Aptness on the Web

een wetenschappelijke proeve op het gebied van de
Natuurwetenschappen Wiskunde & Informatica

Proefschrift

ter verkrijging van de graad van doctor
aan de Radboud Universiteit Nijmegen
op gezag van de Rector Magnificus prof. dr. C.W.P.M. Blom,
volgens besluit van het College van Decanen
in het openbaar te verdedigen op
vrijdag 8 december 2006 om 10.30 uur precies

door

Bas van Gils
geboren op 6 december 1976
te Tilburg

SIKS Dissertation Series No. 2006-21
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems, and IRIS, the Information Retrieval and Information Systems group of Radboud University Nijmegen.

Netherlands Organisation for Scientific Research

This research has been partially funded by the Netherlands Organisation for Scientific Research (NWO) via project PRONIR.

ISBN: 90-9021091-1
Ontwerp omslag: Hans van Gils

A good Book is the precious life-blood of a master spirit, embalmed and treasured up on purpose to a life beyond life.

(This motto from Milton's Areopagitica is placed over a door in the public library of New York, NY, USA)

PREFACE

I actively started looking for a PhD position, while I was in the process of finishing my masters thesis at the Infolab at *Tilburg University* in December 2001. It had taken me years to decide whether I wanted to go to industry or stay at the university after graduation. I was inspired by great motivators such as Kees Takkenberg, Mike Papazoglou, and Jeroen Hoppenbrouwers. I hung around the Infolab frequently and got to know most people working there. At this time I also met Jeroen's brother Stijn who was working on his dissertation at the time. We had many interesting discussions back then about PhD projects, research, and science in general.

During the last year of my studies in Tilburg I took some classes at the faculty of arts where I met Hans Paijmans. His class on *information retrieval* was so interesting that I decided I wanted to know more about the topic. It therefore made sense to start looking for a PhD position in the area of information retrieval which I found in Nijmegen. The PRONIR project (Profile-based Retrieval Of Networked Information Resources) had just started and they were still looking for a PhD student. This project turned out to be a co-operation between the University of Tilburg, particularly the Infolab, and the university of Nijmegen. This meant that I moved on from being a student at the Infolab to being a project partner to my old collegues; Kees Leune in particular.

In Nijmegen we started out with a small group: Erik Proper, Patrick van Bommel, and myself worked on several problems, all of which are discussed in this dissertation. We quickly found a work-mode that turned out to be both pleasant and productive. During the weekly meetings we managed to generate new ideas for papers as well as this dissertation. Even more, Erik and Patrick have been great mentors; they introduced me to the mores of academia, helped me in establishing a healthy attitude towards research, and deepend my knowledge and skills regarding modeling. In doing so I managed to combine the two fields that I find most interesting: information retrieval and (conceptual) modeling. I am in dept to them for introducing me to modeling techniques such as ORM/PSM as well as stimulating me to learn more about mathematics.

Theo van der Weide got involved with our research when we were about ready to submit our first journal paper. One of Theo's strong points is the so called "nitty gritty" of scientific research, especially when it comes to (mathematical) models. The discussions with him really were an eye-opener in the sense that I quickly learned to appreciate the value of elegant, precise modeling. Time and again we went over every little detail untill it was just right. As *Goethe* once said: "*In der Beschränkung zeigt sich der Meister*". Even though I may have found the endless discussions about little details tedious and difficult at times, in the end it turned out to be for the best. Even more, as my skills and insights grew it actually was a lot of fun.

Approximately halfway through the project, Mario van Vliet also started participating in our discussions when we started discussing the details of the information market paradigm. Mario has a sharp eye and always manages to ask the right questions to steer a research project in the proper direction. Working with him was both enlightening and fun.

Throughout the entire project we co-operated with our project partners in Tilburg whenever possible. This allowed me to present my work to my former teachers and engage in discussions regarding our mutual research. These were particularly useful since the group in Tilburg (at the faculty of economics) has a different perspective than our group in Nijmegen (at the faculty of science). In particular I would like to thank Kees Leune, Mike Papazoglou, Jeroen Hoppenbrouwers, and Willem-Jan van den Heuvel for their friendship and support.

Many more people from academia contributed in one way or another to this dissertation in one way or another. Listing them all is sheer impossible and I would probably even forget to mention half of them. However, a few must be mentioned here. First of all, I am in dept to Stijn Hoppenbrouwers; he has been a close friend ever since I started my studies and has been a great inspiration ever since. Eric Schabell started working on the PRONIR project as our scientific programmer. He has a wonderful attitude to life and I loved working with him. Finally I would like to thank the proof readers of my dissertation for the patience to go over large parts of the text and provide me with helpfull comments and suggestions: Perry Groot, Frans Henskens, Robert Helgesson, and Jaqueline Dake.

The last few years my family had to endure quite a bit in terms of strange monologues about my research, strange moods, and fatigue during difficult periods, not to mention many trips abroad for conferences and such. Without the support of my wife, son, and family the project would have come to a grinding halt early on. I want to thank them for their love and support over the past years which has been a tremendous inspiration indeed.

Last but not least, the following quote from Gottfried Wilhelm Leibniz provides a good characterization of the nature of this dissertation: "*When a truth is necessary, the reason for it can be found by analysis, that is, by resolving it into simpler ideas and truths until the primary ones are reached*".

Bas van Gils
Deventer, July 2006

CONTENTS

LIST OF FIGURES

CHAPTER 1 Introduction

1.1 Background

The World Wide Web (the Web) has become increasingly important over the last few decades. What started as a medium for communication between scholars has evolved into one of the most important media in modern days. Several factors have contributed to this development. First of all, the sheer amount of resources available to us has exploded over the last few years. In [SYB98] it is even called an "explosion of online information". It is sometimes stated that anything can be found on the Web. Certainly, resources are available on many different topics. Not only the size of the Web, but also its usage (ranging from online communication via E-mail and instant messaging to E-government and E-business) has evolved. The Web is no longer a mere "static library" with information. People also use it for communication purposes, for making travel arrangements, managing their bank accounts, for performing business transactions etcetera. Last but not least, the kinds of resources available online have evolved also and include web pages, online databases, E-services and other interactive applications (See e.g. [Day00, PPY01]).

To cater for all these changes, the technical infrastructure supporting the Web has evolved over the years as well; a myriad of standards exist for localization and transportation of information over the Web. Examples include:

Localization – search engines, yellow-pages, service repositories

Transportation – HTTP, FTP, JABBER, VOIP

This is illustrated in Figure 1.1 which is inspired by [HP99, GPB04]. The left side of the figure signifies *information supply* which is heterogeneous in nature. The right side of the figure signifies *information demand* stemming from the knowledge intensive tasks that users of the Web wish to perform. Last but not least, the middle part of the figure signifies the *information market* where the localization and transporting of resources is done. In

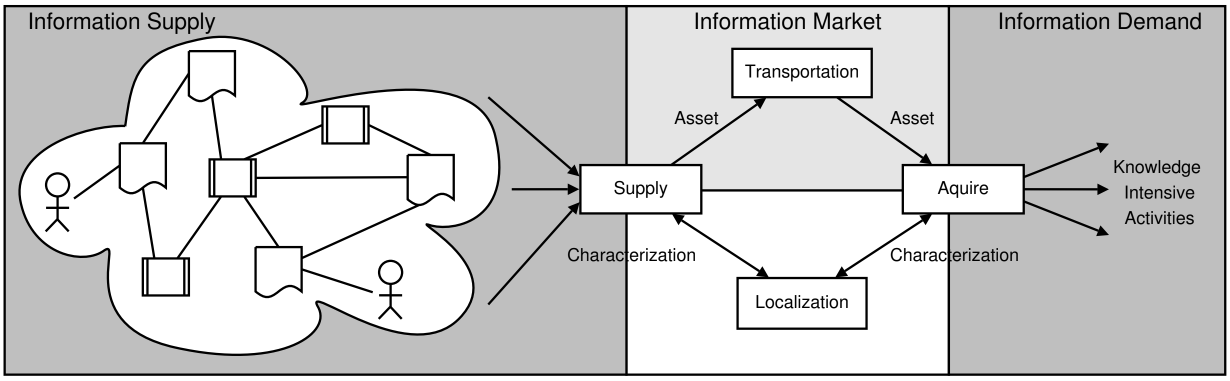


Figure 1.1: The information market

this dissertation we will mainly focus on information supply and the information market. More specifically, we will look at information supply from a market point of view in order to study the retrieval of resources on the Web.

Many tools exist to assist us when performing our tasks on the Web, with search engines (such as GOOGLE) being the most prominent example. We expect these tools to assist us while taking into account the way we access the web (transport) as well as all aspects of our information needs (brokering). In our opinion it seems as if almost all current tools focus solely on the informational aspects of ones information need; i.e., *topical relevance* as used in traditional information retrieval (IR) research [Rij75, SM83, BYRN99]. The traditional IR paradigm is illustrated in Figure 1.2. In this paradigm documents, or

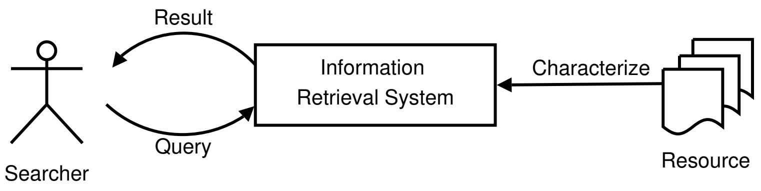


Figure 1.2: The information retrieval paradigm

possibly, resources are characterized after which a query (typically a set of keywords) is matched against these characterizations. The same seems to hold for information retrieval on the Web (web retrieval).

According to [GP99], the three main purposes of web retrieval systems is to gather an information base from which the user can retrieve information, to represent these pages in a fashion that attempts to capture their *content*, and finally to allow the searchers to issue queries such that the relevant pages from the universe are found. In other words, documents are matched against queries based on content, which in practice means keywords. Similarly, in [ZG00] it is observed that "most information retrieval systems on the Internet rely primarily on similarity ranking algorithms based solely on term frequency statistics". In Sections 1.2.2 and 1.2.3 we will further discuss information retrieval and web retrieval respectively.

Given the strong focus on topicality, it seems to be the case that other aspects that might influence the search process are often ignored. Some examples in this respect are: the amount of knowledge that a searcher already has on a certain topic; the amount of time she has available to search and read the resource; the software the user has available to actually acces the resource; the bandwith; the languages that a searcher speaks, etcetera. Several toolmakers have also recognized this and have extended their tools with additional capabilities. For example, with *Google*'s advanced search it is possible to also specify such things as file type. Even more it offers automatic conversion between some of these file types. Similarly, *GMail* adapts its user interface depending on browser capabilities and the device used to access its services.

We dub the observation that modern search tools (almost) solely focus on informational aspects of the information need of searchers, rather than taking the entire information need into account, to be the *aptness problem*, which is the main theme of this dissertation. In the upcoming sections we will see many examples which illustrate the aptness problem in concrete situations. The remainder of this chapter is organized as follows: In Section 1.2 we will present an overview of literature related to the information market which presents our *way of thinking* (i.e., our core paradigm). In Section 1.3 we will present a more detailed view on the aptness problem by means of examples. Finally, in Section 1.4 the focus will be on our research questions, ambitions and approach.

1.2 Literature

There are several ways to look at the information market. To illustrate this we present an overview of ways to look at this phenomenon based on literature. The main goal of this section, hence, is to position our work and firmly ground it in literature.

1.2.1 Structured data access

Computers have been used for storing and retrieving structured data for many years, most notably by using (relational) databases. In [CB02, Dat03] it is even claimed that the history of database research has led to the database system becoming arguably the most important development in the field of software engineering.

A theoretical perspective on information supply, from the perspective of structured data access, is provided by the relational algebra [Ull89] and its 'real life' implementation SQL (Structured Query Language). SQL is the industry-standard language for creating, updating, and querying relational database management systems. In other words, SQL is the de facto standard for querying *structured data* stored in relational database management systems (RDBMS) such as *Oracle* and *Ingres*. As such, relational algebra, and more specifically SQL, provides a viewpoint on that part of the information market which is represented in terms of a relational database management system.

A more recent approach is described in [MM97], where the focus shifts towards the relation between query languages and the Web. As such this work provides a view on the brokering aspect of the information market. In this work it is observed that resources on the Web (typically *Html*-documents) are less structured than, for example, a database schema. This has strong implications for the way we access them. The authors discuss two important features that distinguish the Web from traditional databases: the navigational nature of the Web and the lack of concurrency control. To overcome the problems associated with these two features, a new formalism for *query computation* is developed, which results in a *web calculus*. For this purpose, the web is modeled as a relational database containing a relation of *Node* objects, one per document, and a relation of *Link* objects, one per node-to-node link.

