies with the industrial partners	09/13-02/14

Work package FU 3	**Semantic mining of event data in corporate data for knowledge acquisition**	
Work package FU 3.1	**Analysis and study of semantic event data mining in corporate data**	09/13-02/14
WP 3.1 Task 3.1.1	Analysis of business processes for knowledge acquisition from the business process context	09/13-02/14
WP 3.1 Task 3.1.2	Study of potential event sources	09/13-02/14
WP 3.1 Task 3.1.3	Study of methods and data formats for the acquisition of event data - especially protocol data of search queries	09/13-02/14
WP 3.1 Task 3.1.4	State of the art analysis in the field of process mining and data stream processing (continuous query processing)	09/13-02/14
WP 3.1 Milestone 3.1.1	Validation of the studies with the industrial partners	09/13-02/14

Appendix B

Acknowledgment

This work has been partially supported by the “InnoProfile-Transfer Corporate Smart Content” project funded by the German Federal Ministry of Education and Research (BMBF) and the BMBF Innovation Initiative for the New German Länder - Entrepreneurial Regions.

Bibliography

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *IN: PROCEEDINGS OF THE 1993 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, WASHINGTON DC (USA*, pages 207–216, 1993.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[3] Franz Baader, Ian Horrocks, and Ulrike Sattler. Description Logics. In Frank van Harmelen, Vladimir Lifschitz, and Bruce Porter, editors, *Handbook of Knowledge Representation*, chapter 3, pages 135–180. Elsevier, 2008.

[4] Christian Borgelt. Efficient implementations of apriori and eclat. In *Proc. 1st IEEE ICDM Workshop on Frequent Item Set Mining Implementations (FIMI 2003, Melbourne, FL). CEUR Workshop Proceedings 90*, page 90, 2003.

[5] Marc Bron, Bouke Huurnink, and Maarten de Rijke. Linking archives using document enrichment and term selection. In *Research and Advanced Technology for Digital Libraries*, pages 360–371. Springer, 2011.

[6] Mr Ajay Chakravarthy, Prof Fabio Ciravegna, and Ms Vitakeska Lanfranchi. Cross-media document annotation and enrichment. 2006.

[7] S. Chakravarthy and D. Mishra. Snoop: An expressive event specification language for active databases, 1994.

[8] Sharma Chakravarthy, V. Krishnaprasad, Eman Anwar, and S.-K. Kim. Composite events for active databases: Semantics, contexts and detection. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 606–617, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[9] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey, 2007.

[10] Joong Hyuk Chang and Won Suk Lee. Finding recent frequent itemsets adaptively over online data streams. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 487–492, New York, NY, USA, 2003. ACM.

[11] Marie Chavent, Stéphane Girard, Vanessa Kuentz, Benoît Liquet, Thi Mong Ngoc Nguyen, and Jérôme Saracco. A sliced inverse regression approach for data stream. *Computational Statistics*, to appear, 2014.

[12] Ling Chen and Qingling Mei. Mining frequent items in data stream using time fading model. *Information Sciences*, 257(0):54 – 69, 2014.

[13] Yun Chi, Haixun Wang, Philip S. Yu, and Richard R. Muntz. Moment: Maintaining closed frequent itemsets over a stream sliding window. In *In ICDM*, pages 59–66, 2004.

[14] Gökhan Coskun, Mario Rothe, Kia Teymourian, and Adrian Paschke. Applying community detection algorithms on ontologies for indentifying concept groups. In *Proceedings of the 5th International Workshop on Modular Ontologies*, Ljubljana, Slovenia, September 2011.

[15] Bernardo Cuenca Grau, Bijan Parsia, and Evren Sirin. Combining OWL ontologies using $\mathcal{E}$-Connections. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(1):40–59, January 2006.

[16] Mathieu d'Aquin. Modularizing Ontologies. In Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, Enrico Motta, and Aldo Gangemi, editors, *Ontology Engineering in a Networked World*, pages 213–233. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012 edition, March 2012.