Another important difference between classical databases and the Web is the fact that traditional databases are based on a closed world semantics whereas in case of the Web one is forced to adopt open world semantics. The first view simply means that everything we don't know is considered to be false, whereas in the second view that every thing we don't know is undefined. This distinction is particularly important when trying to reason about resources on the Web. See for example [AH04, DNR02].

Besides relational databases, much research has been conducted in the area of object-oriented databases (OODB) also (see [PRBV90, ZCC95], and [SZ87] for the differences between relational and object-oriented database systems). Even though RDBMS and OODB are two distinct strategies to provide means of storing and retrieving data, they are essentially *implementations* of the same perspective on information supply: try to structure and store data in such a way that it can be queried and retrieved effectively.

1.2.2 Information retrieval

The field of *information retrieval* (IR) was already briefly introduced. In this field, the goal is to retrieve those documents / resources from a collection that are relevant with regard to a user query which is presumed to represent the *information need* of the user. More specifically, for each resource the *relevance* with regard to the query is computed after which the documents are ranked. Documents with a similarity value greater than a certain threshold are presumed to be relevant; documents with a lower similarity value are not. Depending on the way the documents are represented in the IR system, different ranking algorithms can be used (see e.g., [Rij75, SM83, BYRN99]). Furthermore, the constraint that resources must adhere to a certain structure such as a relational database schema is relaxed.

In traditional IR, the retrievable resources consist of textual documents only. Each document can be described in terms of a set of keywords which are either assigned manually or extracted from the document automatically. Queries are also in the form of several keywords and, optionally, uses Boolean connector operators such as `and`, `not` and `or`. Matching is done by finding those documents in which the keywords occur (or do *not* occur, if the `not`-connector is used). A more advanced methodology is the *vector space model*, of which [Pai99] attempts to give an overview. In the vector space model, both

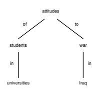


Figure 1.3: Example index expression

documents and queries are represented as vectors of keywords using a weighting scheme such as the binary scheme in which a position in the vector is assigned a 1 if the word occurs in the document and a 0 if it does not or a more advanced scheme such as the *tf.idf* class of word weights. The ranking algorithm simply measures the distance between the query vector and the document vectors: the closer a document vector is to the query vector the more relevant this specific document is with regard to the query and vice versa. Recently, similar approaches have been developed for image retrieval, video retrieval, audio retrieval and even E-service discovery.

Another way of indexing / characterizing (textual) resources uses *index expressions*, which are an extension to *term phrases* whereby the relationships between terms are modeled [Bru90, BW90b]. Consider the phrase *The attitude of students in universities to the war in Iraq.* In a 'normal' keyword-based approach, a representation of this phrase would contain (stemmed / normalized) words: {*attitude, student, university, war, Iraq*}. The representation in an index expression is much more semantically rich; it describes the meaning of the original sentence better:

attitudes of (students in (universities)) to (war in (Iraq)).

Figure 1.3 graphically shows this index expression. With index expressions one achieves a mechanism for information disclosure and hence, the possibility for more accurate retrieval.

1.2.3 Retrieval on the web

With the rise of the Web, the need for a more elaborate view on retrieval rose. Search engines such as GOOGLE attempt to index the heterogeneous resources found on the Web as well as possible. GOOGLE can index several formats found on the Web such as *Html*, *Pdf*, *Word* and also graphical formats such as *Png* and *Jpeg*. Furthermore, GOOGLE is esteemed for its elaborate ranking system called *PageRank* [BP98, PBMW98]. This ranking system is based on the notion of importance. More specifically: GOOGLE makes heavy use of the (link) *structure* present in hypertext for *indexing* and *ranking* web pages in the sense that web pages that are referred to a lot, are likely to be important and thus end up higher in the result set.

The above mentioned Web-based search engines are generally perceived to work rather well. Based on a series of experiments, the authors of [LH03] observe that "individual computer experience, quality of search systems, motivation and perceptions of technology acceptance are all key factors that affect individual feelings to use search engines as an information retrieval tool". Their data, indeed, suggests that most users generally perceive search engines to be both useful and easy to use.

However, in many cases search engines tend to return too many (references to) potentially relevant resources. Users are still required to manually wade through large result sets in search of truly relevant assets. A second problem lies in the area of vocabulary that is used. Practical studies have shown that there is a critical mismatch between a users' vocabulary, and the vocabulary used on the Web [SC96]. Picking the right query terms depends on how familiar searchers are with the vocabulary used in documents they wish to retrieve, which is not always straightforward. *Query by navigation* approaches may help users in selecting the proper keywords for their search. See e.g., [Ber98, BBWW98] for more details on this technique.

Recent approaches have tried to overcome problems related to the heterogeneity and diversity of websites spread all over the Web. For example, in [Fau00] the *Hyperview* approach is discussed. In this approach, a *virtual web site* is presumed to contain concentrated information that has been extracted, homogenized, and combined from several underlying web pages, with the purpose of saving users the trouble of searching for, and browsing through all these (underlying) pages. These three steps are treated uniformly as consecutive "views that map between different levels of abstraction", where the views are represented as graphs. Transitions from one view to another are achieved by means of graph transformations.

1.2.4 Digital libraries

Digital Libraries (DL) often make use of retrieval techniques. A DL is a collection of services as well as a collection of information objects. The services support users in dealing with information objects and the organization and presentation of those objects available directly or indirectly via electronic / digital means [Lei98]. Simply put, a DL provides users with an infrastructure including a bulk of digital resources (which may be heterogeneous) and services that are needed to access these resources. Functionality ranges from searching and browsing to transporting the resources to the user [FHJ01]. The fact that heterogeneity of resources poses constraints on how you access them is recognized in the DL community as well. In [Arm95], for example, it is stated that the architecture underlying the DL should be separated from the content stored in it; the architecture must provide a (uniform) way to specify characteristics of each type of resource. This general mechanism may be extended for specific types of resources.

As with traditional (brick and mortar) libraries, digital libraries deal with a well-known collection of resources. This is quite different from the situation on the Web. From a meta data perspective it is easier to annotate the resources in such a controlled environment e.g. with *Dublin Core* [OAS02, ISO86, WKLW98] which, in a way, adds more structure to the library. For each resource in the library both the *aboutness* is captured (by using keywords, index expressions or other means.) as well as data about the resource (such as the author, year of publication, etcetera). From an information market perspective, DL's attempt to provide advanced techniques to describe and characterize heterogeneous digital resources in a controlled environment.

1.2.5 Meta data

Meta data is data about data. It can be useful to know several things about data such as its structure, last modification date, etcetera. A myriad of standards for meta data exist, especially in the business domain (See also Section 1.2.4). Examples are:

CWM (Common Warehouse Metamodel) is a specification that describes meta data interchange among data warehousing, business intelligence, knowledge management and portal technologies. It provides a framework for "representing meta data about data sources, data targets, transformations and analysis, and the processes and operations that create and manage warehouse data and provide lineage information about its use" [Obj01].

UDDI (Universal Description, Discovery, and Integration) is a standard for locating web services by enabling robust queries against rich meta data. In summary, meta data about web services are stored in repositories. The information provided in a listing consists of three conceptual components: "white pages" of company contact information; "yellow pages" that categorize businesses by standard taxonomies; and "green pages" that document the technical information about services that are exposed [Ste02, JM02].

ebXML (Electronic Business using eXtensible Markup Language) , is a modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet [OAS03]. The ebXML specification provides a standard infrastructure for sending business messages across the internet.

In terms of the information market, these standards focus on describing (aspects of) information supply. Important insights about what information supply is can be gained by studying these standards and how they are applied in practice.

1.2.6 Semantic Web

After the success of the Web, a new form of Web content that is meaningful to computers will unleash a revolution of new possibilities [BLHL01]. One of the problems with the current Web is that the available data (web content) is designed to be read by humans, rather than to be interpreted / manipulated meaningfully by computers. The vision of the Semantic Web is that the current Web must be extended so that the available data are given a well-defined meaning. By doing so, computers (and other agents such as hand-held devices, mobile phones etcetera) can co-operate to perform knowledge-intensive tasks:

> The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation. The first steps in weaving the Semantic Web into the structure of the existing Web are already under way. In the near future, these developments will usher in significant new functionality as machines become much better able to process and "understand" the data that they merely display at present.
>
> —*Taken from: [BLHL01]*

In this context, several techniques have been proposed to model the Web many of which are supported by the *World Wide Web Consortium.* (W3C)[1] and the Semantic Web community (see e.g. [BLHL01, AH04]). The most important results in this area are probably the *Resource Description Framework* (RDF), RDF*Schema* (RDFS) and the *Ontology Web Language* (OWL) [KC04, BG04, MH04].

Simply put, the core concepts for RDF are resources, properties, and statements. Resources can be thought of as things one may want to talk about such as authors, books or web sites. They are identified by means of a URI. Properties are a special kind of resources that describe the relations between different resources such as "written-by" or "has owner". Last but not least, statements assert the properties of resources and are object-attribute-value triples. An example statement would be:

$$\langle \text{John Doe}, \texttt{http://my.domain/site-owner}, \texttt{http://my.domain/foo.html} \rangle$$

which would be a representation of the fact that a John Doe is the owner of a certain web page. Note that RDF is a very simple language that is roughly limited to binary predicates. However, it provides an elegant way for representing characteristics of resources since it lets users describe them using their own vocabulary. Hence, RDF does not make

[1]See `http://www.w3.org`

any assumptions about the underlying application domain. This is where RDFS comes in. The main distinction between the two is that RDF is used to annotate particular *instances* and RDFS is used to model the behavior of *classes of instances*. For example, using RDFS one would be able to express the fact that people can own web pages. In other words, RDFS is used to model relations between classes of resources. In doing so it is (roughly) limited to subclass hierarchies and property hierarchies.

By contrast, OWL goes a step further and can be considered an extension of RDFS that has evolved from DAML and OIL [Dam03, CHH$^+$01]. Ontology languages in general allow users to create conceptualizations of domain models. As such, the requirements for ontology languages include a well-defined syntax, formal semantics, convenience of expression, efficient reasoning support and sufficient expressive power [AH04]. Note that some of these requirements seem to conflict; for example: as efficient reasoning and convenience of expression. Therefore, three different sub languages of OWL have been defined: OWL Full, OWL DL and OWL Lite. Each of these sub languages is geared towards different aspects of the above requirements. A discussion of the intricacies of these (sub)languages is beyond the scope of this dissertation.

1.2.7 Adaptive hypermedia

Since the late 1970's a trend has emerged to adapt the behavior of systems to specific users or groups of users. The fields that are of interest to us are *user modeling* and *adaptive hypermedia*. In [Bru01] it is argued that the field of adaptive hypermedia is "on the crossroads of hypermedia and user modeling". Since we are mainly interested in user modeling in the context of the Web we can safely use the terms interchangeably in this section. We will base our discussion mainly on reviews of the field (such as [Bru01, Bru96, McT93, SH05, Kob01]) application in recommender systems [MLL03] and decision support systems [HCD05].

User models can be seen as a knowledge source which contains assumptions about users. This knowledge may comprise different aspects such as the goals and plans of users, capabilities of users, preferences, beliefs, knowledge, and so on. Adaptive systems, then, use these models of individual users throughout interactions for adaption to the needs of these specific users. Typical issues with respect to user models are:

- How are they controlled and managed?
- Are they for individuals or for groups of users?
- Are they static or dynamic?
- Are they implicit or explicit?
- Do they contain data about the user, about the interactions between the user and the system, or about the environment?

When designing an adaptive system these questions will have to be answered. For example, for retrieval systems:

> The goal of search oriented systems is to create a list of links to documents that satisfy the user's current information request. Unlike simple "one-shot" search engines, adaptive information retrieval systems take into account not only the set of words specifying the current request, but also long-term or/and short-term model of user's interests and preferences.
>
> —*Taken from: [Bru01]*

This suggests that in adaptive information retrieval systems, the combination of a user model and the current set of query terms together represent the information need of the user. As such these user models can be seen as a generic view on the demand side of the information market.

1.2.8 Information modeling

Finally, conceptual information modeling techniques such as ER [Che76], EER [Gog94], ORM [Hal01], UML Class Diagrams [BRJ99] and their associated query/constraint languages such as RIDL [Mee82], CADDY [HE92, HE90], LISA-D [HPW93], ConQuer [BH96], can be used to provide a conceptual model of some given application domain. This conceptual model is usually formulated in terms of a set of entity and relationship types, describing the essence of the application domain.

Usually, the conceptual model is translated to some implementation model such as a relational database schema. In this case, the information supply consists of the representation of the state of affairs of some domain. For example, the state of affairs with regard to car-prices, airplane flights, enrollments of students, etcetera. The actual information then corresponds to the contents of the underlying relational database. The original conceptual model would allow us to view this relational database as a conceptual database, restating the contents of the relational database as a population of the relationship types as identified in the conceptual schema. In other words the conceptual model provides us with a conceptual representation of the state of affairs in a domain. As such it could very well be used to represent the state of affairs on the Web.

1.3 Issues

In the previous section we have, among other things, discussed that tools on the information market should take the *entire information need* of a searcher into account when performing their tasks. This is a departure from the traditional IR paradigm where the sole focus seems to be on topical relevance. Surely, modern search engines offer more advanced capabilities as well. For example, in the case of GOOGLE, the advanced search also includes (limited) support for constraining the search with file-types, selecting a language as well as date-related information such as the modification date. One can only wonder: "What will be next?". More precisely formulated: what other aspects should be included in ones search for resources on the Web? Another important question in this respect is: For what kind of search, or in what situations, are these additional capabilities useful? The following examples illustrate different manifestations of these issues involved and are thus examples of manifestations of the aptness problem:

Bandwidth & medium – Assume that you are on the road and are in urgent need of some last-minute information on the price of your stock and that of your most important competitors. The only device that you have with you is your PDA with which you can connect to the Internet using your mobile phone. Using your favorite search engine you manage to quickly locate a relevant spreadsheet which happens to be rather large since it also stores profiles of the companies, the histories of their respective stock etcetera. In terms of the *content* this may very well be the perfect document for you. However, is it feasible to download such a large spreadsheet on a slow connection? Does the PDA have the proper software to view such a file? In [AK00] it is observed that "with the increasing popularity of handhelds, future web sites must accommodate handheld access". In this work it is observed that several aspects (such as screen size, but also limitations on human short-term memory) have a serious impact on the way information can be accessed by users.

A similar situation occurs when one has a slow or possibly buggy connection such as, for example, in the outback or in some developing countries. Even though one may have access to a desktop computer with all the necessary software to view certain resources, bandwidth limitations may be a serious problem when trying to access certain information. Similarly, even if you have sufficient bandwidth, if certain software is lacking then documents can not be viewed at all, no matter how useful they could be.

Background knowledge & organizational role – Assume two people are searching the Web for information on a certain technology, say, search engines. One of them is the manager responsible for deciding the functionality that will be offered on the corporate web site, the other a technician responsible for implementing a search engine. It is likely that these people will have (very) different information needs, despite the fact that they may use the same keywords for their search.

The above illustrates that the *role* of the searcher (or, to put it differently, the purpose for which the information will be used) is an important factor in the search process. A similar, yet somewhat different, factor is the amount of background knowledge someone already has. Consider the case where one searches for information on *description logic*. A scholar with profound mathematical background is likely to want something with more depth than someone without this background.

Time considerations – In some cases, time is of the essence. In such situations you may not have the time to search for the perfect document for your current information need, to wade through the list of possibly relevant documents, and to convert them into a form where you can digest the information as soon as possible. This may include such things as creating a summary of a long text or to let the computer read a text out-loud while you are doing other things.

Cost – There is a famous adage that goes "Information wants to be free" (See e.g., [Cla01]). This statement seems to be a paradox. In [Var95] provides a discussion on how "information goods" such as documents or stock information should be priced. If value is defined to be "willingness to pay" then it is interesting to observe that the value of information goods drops to zero once the information goods is viewed by a consumer.

Sure enough, many resources are available for free on the Web[2]. This is not always the case, however. Some resources can only be viewed after paying a fee, or after a subscription. The willingness of someone to pay (or not) for certain information puts an additional restriction on the resources that are suitable for this specific searcher.

The above examples illustrate that keyword-based search on the Web is not enough in many cases. Note that the list of examples is not comprehensive, many more examples could be included. We therefore propose to move from keyword-based search (or *search using topical relevance*) to aptness-based search; an approach to search on the Web in which the entire information need of searchers is taken into account.

1.4 Research questions and approach

In this section we present the research questions and the approach that have guided the research presented in this dissertation. We also explain which chapters contribute to answering these questions.

1.4.1 Research questions

In previous sections we have outlined the *aptness problem* which can be summarized as follows: current approaches to (information) brokering on the Web are too limited in the sense that they often take only topical relevance into account. As we stated previously, we feel that it is time for a change; the present situation is not satisfactory. In our opininion users should be able to specify more (in an ideal situation: all) aspects of their information need; search brokers should be able to deal with these semantically rich queries to be able to select resources that are *apt* rather than relevant. Therefore, we propose a more broader approach to brokering which we have dubbed *aptness-based search.* It is, however, not apparent what this means precisely, and how this can be achieved. Hence, the central research question for this dissertation is formulated as follows:

> What is aptness-based search on the Web and how can it be achieved?

[2]The word *free* has the intended meaning *free of charge.*

In order to answer this question we will break it down into smaller questions. Firstly we will return to our earlier observation that (search on) the Web bears some similarity to an *information market*. Our first research question is related to this observation and is intended to set the stage for the remainder of this dissertation:

Q 1 *What is the information market?*

In answering this question we will create a formal model for the information market. This model will provide us with a vocabulary to analyze the aptness problem further. The concept of *value* will play a key role in this model; deciding how apt a resource on the Web is to a searcher is to a certain extent equivalent to deciding how valuable an asset is to an actor.

Valuation (of resources on the Web) is a problem in itself that can be tackled in different ways. A first approach would be to consider it from an information demand point of view. This would put the consumer/ searcher in the spotlight and would involve such things as user modeling (what does the user want?), query formulation (how can the user formulate his needs into a query that is meaningful to the search system) etcetera. Another approach would be to start with information supply. In this case a model for information supply would be the starting point. By carefully examining information supply we hope to be able to derive possible aspects of aptness-based search on the Web. We will take the second approach which leads to:

Q 2 *What is information supply?*

As we stated before, the notion of value plays an important role in market thinking. The notion of *value addition* by participants in a value/ production chain is, therefore, also important. The basic assumption, in economics, is that in each step of a production process some value is added to a product. Translated to the information market, this implies that it should be possible to manipulate resources (from information supply) such that they become more valuable. This leads to the research question:

Q 3 *How can information supply be manipulated?*

As we already briefly mentioned above, many players on the information market have recognized the need for manipulation of information supply. A prominent example is GOOGLE search where the need for conversions between file types is recognized (at least for search results that have certain types). It allows the searcher to select different file types and automatically performs *transformations* in the background. Another example is *Babelfish*[3] which allows for the possiblity of automatic translation of resources on the Web.

Observe that research Question 3 only specifies how resources from information supply can be manipulated. It does not take into account which manipulations (by means of a transformation) are, and which are not meaningfull; that is, it does not take value addition by transformations into account. To be able to study that question we must first be able

[3]See e.g., `http://babelfish.altavista.com/`.

to express what meaningful is in this context. We will use the term *quality* to refer to this question:

Q 4 *How can quality on the Web be measured?*

Even though this is not explicitly mentioned, Question 4 also refers to the issue of finding out which manipulations of resources from information supply are meaningful. The word quality can be considered to be ambiguous since it is used for so many different things in so many different fields. In our opinion, however, quality of artifacts is analogous to aptness of resources on the information market. This brings us back to the aptness issue. In answering Questions 1– 4 we have defined the scope of our research as well as done the ground work for the final question:

Q 5 *What is aptness-based search and how can it work in practice?*

This final question boils down to the definition and applicability of aptness-based search in practice. It requires a clear definition of what aptness is, and will show how it can be used in practice.

1.4.2 Ambition & approach

In the previous subsection we have presented the research questions that have guided the research presented in this dissertation. We will now shift the focus to our ambitions in order to explain what we do and what we do not consider to be part of the scope of our research.

To start with Question 1, we already indicated that our goal is to set the stage for our research by defining what we mean by the term information market and that we will adopt a modeling approach to answer this question. More accurately, our ambition is to define a formal and descriptive model of the information market. Since the core observation is that information exchange is similar to a transaction on a market, the notion of transaction will be central to this discussion. By formalizing aspects of these transactions we hope to be able to very precisely indicate these similarities. To illustrate the working of our models we will present extensive examples that describe real-world transactions. In other words, we will perform a *population check*. Note that it is not our ambition to build a tool that predicts if, or how, transactions will occur.

A similar line of reasoning holds for Questions 2 and 3. Again, our ambition is to define a formal and descriptive reference model. This time the domain under consideration is that of information supply and transformations that manipulate information supply. On top of the usual motivations for formal modeling we add the observation that these models are tightly interwoven: transformations affect (resources from) information supply. This implies that we must have a thorough understanding of information supply before we can effectively model what transformations are, what their effects are, or how to deal with the issue of determining which transformations are useful to searchers. Even more, our reference model will provide a stable and robust architecture despite the ongoing evolution

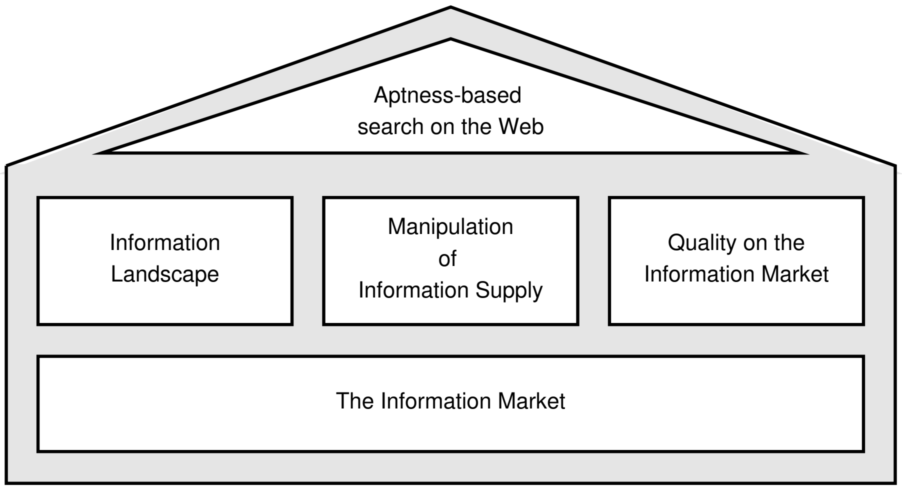

Aptness-based search on the Web
Information Landscape
Manipulation of Information Supply
Quality on the Information Market
The Information Market

Figure 1.4: Research setup

of the Web. A population check in the form of extensive examples will, again, show how our model can be used to clarify the state of affairs on the Web.

In case of the transformations we will go one step further, though. Our ambition is to not only build a generic theory that describes how information supply can be manipulated (meaningfully) by means of transformations, but to also illustrate that this framework can be used in practice by means of some small experiments.

Finally, when understanding what transactions are, what information supply is (i.e., which *assets* can be exchanged on the information market) and how supply can be meaningfully manipulated we can define a metric for value on the information market. In other words, in answering Question 4 we will define a metric for quality and show how it can be used as a measure for the aptness of resources on the Web. In doing so we will firstly present a formal model with which we can express what quality is. Secondly we will present a way of actually calculating quality (of resources on the Web) in such a way that resources can be compared with respect to their quality for a specific searcher in a situation. Again, our ambition is to come up with a descriptive formalism that can be validated by means of a population check rather than implementing an actual quality assessment system.

Last but not least, our ambition with respect to Question 5 is to show the applicability of our approach in practice. Ideally this would be done by building an aptness-based search system for the web that incorporates all our findings. By doing large scale experiments with users in which users compare several aspects of this system with other, pre-existing search systems one would hope to gain insight into the usefulness and applicability of aptness based search. This is, however, beyond the scope of this dissertation. Our ambition with respect to this question is more modest; we will present the architecture of a search system. Furthermore, we will present a toy-implementation as a proof of concept of our ideas.

Using the analogy of building a house, Questions 1 lays the foundation. The answers to Questions 2– 4 are the walls and the answer to Question 5 provides the roof. This analogy is illustrated in Figure 1.4. Note that the results to the individual questions are valuable by them selves. In this thesis, however, they mainly contribute to getting more insight in aptness-based retrieval.

1.5 Overview

We have chosen to let the structure of the chapters follow the research questions as much as possible. That is, Chapter 2 mainly deals with research Question 1, Chapter 3 mainly deals with Question 2, etcetera. The exception to this sequence is Chapter 5 in which we present some experiments pertaining to manipulation of information supply. Furthermore, an overview of the mathematical notation that was used in this dissertation is given in Appendix A. We will use the ORM notation (and more specifically, the PSM extension of ORM) throughout this dissertation. A brief introduction to this notation can be found in Appendix B.

CHAPTER 2 The information market

Outline The main goal of this chapter is to better understand the parallels between searching on the Web on the one hand, and market thinking on the other hand. As such it provides the background (or framework, if you wish) for the remainder of this dissertation. This chapter is based on [BGP$^+$05a, BGP$^+$05b, BGP$^+$05c].