[17] Mathieu d'Aquin, Paul Doran, Enrico Motta, and Valentina A. M. Tamma. Towards a parametric ontology modularization framework based on graph transformation. In Grau et al. [25].

[18] Paul Doran, Ignazio Palmisano, and Valentina A. M. Tamma. Somet: Algorithm and tool for sparql based ontology module extraction. In *WoMO*, 2008.

[19] R.E. Filman and D.P. Friedman. Aspect-Oriented Programming Is Quantification and Obliviousness. *Workshop on Advanced Separation of Concerns, OOPSLA*, 2000.

[20] Mohamed Medhat Gaber, Arkady Zaslavsky, and Shonali Krishnaswamy. Mining data streams: A review. *SIGMOD Rec.*, 34(2):18–26, June 2005.

[21] Aldo Gangemi. Ontology Design Patterns for Semantic Web Content. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *The Semantic Web – ISWC 2005*, number 3729 in Lecture Notes in Computer Science, pages 262–276. Springer Berlin Heidelberg, January 2005.

[22] N. H. Gehani, H. V. Jagadish, and O. Shmueli. Event specification in an active object-oriented database. *SIGMOD Rec.*, 21(2):81–90, June 1992.

[23] Narain H. Gehani, H. V. Jagadish, and Oded Shmueli. Composite event specification in active databases: Model &amp; implementation. In *Proceedings of the 18th International Conference on Very Large Data Bases*, VLDB '92, pages 327–338, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

[24] Antonio Gomariz, Manuel Campos, Roque Marn, and Bart Goethals. Clasp: An efficient algorithm for mining frequent closed sequences. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *PAKDD (1)*, volume 7818 of *Lecture Notes in Computer Science*, pages 50–61. Springer, 2013.

[25] Bernardo Cuenca Grau, Vasant Honavar, Anne Schlicht, and Frank Wolter, editors. *Proceedings of the 2nd International Workshop on Modular Ontologies, WoMO 2007, Whistler, Canada, October 28, 2007*, volume 315 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[26] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Extracting Modules from Ontologies: A Logic-Based Approach. In Stuckenschmidt et al. [63], pages 159–186. DOI: 10.1007/978-3-642-01907-4.

[27] Bernardo Cuenca Grau, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Automatic Partitioning of OWL Ontologies Using $\mathcal{E}$-Connections, 2005.

[28] IEEE Architecture Working Group. IEEE standard 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE, 2000.

[29] Thomas R. Gruber. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human-Computer Studies - Special Issue: the role of formal ontology in the information technology*, 43(5-6):907–928, December 1995.

[30] Michael Gruninger and Jintae Lee. Ontology – Applications and Design. *Communications of the ACM – Special Issue: Ontology: different ways of representing the same concept*, 45(2):39–41, February 2002.

[31] Christian W. Günther and Wil M. P. Van Der Aalst. Fuzzy mining: Adaptive process simplification based on multi-perspective metrics. In *Proceedings of the 5th International Conference on Business Process Management*, BPM'07, pages 328–343, Berlin, Heidelberg, 2007. Springer-Verlag.

[32] Jiawei Han, Hong Cheng, Dong Xin, and Xifeng Yan. Frequent pattern mining: Current status and future directions. *Data Min. Knowl. Discov.*, 15(1):55–86, August 2007.

[33] Sebastian Hellmann, Jens Lehmann, Sören Auer, and Martin Brümmer. Integrating nlp using linked data. In *The Semantic Web–ISWC 2013*, pages 98–113. Springer, 2013.

[34] Richard Hull and Dean Jacobs. Language constructs for programming active databases. In *Proceedings of the 17th International Conference on Very Large Data Bases*, VLDB '91, pages 455–467, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.