2.1 Introduction

Our modern day western societies are dominated by information systems. Nevertheless, already in cultures and empires long gone, information systems played an important role. The Egyptians, the Greeks as well as the Romans, already used forms of (manual) information systems to administer trade and affairs of state. The industrial age resulted in an explosion of the amounts of information that needed to be handled. When the invention of the transistor gave us computers, these ‘information processing machines’ were gladly accepted as a means to automate parts of the information processing required. Hand in hand with the increase of the amounts of information that needed processing, the problem of information overload started to surface. As more and more data were amassed in information systems, it became harder and harder to find those bits of data that really mattered, i.e., that are really of information to someone. This has led to the introduction of the aforementioned field of information retrieval.

The development of the Internet provided our society with the opportunity to interconnect computers, leading to networked information systems. When the Internet matured, it gave birth to the World-Wide-Web (the Web). Resources that are available on the Web include: Web pages, newsgroups, mailing-list archives, networked databases, applications, business services, as well as indexing services. For users of the Web, these resources are at their disposal for doing business, searching for information, educational purposes, or relaxation. Since the Web literally spans the world, the number of accessible information resources is astronomical. This makes life rather difficult for the average user who shops around to discover information resources that fulfill his or her given information need. These developments have shifted the attention of information retrieval research away from 'stand alone' collections to information retrieval on the Web (e.g., [CHSS98, Des97]) as discussed in Section 1.2.2.

When the World-Wide-Web matured it gave birth to e-business (e.g., [TLKC99]). Given the abundance of information available via the Web, an important part of the commodities traded on the Internet are actually 'carriers' of information. In other words, we use the Web for many different things ranging from looking something up, staying in touch with friends, to making online reservations for hotels in far away countries. For reasons that are not always apparent we find some resources to be of higher value than others in any given situation. The concept of *value* is complex, is used in many different fields, and is central to our discussion in this chapter.

A second important concept is that of a *transaction* which can loosely be defined as an exchange of value. In case of search on the Web this is exemplified by the observation that one puts in effort in order to select / consume certain resources from the Web. Another view on these transactions is put to the fore by observing that people or organizations publish resources on the Web and that others consume them.

In the following sections we will first provide a brief overview of theory relevant to the definition of the information market. More specifically, we will present a formal model to illustrate our view on market thinking in Section 2.2. This section is mainly inspired by [TLKC99, Var96, KR94, Hol99]. In Section 2.3 we zoom in on the information market and further elaborate our formal theory. This section is mainly inspired by [SBS98, SS04, SV99].

2.2 Market thinking

2.2.1 Assets

In economic markets we observe that assets are being exchanged between players. These assets can vary from tangible things such as a book, a car, or a cup of coffee to complex services such as the massaging of a sore back, or fire insurance. Several definitions of what an asset is can be found in literature such as:

> Assets are goods that provide a flow of services over time. Assets can provide a flow of consumption services, like housing services, or can provide a flow of money that can be used to purchase consumption. Assets that provide a monetary flow are called financial assets.
>
> —*Taken from: [Var96]*

It is interesting to observe that, in this definition, a distinction is made between financial assets and 'other' assets. We do not make this distinction and define an asset to be:

Definition 2.2.1 (Asset) *Any thing that can be exchanged in a transaction.*

In our formal model, $\mathcal{AS}$ denotes the set of assets. Not that assets need not be physical entities such as a house, a book or a car; the assets that are actually traded are typically rights on these entities. As such, we adopt the point of view that two main classes of assets can be traded on a market. First of all, there is the *trade of ownership* which implies that the ownership of a physical entity (a book, land, bank notes) is transferred. This does, however, not directly imply that the entity itself is moved from one spot to another, which would be the execution of a service on this asset. That is precisely the reason why we argue that the ownership-right is transferred (as opposed to the asset itself). More simply put, trading an entity involves a change of ownership.

A second class of assets is that of *execution of services*. Services may be applied on/to/over entities, such as the painting of a house, treatment of an illness, etcetera. The right to execute such a service may be transferred from one actor to another. This class could be further split into two subclasses: *transformation of entities* and *reduction of uncertainty*. Examples of the former would be: the transportation of a chair from some warehouse to someone's living room, composition of a car from its components, the creation of an abstract from a book, etcetera. Examples of the latter would be: quality appraisal, credit card checking, etcetera.

2.2.2 Transactions and players

The definition of an asset calls for a clear-cut definition of a transaction. The notion of transaction is, perhaps, best known from the field of databases. For example, in [Dat03] a transaction is defined to be "a logical unit of work, typically involving several database operations" and in [Wat99] as "an event about which data are recorded and processed". The key point of the definitions of transactions as used by the database community is that they form a logical unit. We also adopt this approach and define a transaction as:

Definition 2.2.2 (Transaction) *A specific, identifiable exchange of assets between two or more players where each participant in the transaction pays something and receives something in return.*

In this definition the word 'player' refers to persons or organizations that may participate in a transaction. We let $\mathcal{PL}$ denote the set of players and $\mathcal{TA}$ denote the set of transactions. The following two examples of transactions are illustrated in Figure 2.1[1]. Figure 2.1a illustrates the situation where two players (p_1 and p_2) exchange two assets (a_1 and a_2). This occurs when, for example, John (p_1) buys a book (a_2) for €20 (a_1) from a bookstore (p_2). Figure 2.1b illustrates the case where three players are involved in a single transaction. This illustrates the case where John (p_1) pays the bookstore (p_3) a sum of €15 (a_1) to receive a book (a_3) directly from a publisher (p_2) after being paid (a_2) by the bookstore. The asset(s) that a player receives in a transaction is defined to be his *benefit* and the

[1]Even though the notation in Figure 2.1 resembles that of UML sequence diagrams, the 'swimming lanes' do not imply a time-aspect. They only exemplify exchanges, not the order in which they are executed.

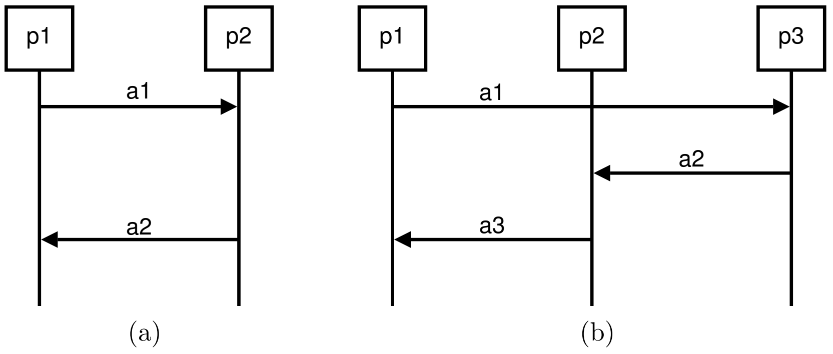


Figure 2.1: Transactions between two or three players

asset(s) that he pays are defined to be his *cost:*

Definition 2.2.3 (Benefit) *The assets that a player receives in a transaction.*

Definition 2.2.4 (Cost) *The assets that a player pays in a transaction.*

Note that these definitions are at the level of transactions; they do not include any valuations. These are introduced later. Observe that, for the time being, we have refrained from introducing the terms 'buyer' and 'seller'. In our view, the notion of selling and buying can only be defined relative to a specific asset that is involved in a transaction. To illustrate this consider the following example:

Example 2.2.1 *When someone visits a flower shop to obtain a bouquet of roses, they may do so by handing over €10 to the person behind the counter. In this situation, we argue that the person behind the counter sells the flowers to the person in exchange for €10. However, one could also state that the person visiting the flower shop sells a note of €10 in exchange for a bouquet of roses.*

In our formal model we distinguish between two views on transactions:

1. A player exchanges one asset for another.

2. Assets are transferred from one player to another.

Transactor view

The notion of a *transactor* reflects the first view. A transactor $a_1\,[p]\,a_2$ denotes the fact that player p exchanges asset a_2 for asset a_1. As such, $\_\,[\_]\,\_ \subseteq \mathcal{AS} \times \mathcal{PL} \times \mathcal{AS}$. This allows us to model a transaction as a set of transactors. For example, in Figure 2.1a we have modeled the transaction $T = \{t_1, t_2\}$ where t_1 is the transactor $a_2\,[p_1]\,a_1$ and t_2 is the transactor $a_1\,[p_2]\,a_2$.

For the remainder of this discussion we will use the term *exchange object* to denote the asset that a player receives and the term *exchanged object* to denote the asset that a player pays in a transaction. Recall that transactions form a logical unit (Definition 2.2.2). More specifically, we consider them to be atomic in the sense that an asset can be exchanged only once in a single transaction:

Axiom 1 (Unique exchange asset)

$$a_1\,[p_1]\,b \in T \;\;\wedge\;\; a_2\,[p_2]\,b \in T \;\;\Rightarrow\;\; p_1 = p_2 \;\;\wedge\;\; a_1 = a_2$$

Axiom 2 (Unique exchanged asset)

$$a\,[p_1]\,b_1 \in T \;\;\wedge\;\; a\,[p_2]\,b_2 \in T \;\;\Rightarrow\;\; p_1 = p_2 \;\;\wedge\;\; b_1 = b_2$$

As an additional requirement we forbid that a player exchanges an asset for itself:

Axiom 3 (Real exchange) $a\,[p]\,b \;\;\Rightarrow\;\; a \neq b$

So far we have described how assets and players behave in transactions and have used (single) transactors. We now focus on properties of transactions. To this end we must firstly introduce an axiom of structural induction:

Axiom 4 (Structural induction)
If F is a property of assets such that

- $\exists_a\,[F(a)]$
- $\forall_{a,p,b}\,[F(a) \wedge a\,[p]\,b \;\;\Rightarrow\;\; F(b)]$

then it follows that $\forall_a\,[F(a)]$

A useful anology for studying transactions using transactors is a labelled directed graph which is spanned by $\_\,[\_]\,\_$ where the nodes are assets and the arcs are players. We will introduce the following re-write rules:

$$\begin{array}{rcl} a \in T & \vdash & a \overset{T}{\mapsto} a \\ a \overset{T}{\mapsto} b \;\;\wedge\;\; b\,[p]\,c & \vdash & a \overset{T}{\mapsto} c \end{array}$$

where $a \in T$ has the intended meaning of asset a being involved in transaction T either as exchange asset or as exchanged asset (i.e., either $a\,[p]\,b$ or $b\,[p]\,a$ for some p and b). We can now use the structural induction scheme to prove that by following the transactors that span the graph we will eventually return to the point of departure. In other words, this means that we have to 'close the circle' in the sense that if one asset is exchanged then eventually everyone who pays an asset will receive one and every asset that plays the role of exchanged object will also play the role of exchange object. This is illustrated in Figure 2.2 which depicts a transaction as a graph where the nodes are assets and the arcs are players. We can prove this as follows:

Lemma 1 $a\,[p]\,b \in T \;\;\Rightarrow\;\; b \overset{T}{\mapsto} a$

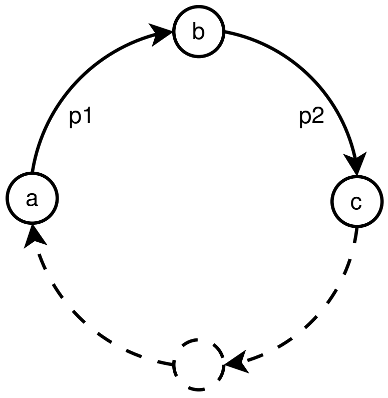


Figure 2.2: Finite transactions

Proof:

Define for some $x \in T$ the property $F_x(a) = x \overset{T}{\mapsto} a$.

1. Let $F_x(b)$ such that $b = x$. As $x \in T$ we have $x \overset{T}{\mapsto} x$. Therefore $F_x(x)$ and we can conclude that $\exists_a [F_x(a)]$.
2. Assume $F_x(a) \;\wedge\; a\,[p]\,b$ for some a. We must prove that $F_x(b)$. Since $F_x(a)$ we know that $x \overset{T}{\mapsto} a$. Therefore we have $x \overset{T}{\mapsto} a \;\wedge\; a\,[p]\,b$ which leads to $x \overset{T}{\mapsto} b$. This proves $F_x(b)$.

Using Axiom 4 we have now proven that an asset which is received by a player must also be payed by a player in the same transaction.

□

It is now relatively straightforward to prove that every asset that is received by someone in a transaction must also be payed by someone in that same transaction:

Lemma 2 $c\,[q]\,a \in T \;\Rightarrow\; \exists_{b,p}\,[b\,[p]\,c]$

Proof:

Let $c\,[q]\,a \in T$. Therefore $a \overset{T}{\mapsto} c$ (Lemma 1). Also, we know that $c \neq a$ (Axiom 3). Given the re-write rule $a \overset{T}{\mapsto} b \;\wedge\; b\,[p]\,c \;\;\vdash\;\; a \overset{T}{\mapsto} c$ we know that $a \overset{T}{\mapsto} b \;\wedge\; b\,[p]\,c$ must hold. We can now conclude that $\exists_{b,p}\,[b\,[p]\,c]$.

□

Transactand view

The notion of a *transactand* reflects the second view on transactions. In this case, a transaction is seen as a flow of an asset from one player to another. A transactand $p_1\,[a]\,p_2$ denotes the fact that asset a is transferred from player p_1 to player p_2. More formally, $\_\,[\_]\,\_ \subseteq \mathcal{PL} \times \mathcal{AS} \times \mathcal{PL}$. Analogous to the transactor case, a transaction can also be seen as a set of transactands. Using transactors, the situation in Figure 2.1a is modeled as the transaction $\widetilde{T} = \{t_1, t_2\}$ where t_1 denotes the transactand $p_1\,[a_1]\,p_2$ and t_2 denotes the transactand $p_2\,[a_2]\,p_1$. We use the $\widetilde{T}$ notation, rather than, T, to denote that we presently use the transactand view. The formal relation between transactors and transactands is given by:

$$p_1\,[a_2]\,p_2 \in \widetilde{T} \;\triangleq\; \exists_{a_1,a_3}\,[a_1\,[p_1]\,a_2 \in T \;\wedge\; a_2\,[p_2]\,a_3 \in T]$$

In the remainder of this dissertation we will either use transactors or transactands, depending on what we are trying to express. Summarizing the above: transactions can either be seen as a set of transactors or as a set of transactands.

We now shift our attention to the players that are involved in a transaction. From the definition of transaction it follows that at least two players must participate in a transaction. This can be expressed as follows. The participant in a transactor t is given by $\mathsf{Participant}(a_1\,[p]\,a_2) \;\triangleq\; p$. Similarly, the set of participants of a transaction $\widetilde{T}$ is defined as:

$$\mathsf{Participant}(\widetilde{T}) \;\triangleq\; \bigcup_{t \in T} \mathsf{Participant}(t)$$

Recall that we already observed that the same player can participate in a transaction once. We can now easily prove that:

Corollary 1 $t_1, t_2 \in \widetilde{T} \;\wedge\; \mathsf{Participant}(t_1) = \mathsf{Participant}(t_2) \;\Rightarrow\; t_1 = t_2$

We end this discussion with the roles that players can have in a transaction. Earlier we already stated that these roles only make sense relative to a specific asset in a transaction. Let $\widetilde{T} = \{p_1\,[a_1]\,p_2, p_2\,[a_2]\,p_1\}$ be a transaction. In this transaction, player p_1 is the seller of a_1 and player p_2 the buyer of a_1. To express this formally we introduce $\mathsf{Buyer}, \mathsf{Seller} : \mathcal{TA} \times \mathcal{AS} \to \mathcal{PL}$ such that:

$$p_1\,[a]\,p_2 \in \widetilde{T} \;\Rightarrow\; \mathsf{Seller}(\widetilde{T}, a) = p_1 \text{ and } \mathsf{Buyer}(\widetilde{T}, a) = p_2$$

It is now straightforward to prove that a player can only play the buyer and seller role in a transaction only once:

Corollary 2 $\mathsf{Buyer}(\widetilde{T}, a) \neq \mathsf{Seller}(\widetilde{T}, a)$

Using these definitions for buyer and seller roles we can model a wide variety of transactions. There is, however, a third role that can be played in a transaction. This class of players is called *brokers* and is defined as follows:

Definition 2.2.5 (Broker) *A broker is a player that participates in a transaction but does not alter the asset that is exchanged and is value adding for the other players involved in the transaction.*

We have not yet formally defined what we mean by value (which is the topic of Section 2.2.3). Relying on the reader's intuition regarding the meaning of the term value, we end this subsection with a typical example of a transaction where a broker is involved.

		Extrinsic	Intrinsic
Self-oriented	Active	*Efficiency*	*Play*
	Reactive	*Excellence*	*Aesthetics*
Other-oriented	Active	*Status*	*Ethics*
	Reactive	*Esteem*	*Spirituality*

Figure 2.3: Typology of consumer value

Example 2.2.2 *Consider the market for antiquities such as paintings. Consumers can either buy a painting directly, or via an intermediary (broker) at an auction. If the transaction takes place via a broker then this broker must be value adding by definition:*

- From the consumer point of view*: Finding a specific antiquity can be very hard if no intermediary is involved. For example, how would a person in the Netherlands ever find out that a person in the USA is selling a painting by Rembrandt? Furthermore, the fact that a well-established broker (i.e., an auctioneering firm such as Sotheby's) is selling the piece will give the consumer more confidence in its genuineness. He may even be willing to pay an additional fee in return for this added value.*

- From the supplier point of view*: The supplier (i.e., the person selling the antiquity) knows that there is a better chance of selling via a broker since all consumers will go there. There is also a better chance of receiving a higher price. Also, the broker will take care of shipping the item, insurance of the item during transportation, and so on.*

Note that the broker does not alter *the asset: an auctioneer will not re-paint a* Van Gogh *painting, he merely facilitates the transaction.*

2.2.3 Value

The notion of value is abstract and rather difficult to capture in a definition. It is used in many different fields such as marketing, computer science, mathematics and even in the context of personal and cultural values. An extensive discussion of the value notion in the context of marketing is given in [Hol99]. As the discussion also covers aspects of other fields we include an overview here after which we present our own view on the concept of value in markets.

It is interesting to observe that in [Hol99] it is pointed out that "the theory of value is a topic neglected not only by marketeers but even by axiologists[2] themselves". After carefully studying the available literature on axiology and marketing, the author has derived a framework for classifying different kinds of value. This framework is summarized in Figure 2.3. The framework is built along the following three dimensions:

[2]Axiology: the study of values and value judgments.

1. *Extrinsic* value pertains to a rather functional or utilarian view on value, whereas *intrinsic* value occurs when an artifact or consumption is appreciated as an end in itself.

2. *Self-oriented* value occurs when consumption is done for one's own sake; i.e., is hedonistic. On the other hand, *other-oriented* value looks beyond the self and occurs when consumption is intended to please another.

3. Value is *active* when consumption involves things done by a consumer to the good / service that is consumed, whereas value is *reactive* when it results from things done by a product / service to the consumer.

The framework is mainly used and validated in the context of consumers and marketing. Even though it does not provide us with a method to quantify value, interesting lessons can be learned from it still. For our purposes, the most important observation is that the notion of value can be seen from many different points of view. In our own model for value, which we will present shortly, we propose to express value in an abstract domain which is mainly self-orriented in nature.

The notion of value

The dictionary definition[3] for value is "relative worth, utility, or importance". This definition stresses the fact that value is personal. Something that is valuable to one person may be 'worthless' to another. In an economic market the notion of value has the following two important characteristics:

1. Assets have an intrinsic value that may differ from person to person. In other words, players value assets.

2. The value of an asset can be expressed in its comparison to other assets.

The latter aspect refers to transactions, where assets are exchanged between players. It stresses the fact here is no domain in which value can be expressed. Rather, values can be compared such that the $>$ is a partial order relation. In our formal model, however, we do assume that there is an abstract value domain in which we can express value, which leads to the following definition:

Definition 2.2.6 (Value of an asset) *The value of an asset to a player is defined as the increment/decrement in the satisfaction level of this player when this asset is exchanged in a transaction.*

The value of an asset is,thus, highly personal and can only be expressed in an abstract value domain. The personal and abstract nature is captured by its formal definition: $\mathsf{Val} : \mathcal{PL} \times \mathcal{AS} \rightarrow \mathcal{VD}$, where $\mathcal{VD}$ is the abstract value domain. The only requirement with

[3]The Consice Oxford English Dictionary.

regard to this value domain is that $>$ must be a partial order relation which allows us to compare the values of assets for a certain player. For example, we want to be able to write $\mathsf{Val}(p, a_1) > \mathsf{Val}(p, a_2)$ in order to indicate that player p finds the value of asset a_1 to be more valuable than asset a_2.

The personal nature of this value function is stressed even more by introducing states of players. A state is defined as the present satisfaction of a player with respect to his/her goals[4]. Let $\mathcal{ST}$ be the set of states. The function $\mathsf{Id} : \mathcal{ST} \rightarrow \mathcal{PL}$ identifies which state belongs to which player. In conformance to this view we also define a second view on the Val function, $\mathsf{Val} : \mathcal{ST} \times \mathcal{AS} \rightarrow \mathcal{VD}$ such that:

$$\mathsf{Val}(s, a) \triangleq \mathsf{Val}(\mathsf{Id}(s), a)$$

Using states, rather than players, as the basis for valuation results in the observation that players in the exact same state will assign the same value to a specific asset.

Choice

This notion of value is the basis for making choices. If a player has a choice between several options (i.e., buying bundles of assets) he will choose the option with the highest value to him. The fact that asset a_1 has a strictly higher value than a_2 is a transitive, and irreflexive relation and is denoted as $a_1 \succ_p a_2$ which is defined as:

$$a_1 \succ_p a_2 \triangleq \mathsf{Val}(p, a_1) > \mathsf{Val}(p, a_2)$$

Similarly, indifference between two assets a_1 and a_2 (i.e., two assets having the same value) is denoted $a_1 \sim_p a_2$. Weak preference is then defined as $a_1 \succeq_p a_2 \triangleq a_1 \succ_p a_2 \vee a_1 \sim_p a_2$. For an overview of preference see e.g., [KR94, Var96].

More elaborate schemes for preference exist as well. For example, [Sug03] describes a reference-dependent approach to utility and preference. The core of the described approach is that preference is dependent on a current position. Strict preference and indifference are defined similarly. Also, the preference relation is defined to be complete and transitive. Even more:

> A decision problem can be described by a reference act and an opportunity set of acts (the set of options from which the agent must choose), of which the reference act is one element. The agent chooses either to stay at the status quo or to move to one of the other options.
>
> —*Taken from: [Sug03]*

In other words, for a decision problem the preference (strict preference, weak preference or indifference) is dependent on the current position.

The value-notion is the basis for decision making of players (cost / benefit analysis). We presume that players of the market behave in a goal-driven manner. That is, they want to satisfy their goals by engaging in transactions. These goals can be either explicit or implicit based on such things as political situation, mental state, etcetera.

[4]We get back to the discussion on goals of players in Chapter 6.

Cost and Benefit

If T is a transaction and $s \in \mathcal{ST}$ a state then $s \ltimes T$ is the state which results if participant $\mathsf{Id}(s)$ participates in T. To make the discussion of state-changes in transactions easier we introduce the abbreviations:

$$\begin{aligned} a_1\,[s]\,a_2 &\triangleq a_1\,[\mathsf{Id}(s)]\,a_2 \\ s_1\,[a]\,s_2 &\triangleq \mathsf{Id}(s_1)\,[a]\,\mathsf{Id}(s_2) \end{aligned}$$

to denote the fact that an actual transaction will take place between *participants who hold a specific state*. We require the resulting state after a transaction to belong to the original participant.

Axiom 5 (State-change in a transaction) $\mathsf{Id}(s) = \mathsf{Id}(s \ltimes T)$

Players will only participate in a transaction if they (expect to) gain something from it. This can easily be expressed using the trasactand view. If $a_2\,[s]\,a_2 \in \widetilde{T}$ then player $\mathsf{Id}(s)$ values a_2 higher than a_2

Axiom 6 (Rational behavior) $a_1\,[s]\,a_2 \;\Rightarrow\; \mathsf{Val}(s, a_1) > \mathsf{Val}(s, a_2)$

A more refined view uses the notions of *cost* and *benefit*, which are introduced previously. We can now formalize these as follows. Let $a_1\,[s]\,a_2 \in \widetilde{T}$, then:

$$\begin{aligned} \mathsf{Cost}(s, T) &\triangleq a_2 \\ \mathsf{Benefit}(s, T) &\triangleq a_2 \end{aligned}$$

Given the fact that we know that players only participate in a transaction if they expect to gain from it (Axiom 6) it is easy to prove that:

Corollary 3 $\mathsf{Benefit}(s, T) > \mathsf{Cost}(s, T)$

We will return to this discussion in Chapter 6 where we present a model for quality. In that model we will also take the *goals* of players into account, and study how (consumption of) assets have an impact on the satisfaction of these goals.

Last but not least, we need to model the value adding nature of brokers. We will discuss this from the *transactor* point of view. This value adding nature was already illustrated in example 2.2.2 where we discussed brokers in the context of selling/buying antiquities. The case where a consumer (p_1) buys a painting (a_1) directly from another person (p_2) for a certain amount of money (a_2) can easily be modeled as a transaction $T = \{p_1\,[a_2]\,p_2, p_2\,[a_1]\,a_2\}$. However, the case where a broker is involved is not as easy to model with the theory introduced so far. Note that:

- Brokers do not alter the asset to be exchanged.
- Brokers would not exist if they wouldn't be able to 'get something out of brokering' (See axiom 6).