[35] Ruoming Jin and Gagan Agrawal. Frequent pattern mining in data streams. In Charu C. Aggarwal, editor, *Data Streams - Models and Algorithms*, volume 31 of *Advances in Database Systems*, pages 61–84. Springer, 2007.

[36] Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. Semantic Modularity and Module Extraction in Description Logics. In *Proceedings of the 2008 conference on ECAI 2008: 18th European Conference on Artificial Intelligence*, pages 55–59, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press.

[37] Roman Kontchakov, Frank Wolter, and Michael Zakharyaschev. Logic-based ontology comparison and module extraction, with an application to DL-Lite. *Artificial Intelligence*, 174(15):1093–1141, October 2010.

[38] Michael David Lee, BM Pincombe, and Matthew Brian Welsh. An empirical evaluation of models of text document similarity. *Cognitive Science*, 2005.

[39] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, et al. Dbpedia-a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2013.

[40] David Luckham and W. Roy Schulte. Event processing glossary version 2.0, 2011.

[41] Gurmeet Singh Manku and Rajeev Motwani. Approximate frequency counts over data streams. In *Proceedings of VLDB'02*, pages 346–357, 2002.

[42] Alessandro Margara, Gianpaolo Cugola, and Giordano Tamburrelli. Towards automated rule learning for complex event processing, 2013.

[43] Alessandro Margara, Gianpaolo Cugola, and Giordano Tamburrelli. Learning from the past: Automated rule generation for complex event processing. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, DEBS '14, pages 47–58, New York, NY, USA, 2014. ACM.

[44] Enrico Motta, Simon Buckingham Shum, and John Domingue. Ontology-driven document enrichment: principles, tools and applications. *International Journal of Human-Computer Studies*, 52(6):1071–1109, 2000.

[45] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[46] Christine Parent and Stefano Spaccapietra. An Overview of Modularity. In Stuckenschmidt et al. [63], pages 5–23. DOI: 10.1007/978-3-642-01907-4.

[47] Adrian Paschke, Gökhan Coskun, Marko Harasic, Ralf Heese, Radoslaw Oldakowski, Ralph Schäfermeier, Olga Streibel, Kia Teymourian, and Alexandru Todor. Corporate semantic web report vi: Validation and evaluation. Technical Report TR-B-13-01, Freie Universität Berlin, 2013.

[48] Adrian Paschke, Paul Vincent, Alex Alves, and Catherine Moxey. Tutorial on advanced design patterns in event processing. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS '12, pages 324–334, New York, NY, USA, 2012. ACM.

[49] Adrian Paschke, Paul Vincent, and Florian Springer. Standards for complex event processing and reaction rules. In *Proceedings of the 5th International Conference on Rule-based Modeling and Computing on the Semantic Web*, RuleML'11, pages 128–139, Berlin, Heidelberg, 2011. Springer-Verlag.

[50] Jyotishman Pathak, Thomas M. Johnson, and Christopher G. Chute. Survey of modular ontology techniques and their applications in the biomedical domain. *Integrated Computer-Aided Engineering*, 16(3):225–242, January 2009.

[51] Jian Pei, Jiawei Han, Behzad Mortazavi-asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei chun Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering*, ICDE '01, pages 215–, Washington, DC, USA, 2001. IEEE Computer Society.

[52] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March 1986.

[53] Ellen Riloff, Rosie Jones, et al. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, pages 474–479, 1999.

[54] Giuseppe Rizzo and Raphaël Troncy. Nerd: a framework for unifying named entity recognition and disambiguation extraction tools. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 73–76. Association for Computational Linguistics, 2012.

[55] Robert Sanderson, Paolo Ciccarese, and Herbert Van de Sompel. Designing the w3c open annotation data model. In *Proceedings of the 5th Annual ACM Web Science Conference*, pages 366–375. ACM, 2013.

[56] Robert Sanderson, Paolo Ciccarese, and Herbert Van de Sompel. Open annotation data model. *Community Draft*, 8, 2013.