- Even if transactions via a broker 'cost more' to the participants involved in the transaction, it must still be value-adding to all these participants. Otherwise the transaction would not be executed.

It seems natural to model this situation such that brokers are not part of the actual transaction because they merely facilitate it. This is, however, not very elegant. We consider brokers to be normal, regular players. The following example illustrates a transaction where a broker is involved:

Example 2.2.3 *Suppose p_1 has a* Van Gogh *(a_1) for sale. To support him in selling it for a proper price (a_2) he asks an auctioneer (p_2) to assist him for a fee (a_3). The execution of this service is denoted a_4. When person p_3 buys the paining for a_5 via this broker then two transactions are completed:*

- $T_1 = \{a_2\,[p_1]\,a_1, a_1\,[p_3]\,a_5, a_5\,[p_2]\,a_2\}$
- $T_2 = \{a_4\,[p_1]\,a_3, a_3\,[p_2]\,a_4\}$

Note that the broker does not alter the assets a_1 and a_2. The broker merely facilitates the transaction. However, the participants involved do perceive them to be more valuable.

2.3 The information market

In the previous section we presented our view on market-thinking in general, and clearly positioned our view with regard to economic markets. In this section we will apply our findings to the more specific case of the information market which we define as:

Definition 2.3.1 (Information market) *The information market is the market where* resources *are exchanged between* searchers *and* publishers*, possibly by means of* brokers*.*

In this definition, resources can be thought of as the 'entities' on the Web that make up information supply. We will present a more formal definition in Chapter 3. Recall that a transaction is a specific, identifiable exchange. The publication of a resource on the Web can be considered as a transaction between the publisher of the webserver. Similarly, downloading a resource can be considered a transaction between the webserver and the searcher. On a different level of abstraction we assert that a transaction between the publisher and searcher is completed once a copy of the resource is downloaded by the searcher. From this we can infer that transactions have a time-aspect since the moment of publishing a resource and actually consuming (downloading) it may be far apart in time. Even more, on 'normal' markets this is not the case since transactions are instanteneous in that situation. Also, transactions on the information market are one-to-many since many searchers may download (consume) the same resource. This is illustrated in Figure 2.4: the publisher publishes (the original) resources (denoted by the letter 'o') after which many searchers download copies (denoted by the letter 'c') of it. This figure illustrates another

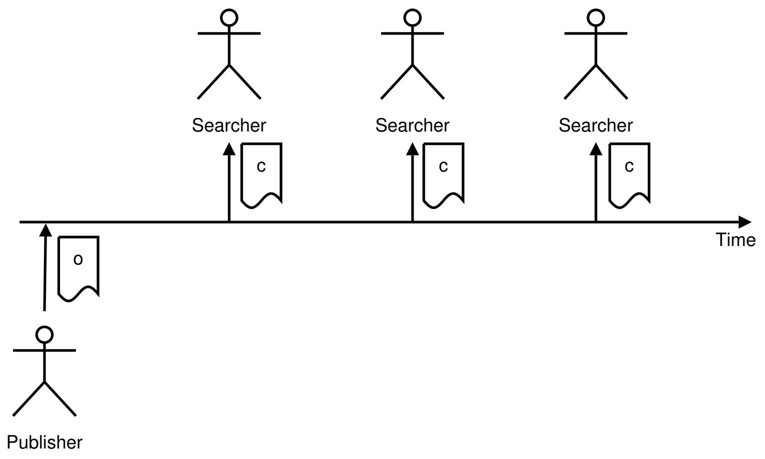


Figure 2.4: Time aspects of transactions on the information market

distinguishing feature of transactions on the information market. Recall that there are two kinds of rights on assets: ownership and execution of services. On the information market, the ownership right of a resource is not transferred as such; searchers receive a *copy* of the original resource. As such, downloading a (copy of) a resource is the execution of a service, not the transfer of ownership rights.

The value of a resource is difficult to measure. As a consequence, it is hard to put a price on them. Also, it is hard for consumers to assess whether they wish to purchase/consume the resource or not: the only way to assess the value is by consuming it. Similarly, it is often unclear why publishers actually publish resources on the Web. Surely enough, for companies a transaction may increase popularity, or people may even pay to see certain information. Often, however, this is not the case. Consider, for example, the *Wikipedia*[5] case. Wikipedia is a free, online encyclopedia. What do authors participating in this project gain? Similarly one may wonder how valuable the information is that is published on this site. Would the information be more valuable if it were not free of charge?

In [Als99] it is stressed that resources typically do not behave like assets and that information quantity can not be used directly to decide which resource is better. The authors observe that two approaches to information should be treated as a dual concept. On the one hand, information can be seen as a reduction in uncertainty (i.e., the Bayesian approach). On the other hand, it can be seen as a description of a state transition (i.e., a Turing Machine approach). This observation is the basis for a framework to assess the value of resources. Another approach for pricing and valuing information can be found in [SV99].

Several other approaches relating to the value of resources have been proposed in the literature over the last few years. For example, the work of Grice (see e.g., [Cru00, p. 355–358]) focuses on conversations but can also be applied to analyze the value of resources. In

[5]`http://www.wikipedia.org/`

this respect Gryce proposes four maxims: the maxim of quantity, the maxim of quality, the maxim of relevance and the maxim of manner. Results from the field of multi-dimensional data modeling can also be used to model the different characterizations of the value of a resource. In [PJ98] many different dimensional types characterize a fact type; e.g., the fact type *Patient* can be characterized by the dimensional types *Diagnosis*, *Residence*, *Social Security Number*, *Name*, etcetera. In [VW99] the *double coincidence of wants* is described as:

> Double coincidence of wants relates to the fact that both traders involved in an exchange transaction without a recognizable currency should find the other agent's offering useful and desirable.

The authors then observe that this is the core source of inefficiency in resource-based transactions; instead of simply acquiring a desired resource a player has to locate another player that not only offers the desired resource but is also willing to exchange it for the proposed payment. It is argued that the main function of brokers is to eliminate friction in the market by decreasing the search efforts. Brokers are considered to be value adding because most users do not have the expertise to properly assess the quality of resources.

In summary: it is hard, to say the least, to pick a value domain for the information market. We will adopt a multi-dimensional view on this domain:

Information : the information that may be provided by a resource. This refers to the actual 'content' of a resource.

Structure : concerned with the form (report, audio, summary, outline) and format (*Pdf*, *Xml*,*Doc*) of a resource.

Emotion : dealing with the emotional effect (pretty/ugly/inspiring) that a resource may have when it is consumed.

These value domains closely correspond to three aspects of *architecture* as introduced by Vitruvius, a Roman writer, architect and engineer, active in the 1st century BC. These aspects were called *utilitas* which corresponds to our informational domain, *firmitas* which corresponds to our structural domain and *venustas* which corresponds to our emotional domain (see e.g., [Rij04, Vit99] for details). This value taxonomy can be used to describe both costs and benefits of transactions for both searchers and publishers. To see how this would work out for *costs* consider the following:

Searcher – The costs of a transaction for a searcher could very well include the costs of actually obtaining the resource such as search costs (information dimension), the amount of disk space needed to store the resources (structure dimension), or the costs associated with actually conceiving the resource; the cognitive load (see e.g., [TG03]) associated with interpreting and understanding the resource (emotional dimension).

Publisher – The costs of a transaction for a publisher could very well include such things as the time and effort associated with creating the resource (information dimension), the costs associated with storing the resource such as disk space, as well as required computing power in creating the resource (structural dimension) or the energy needed to create the contents of the resource. This may also be referred to as cognitive load [BDM00] (emotional dimension).

Surely, this crude classification of types of value that play a role on the information market gives us some insight in how the market works, in what the concept of value means, etcetera. It is, however, not specific enough to do any real computation with them; for example, to compute the cost/benefit of a resource to an actor. In the upcoming subsections we will outline more specific models for the information and emotion dimensions of value. A model for structural aspects is presented in Chapter 3. As an intermezzo we present a possible model for information value based on *infon theory* in Section 2.4. This discussion is based mainly on [BH94, BGP+05a].

2.4 Information value

In this section we focus mainly on the informational domain. Our goal is to gain insight in the strategies of information consumers on the information market as well as to show that infon algebras can be used to model the intentional description of the *information gap of searchers* and the *characterization of resources*. As such, infon algebras can aid us in determining the information value of resources on the Web. In this section we aim to provide a deeper understanding of the informational value domain and its application in the information market.

What information exactly is has been studied intensively before, see for example [Dev90, BP96]. Different authors from different fields have provided diverse theories of the nature of information. The notion of information plays an important role in fields such as information retrieval [Rij75, SM83], cognitive science [SWC+95, Oos03] database systems [Dat03, Cod70], data modeling [Che76, Nij89, Hal01, HW93, HPW93] etcetera.

We take a modest approach to information theory, and only assume information to consist of *information particles* called *infons* as well as a *specialization operator*. Infon theory has been suggested by Barwise [Bar89, Dev90], and applied to the field of information retrieval by [RL96]. Infons can be thought of as imaginary objects in the sense that they cannot be denoted or named explicitly.

An infon algebra is referred to as $\mathcal{IF}$. Formally, it is a structure

$$\mathcal{IF} = \langle \mathcal{I}, \rightarrow, \bot, \top \rangle$$

where $\mathcal{I}$ is the set of all infons. $\bot$ and $\top$ are special infons, corresponding to the least and the most meaningful information particles respectively. Furthermore, $\rightarrow$ is a relation to compare the information content of infons; it denotes the *specialization* relation.

The specialization operator

The main property of infons is that they can be compared with respect to their informational content. We use the generic term *specialization* for such a comparison. If $i \rightarrow j$ then we say that i is a specialization of j or, j a generalization of i. The specialization of infons can be interpreted as either information containment or precognition:

Information containment expresses the fact that some information particles contain more information than others. For example, the statement (referred to as i_1) *grass tends to be green, but varies between brown and green* contains more information than the statement (referred to as j_1) *grass is usually green.* Statement j_1 is obviously less informative than i_1. In this case the information of i_1 contains the information of j_1, or: *grass tends to be green, but varies between brown and green* contains *grass is usually green.* This is denoted as $i_1 \rightarrow j_1$. The specialization relation is interpreted as an information containment relation.

Precognition expresses the fact that, in order to understand an information particle, another information particle is required. Consider the following example: it is impossible to understand *Pythagoras' Theorem* (referred to as infon i_2) without understanding the concept of *triangle* (referred to as infon j_2). In other words, infon j_2 is a prerequisite for infon i_2. This is expressed as $i_2 \rightarrow j_2$. The fact that *Pythagoras' Theorem* is a specialization of *triangle* is interpreted as a precognition relation.

From a logical point of view, we would express this as: infon i_1 involves infon j_1, or as: infon j_1 is a consequence of i_1. So from having the knowledge *grass tends to be green, but varies between brown and green* we can conclude the knowledge *grass is usually green* as a consequence. This is denoted as $i_1 \rightarrow j_1$. Our second example may be formulated as: if a person has knowledge of *Pythagoras' Theorem* then we can conclude this person has knowledge of *triangle.* This is denoted as: $i_2 \rightarrow j_2$. As an analogy, consider the boolean proposition $p \Rightarrow q$. Then it is said that p is a sufficient condition for q, or that q is a necessary condition for p. Using the analogy of the triangles, it seems reasonable to view j_2 as information that is prerequisite to grasp the infon i_2: knowledge of triangles is prerequisite to knowledge of Pythagoras' Theorem.

Properties of the specialization operator

The properties of an infon algebra are described as properties of the relation $\rightarrow$. This is assumed to be a partial order on infons.

Axiom 7 (reflexivity) $i \rightarrow i$

Axiom 8 (anti-symmetry) $i \rightarrow j \wedge j \rightarrow i \;\Rightarrow\; i = j$

Axiom 9 (transitivity) $i \rightarrow j \wedge j \rightarrow k \;\Rightarrow\; i \rightarrow k$

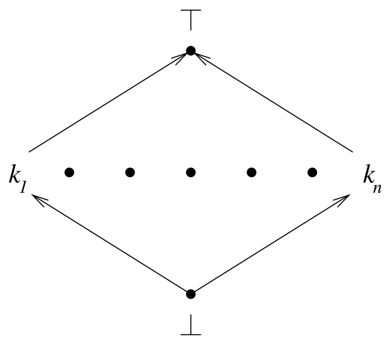


Figure 2.5: The lattice for keywords

Two special infons are assumed, a most specific infon ($\perp$) and a least specific (most general) one ($\top$). They are characterized by:

Axiom 10 (top element) $i \rightarrow \top$

Axiom 11 (bottom element) $\perp \rightarrow i$

These properties state that the infon $\perp$ is a specialization of every infon whereas every infon is a specialization of $\top$. As such, $\top$ can be interpreted as a *world view.* An example would be the view that the world consists of keywords and collocation of keywords. Another example would be the view that the world consists of *concepts* (in which case the lattice-structure would be a *concept lattice*). Similarly, $\perp$ can be interpreted as the infon that is so specific that it is no longer meaningful. Example 2.4.1 illustrates how keywords can be used to operationalize an infon algebra in practice.