[57] Ralph Schäfermeier. Aspect-Oriented Ontology Development. In Witold Abramowicz, editor, *Business Information Systems Workshops*, number 160 in Lecture Notes in Business Information Processing, pages 208–219. Springer Berlin Heidelberg, 2013.

[58] Ralph Schäfermeier and Adrian Paschke. Towards a Unified Approach to Modular Ontology Development Using the Aspect-Oriented Paradigm. In *7th International Workshop on Modular Ontologies (WoMO) 2013*, pages 73–78, 2013.

[59] Anne Schlicht and Heiner Stuckenschmidt. A Flexible Partitioning Tool for Large Ontologies. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '08, pages 482—488, Washington, DC, USA, 2008. IEEE Computer Society.

[60] Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.

[61] Ramakrishnan Srikant and Rakesh Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '96, pages 3–17, London, UK, UK, 1996. Springer-Verlag.

[62] Friedrich Steimann. Domain Models Are Aspect Free. In Lionel Briand and Clay Williams, editors, *Model Driven Engineering Languages and Systems*, number 3713 in Lecture Notes in Computer Science, pages 171–185. Springer Berlin Heidelberg, January 2005.

[63] Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2009. DOI: 10.1007/978-3-642-01907-4.

[64] Boontawee Suntisrivaraporn. Module Extraction and Incremental Classification: A Pragmatic Approach for $mathcalEL^+$ Ontologies. In Sean Bechhofer, Manfred Hauswirth, Jörg Hoffmann, and Manolis Koubarakis, editors, *The Semantic Web: Research and Applications*, number 5021 in Lecture Notes in Computer Science, pages 230–244. Springer Berlin Heidelberg, January 2008.

[65] Kia Teymourian and Adrian Paschke. Semantic rule-based complex event processing. In *RuleML 2009: Proceedings of the International RuleML Symposium on Rule Interchange and Applications*, 2009.

[66] Kia Teymourian and Adrian Paschke. Towards semantic event processing. In *DEBS '09: Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, pages 1–2, New York, NY, USA, 2009. ACM.

[67] Kia Teymourian and Adrian Paschke. Plan-based semantic enrichment of event streams. In Valentina Presutti, Claudia d'Amato, Fabien Gandon, Mathieu d'Aquin, Steffen Staab, and Anna Tordai, editors, *ESWC*, volume 8465 of *Lecture Notes in Computer Science*, pages 21–35. Springer, 2014.

[68] Kia Teymourian, Malte Rohde, and Adrian Paschke. Fusion of background knowledge and streams of events. In François Bry, Adrian Paschke, Patrick Th. Eugster, Christof Fetzer, and Andreas Behrend, editors, *DEBS*, pages 302–313. ACM, 2012.

[69] Dhavalkumar Thakker, Vania Dimitrova, Lydia Lau, Ronald Denaux, Stan Karanasios, and Fan Yang-Turner. A priori ontology modularisation in ill-defined domains. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 167–170, New York, NY, USA, 2011. ACM.

[70] Wil van der Aalst. Process mining: Overview and opportunities. *ACM Trans. Manage. Inf. Syst.*, 3(2):7:1–7:17, July 2012.

[71] Denny Vrandečić. Wikidata: a new platform for collaborative data collection. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 1063–1064. ACM, 2012.

[72] Xifeng Yan, Jiawei Han, and Ramin Afshar. Clospan: Mining closed sequential patterns in large datasets. In *In SDM*, pages 166–177, 2003.

[73] Jeffery Xu Yu, Zhihong Chong, Hongjun Lu, and Aoying Zhou. False positive or false negative: Mining frequent itemsets from high speed transactional data streams. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB '04, pages 204–215. VLDB Endowment, 2004.

[74] Mohammed J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Mach. Learn.*, 42(1-2):31–60, January 2001.