Example 2.4.1 (Flat keyword lattice) *The most simple indexing mechanism for documents is to use a set of keywords. Each keyword represents some semantical unit. We extend this set with two special 'keywords': $\perp$ and $\top$. In its most simple form, all keywords are assumed to be independent. As a consequence, if $i \rightarrow j$ then either $i = \perp$ or $j = \top$. The resulting structure is called the* flat keyword lattice. *Figure 2.5 illustrates such a structure.*

The information value of resources on the Web can be modeled by using the containment relation: the resource is seen as a big infon which can be decomposed using the specialization operator. Similarly, the knowledge / information gap of searchers can be expressed as an infon. Still, we're faced with the problem of "implementing" infons as they are merely a conceptual construct. Concept lattices [Wil82] or (power)index expressions [Bru90] seem to be a logical choice.

One of the main properties of the described infon algebra is that it forms a lattice. We propose to use index expressions to construct such a lattice structure which is commonly called a power index expression (See e.g., [Bru90, BW92b]). Index expressions have the

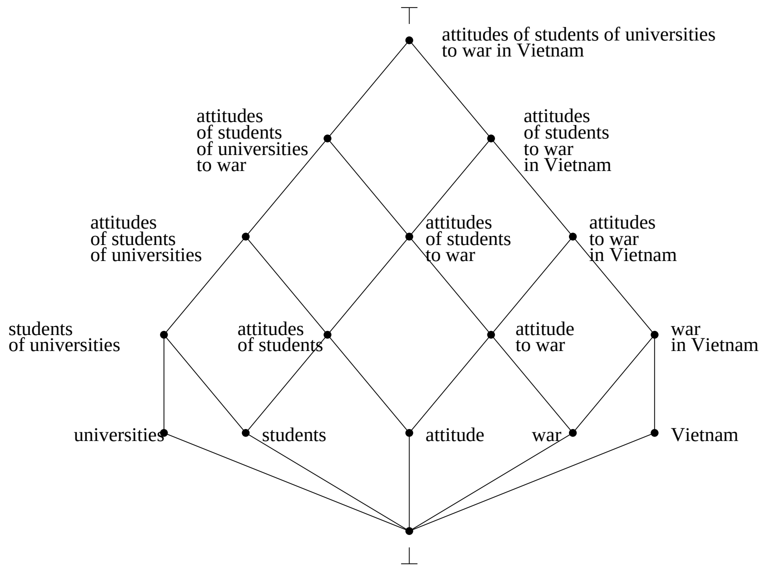

⊤
attitudes of students of universities to war in Vietnam
attitudes of students of universities to war
attitudes of students to war in Vietnam
attitudes of students of universities
attitudes of students to war
attitudes to war in Vietnam
students of universities
attitudes of students
attitude to war
war in Vietnam
universities
students
attitude
war
Vietnam
⊥

Figure 2.6: Example of a power index expression

following syntax:

$$\begin{array}{lcl} \mathsf{IdxExpr} & \rightarrow & \mathsf{Term}\,\{\mathsf{Connector}\ \ \mathsf{IdxExpr}\}^* \\ \mathsf{Term} & \rightarrow & \mathsf{String} \\ \mathsf{Connector} & \rightarrow & \mathsf{String} \end{array}$$

Brackets can be used to disambiguate base index expressions. Furthermore, $\cdot$ denotes the empty connector. An example of such a base index expression is attitudes to (courses of students) in universities. Another example is the expression attitudes of (students of universities) to (war in Vietnam). The lattice structure called power index expression is the set of all index subexpressions including the empty index expression denoted (in conformance to our infon algebra) $\perp$. The power index expression of the last example is shown in Figure 2.6. More details on the construction of index expressions is provided in [BW90b, Bru90].

2.5 Conclusion

The main goal of this chapter was to get a grip on the parallels between search on the Web on the one hand and market thinking on the other. To this end we started the chapter by presenting a model for economic markets. The core concepts in this model are transactions and value. We successively extended this model for the information market. This resulted, among other things, in the observation that the notion of value on the information market is highly complex. This seems to contrast with the traditional view from the field of information retrieval in which topical relevance seems to be the sole basis for valuing resources. In our model for value we include these informational aspects as well as structural and emotional aspects. Two other observations that result from our analysis is the fact that transformations on the information market have a time aspect and are one-to-many. Last but not least we observed that brokers (usually search engines) play an important role on the information market since they facilitate almost all transactions.

CHAPTER 3 The information landscape

Outline In this chapter we will zoom in on the information landscape. That is, we set out to present a model for the resources on the Web, their types, relationships, etcetera. The material presented in this chapter is based on [GPB04, GPBW05].

3.1 Introduction

As we already observed, there have been several important developments in the area of modeling, characterizing and understanding the true nature of the Web. The most well-known efforts in this respect are probably RDF, RDFS, and OWL (see e.g., Section 1.2.6).

Despite the (increasing) popularity of these languages we will adopt a different approach. That is, we will *not* present our model as an OWL vocabulary, nor will we explicitly describe how resources on the Web should be annotated using RDF. The main motivation for this is that, in our opinion, these languages are too implementation oriented and our present goal is merely to gain insight in what the information landscape looks like. Instead we will use the conceptual modeling technique ORM to find the essential concepts involved and their relations. This will also enable us to use the associated conceptual language LISA-D to describe the rules that govern the behavior of these concepts[1]. The conceptual language provides the opportunity to describe properties of the underlying domain in terms of the domain language, and yet has the full power of logical languages as the first order predicate calculus. The concpets and their relations form the signature of a logical structure. The signature of our formalism is shown in Figure 3.1. For the time being this figure is intended to give the reader a frame of reference while we will introduce the model one step at a time. We will later extend this. We do acknowledge that the above languages could be (incredibly) useful when actually implementing real systems. We will return to this discussion at the end of this chapter.

[1]See Appendix B for an overview of ORM/PSM.

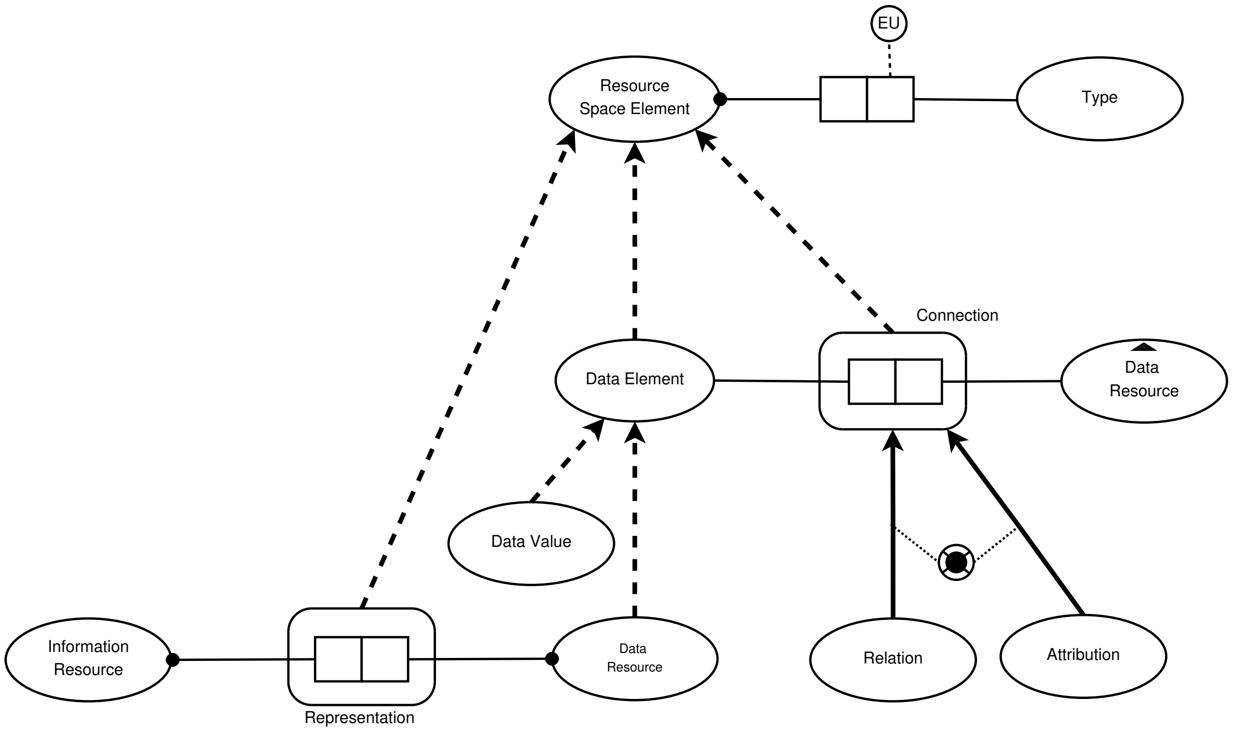


Figure 3.1: Signature of the model for resource space

In our model we make an explicit distincion between *resource space* (defined as the expanse of possible resources on the information market) and *resource base* (representing the set of resources that are available at a specific point in time).

3.2 Resource space

An important dichotomy is that of data versus information. This has been extensively studied in literature. A good overview of the current discussion on this topic is presented in [BCM04] which mainly uses ideas presented in [Ack89]:

> The content of the human mind can be classified into five categories:
>
> 1. Data: symbols
> 2. Information: data that are processed to be useful; provides answer to “who”, “what”, “where”, and “when” questions
> 3. Knowledge: application of data and information; answers “how” questions
> 4. Understanding: appreciation of “why”
> 5. Wisdom: evaluated understanding

Another interesting view is presented in [IJ05, Chapter 2] which discusses a *cognitive framework for information* based on the view that 4 stages of information processing can be distinguished:

The nomadic stage information units are handled separately and independently of each other as if they were simple self-contained entities.

The structural stage : information is seen as a more complex entity consisting of several information units.

The contextual stage : in addition to an analysis of the structural organization of the information units, there is required information on context to disambiguate the meaning of the message.

A cognitive or epistemic stage : information is seen as supplementary or complementary to a conceptual system that represents the information-proceessing system's knowledge of the world

Furthermore, it is pointed out that cognitive structures are the result of social interactions between individual actors entaining shared understanding of concepts.

We adopt the point of view that data can be seen as a collection of facts, observations, etcetera. In more technical terms, data are "just bits". Information, then, can be defined as the meaning (semantics) of this data. The following example illustrates this: if somebody gives you a piece of paper, containing the text "€25" without saying what the purpose of this text is, then this probably does not mean much to you. Therefore, we also need to specify the meaning (semantics) of that data and state that this €25 is the price of a given book. As such, data can be seen as en extensional representation of information.

A similar and equally important distinction is that of data itself and the technology that was used to store it. The word technology, in this context, is used in the broad sense and can be paper, a database, a file but also the knowledge in peoples heads (to be accessed using e.g., a conversation).

Definition 3.2.1 (Data resource) *A data resource is an entity on the Web that may provide information to some player. Data resources are identified by means of a* URI.

Definition 3.2.2 (Information resource) *An information resource is an abstract entity. Facts or observations pertaining to an information resource can be represented in a data resource. As such, information resources can be thought of as the (real-world) entities that one may wish to converse about.*

In our formal model we let $\mathcal{DR}$ denote the set of data resources and $\mathcal{IR}$ denote the set of information resources. Consider the following example to illustrate the distinction between the two:

Example 3.2.1 *We all know the famous painting* the Mona Lisa*; it is a thing that can be thought about, talked about, photographed etcetera. As such it is an information resource. Several data resources may be associated with this information resource such as a photograph, or a detailed textual description that describes the painting, its current location, value, etcetera.*

Observe that information resources deal with informational value aspects (see Section 2.3 but also Section 2.4 for a possible "implementation" of information resources). The example also illustrates that different data resources may 'implement' an information resource in different ways. We will get back to this discussion later. For the time being, we introduce the set of representations $\mathcal{RP}$ with $\mathsf{IRes} : \mathcal{RP} \rightarrow \mathcal{IR}$ and $\mathsf{DRes} : \mathcal{RP} \rightarrow \mathcal{DR}$. As such they form the bridge between the abstract world of information resources and the concrete world of data resources:

Definition 3.2.3 (Representation) *A representation represents the fact that information resources are represented as a data resource.*

Each information resource should have some representation and each data resource should be involved in some representation. The intuition is that each data resource is about at least one information resource (see for example [HLR96]). In our formal theory this is enforced by the following two axioms:

Axiom 12 IRes is a surjective function

Axiom 13 DRes is a surjective function

Similar to the approach taken in RDF, which makes a distinction between resources and labels, we also introduce the notion of data values which may be associated with data resources by means of attributions. Even more, data resources themselves can be interconnected. On the web the most prominent way of implementing these connections is by way of *hyperlinks*, a mechanism which dates back to the notion of hypertext (For more details see e.g., [Bus45, Con87]). Therefore:

Definition 3.2.4 (Data value) *Literals such as strings or integers that may be associated to data resources by means of attributions.*

Definition 3.2.5 (Attribution) *Attributions associate data values to data resources.*

Definition 3.2.6 (Relation) *Relations connect data resources to data resources. These relations have a direction.*

In our formal model we let $\mathcal{DV}$ be the set of data values, $\mathcal{AT}$ be the set of attributions and $\mathcal{RL}$ be the set of relations. Collectively the data values and data resources are referred to as data elements: $\mathcal{DL} \triangleq \mathcal{DR} \cup \mathcal{DV}$. Similary, attributions and relations are dubbed connections: $\mathcal{CN} \triangleq \mathcal{AT} \cup \mathcal{RL}$. The sources and destinations of connections between data elements are yielded by the functions $\mathsf{Src}, \mathsf{Dst} : \mathcal{CN} \rightarrow \mathcal{DL}$ respectively. As an abbreviation we introduce:

$$\begin{array}{lcl} s \overset{c}{\rightsquigarrow} d & \triangleq & \mathsf{Src}(c) = s \ \wedge \ \mathsf{Dst}(c) = d \\ s \rightsquigarrow d & \triangleq & \exists_c \left[s \overset{c}{\rightsquigarrow} d \right] \end{array}$$

The following example illustrates how this may be used in practice:

Example 3.2.2 *Let* `monalisa.htm` *be a webpage about the mona lisa,* `louvre.htm` *be the website of* Le Louvre *in Paris, and* `monalisa.jpg` *be an image. The relations between these data resources could be modeled as:* `louvre.htm` $\rightsquigarrow$ `monalisa.htm`*,* `monalisa.htm` $\rightsquigarrow$ `monalisa.jpg`*. Furthermore, the fact that* `louvre.htm` *presently has version* 2.1 *could be represented as* `louvre.htm` $\rightsquigarrow$ 2.1*.*

Since Src and Dst are total functions, it follows that for each connection its source and destination can not be void.

Corollary 4 $c \in \mathcal{CN} \ \Rightarrow \ \exists_{e_1,e_2} \left[e_1 \overset{c}{\rightsquigarrow} e_2 \right]$

Even more, connectors can not be connected to by yet other connectors; they can not be nested. In order to prove this we must firstly enforce that the base sets introduced so far are disjoint:

Axiom 14 $\mathcal{IR}, \mathcal{DR}, \mathcal{RP}, \mathcal{DV}, \mathcal{AT}$, and $\mathcal{RL}$ are disjoint sets.

We can now prove that:

Lemma 3 $e_1 \overset{c}{\rightsquigarrow} e_2 \ \Rightarrow \ \{e_1, e_2\} \cap \mathcal{CN} = \emptyset$

Proof:

Let $e_1 \overset{c}{\rightsquigarrow} e_2$. From the definition of $\_ \overset{-}{\rightsquigarrow} \_$ we know that $\mathsf{Src}(c) = e_1$ and that $\mathsf{Dst}(c) = e_2$. From the definitions of Src and Dst we know that $e_1, e_2 \in \mathcal{DL}$. From Axiom 14 it now follows that e_1, e_2 can not be members of $\mathcal{CN}$ and thus that $\{e_1, e_2\} \cap \mathcal{CN} = \emptyset$.

□

Recall that relations connect data resources to data resources and that attributions connect data resources to data values. Hence it follows that the source of a connection is always a data resource, that the destination of a relation is always a data resource and that the destination of an attribution is a data value:

Axiom 15 (Relations) $\forall_{a \in \mathcal{RL}} \left[\mathsf{Src}(a) \in \mathcal{DR} \ \wedge \ \mathsf{Dst}(a) \in \mathcal{DR} \right]$

Axiom 16 (Attributions) $\forall_{a \in \mathcal{AT}} \left[\mathsf{Src}(a) \in \mathcal{DR} \ \wedge \ \mathsf{Dst}(a) \in \mathcal{DV} \right]$

Connections can be used to construct *complex data elements*. The intuition is that complex elements can be constructed from other elements. As such, the distinguishing characteristic of complex objects is that their existence depends on the composing data elements, similar to the notion of aggregation in modeling languages such as UML (See e.g., [BRJ99]). The precise mechanism for the construction of complex data elements is discussed later in Section 3.4.4. The following examples illustrate the main idea:

Example 3.2.3 *A Zip-file consisting on a number of Postscript files is a complex element (a complex data resource, to be precise). An Office document containing embedded objects, such as a diagram and tables, is a complex element as well.*

The connections that are used to construct complex data elements are referred to as *accessors* since they provide access to the constituent elements of a complex element. Let $\mathcal{AC} \subseteq \mathcal{CN}$ be the set of accessors. Note that not all connections in a complex instance have to be accessors. For example, it may be the case that a *Zip* file has several *Files* (accessor) at its base, but also have a *Version* attribute (not an accessor).

The construction of complex instances is restricted in the sense that cyclic behavior is forbidden: it is illegal if an instance a is used to construct instance b while at the same time b is used to construct a:

Axiom 17 (Acyclic construction) The graph spanned by the relation R defined as $e_1 R e_2 \triangleq \exists_{a \in \mathcal{AC}} [e_1 \overset{a}{\rightsquigarrow} e_2]$ is acyclic.

In sum, we define resource space to be defined by the following signature:

$$\Sigma = \langle \mathcal{IR}, \mathcal{DR}, \mathcal{RP}, \mathcal{RL}, \mathcal{AT}, \mathcal{DV}, \mathcal{AC}, \mathsf{IRes}, \mathsf{DRes}, \mathsf{Src}, \mathsf{Dst} \rangle$$

3.3 Resource base

In the introduction we already briefly introduced the notion of *resource base* and defined it to be the set of resources that are available at a certain point in time. In other words, given a resource space Σ, a resource base therefore essentially corresponds to a substructure of Σ that on itself forms a resource (Sub)space. More formally, a resource base Σ_B is a sub-structure of Σ:

$$\Sigma_B = \langle \mathcal{IR}_B, \mathcal{DR}_B, \mathcal{RP}_B, \mathcal{RL}_B, \mathcal{AT}_B, \mathcal{DV}_B, \mathcal{AC}_B, \mathsf{IRes}_B, \mathsf{DRes}_B, \mathsf{Src}_B, \mathsf{Dst}_B \rangle$$

such that:

- the $X_B \subseteq X$ for all the base sets,
- $\mathsf{IRes}_B, \mathsf{DRes}_B, \mathsf{Src}_B, \mathsf{Dst}_B$ are restricted to the respective base sets
- all the above discussed axioms apply to the substructure.

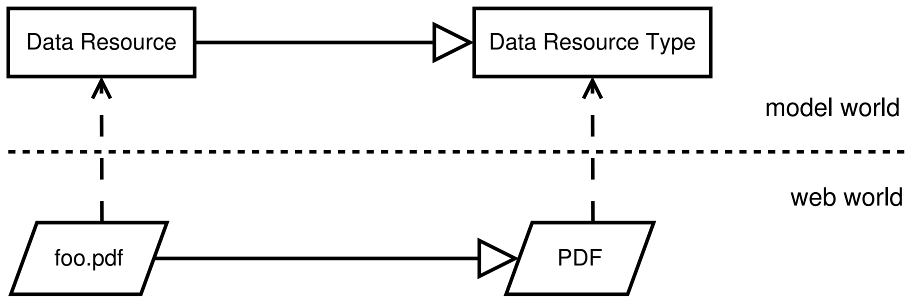


Figure 3.2: Abstraction levels for type-instance relations

Using this mechanism it is quite possible to distinguish between different resource bases such as the state of affairs on the Web, or in a certain digital library. Even more, since a resource base is considered at a specific point in time it also allows us to study the evolution of the population of a certain resource base over time. These topics are beyond the scope of this dissertation but the interested reader is referred to [Pro94] for an overview of theory on evolution. A small example population of a resource base would be:

Example 3.3.1 *We consider a small resource base with:*

- *In our resource base we have two data resources:* $\mathcal{DR}_B = \{\texttt{a.html}, \texttt{b.html}\}$.
- *Both pertain to the same same information resource* $i \in \mathcal{IR}_B$*:* $\mathsf{IRes}_B(\texttt{a.html}) = \mathsf{IRes}_B(\texttt{b.html}) = i$.
- *Finally, these data resources refer to each other by means of hyperlinks. This implies that there are* two *relations* $r_1, r_2 \in \mathcal{RL}$*:* $\texttt{a.html} \overset{r_1}{\rightsquigarrow} \texttt{b.html}$ *and* $\texttt{b.html} \overset{r_2}{\rightsquigarrow} \texttt{a.html}$.

3.4 Typing

In this section we will present our typing mechanism. Before we set out to do so we point out the different levels of abstraction when considering type-instance relations. These two levels are illustrated in Figure 3.2. The first level of abstraction deals with type-instance relations that cross the border between the *model world* and the *web world*. In the figure this is exemplified by the observation that `foo.pdf` is a data resource and the observation that *Pdf* is a data resource type. The second level abstraction deals with type-instance relations within the web world (and their reflection in the model-world). This is exemplified by the observation that `foo.pdf` has the type *Pdf*. In the remainder of this section we will focus on the second abstraction level.

3.4.1 Typing mechanism for descriptive elements

It is important to observe that there are different kinds of data resources, representations, attributes, etcetera. This is best illustrated with typical examples of kinds of data resources: *Html*-files, *Pdf*-files, *E-services* and even *humanoid* can be considered classes of data resources. As such they can be thought of as MIME-types (See e.g., [BF92, BF92]). Similarly, examples of classes of relations are *Hyperlink* and *Part-of*; examples of classes of attributions are *Version* and *Price* and so on.

In order to deal with such a classification mechanism we introduce a typing mechanism on resource space (and consequently any resource base within this space). Firstly, all elements in resource space are typed. Therefore, let $\mathcal{RE}$ denote the set of all elements in resource space:

$$\mathcal{RE} \triangleq \mathcal{DL} \cup \mathcal{CN} \cup \mathcal{RP}$$

These resource space elements form the basis for a uniform typing mechanism. Let $\mathcal{TP}$ be the set of types and $\mathsf{HasType} \subseteq \mathcal{RE} \times \mathcal{TP}$ be the relation for typing descriptive elements in our model. Our typing mechanism (and thus also the set $\mathcal{TP}$) is inspired by the notion of *abstract data types* and *many sorted algebras* as introduced in e.g., [GTWW77, BW90a] since it allows us to focus on the mechanism rather than the underlying "implementation details". It is not our intention to define a 'definite' set/taxonomy of types. Instead, our goal is to describe the typing mechanism as it exists in resource space, and use this model as a basis for (value adding) transformations in Chapter 4. The following example describes how the data resource type *Ascii* can be represented as a many sorted algebra:

Example 3.4.1 *Let $a \in \mathcal{TP}$ denote the type Ascii, then Σ_a is presumed to denote the signature of the many sorted algebra associated to a. This algebra consists of a carrier set (which in turn consists of the base sets that are already known) and operations on these carrier set. In the Ascii case the carrier set consists of the set of all natural numbers $\mathbb{N}$ and $\mathcal{C}$ the set of all characters. Furthermore, the signature holds two functions: $\mathsf{Char} : \mathcal{C} \to \mathbb{N}$ (takes a number n as parameter and returns the nth character from an Ascii string) and $\mathsf{Len} : \to \mathbb{N}$ (Returns the length of an Ascii string). In summary, the signature for a is:*

$$\Sigma_a = \langle \{\mathbb{N}, \mathcal{C}\}, \mathsf{Char}, \mathsf{Len} \rangle$$

3.4.2 Types and population

Given some element from resource space we can use $\mathsf{HasType}$ to determine the set of types of this element. For example, the types of a given file (a data resource) may be *Xml*, *Sgml* and *File*. Conversely, we can determine the set of elements of a given type. Formally, we use the functions τ and π to yield these sets:

$$\tau(e) \triangleq \{t \mid e \,\mathsf{HasType}\, t\} \qquad \pi(t) \triangleq \{e \mid e \,\mathsf{HasType}\, t\}$$

These functions may be generalized to sets of elements and types respectively:

$$\tau(E) \triangleq \bigcup_{e \in E} \tau(e) \qquad \pi(T) \triangleq \bigcup_{t \in T} \pi(t)$$

If $X \subseteq \mathcal{RE}$ is a set of resource space elements, in particular one of the base sets such as $\mathcal{DR}$ or $\mathcal{AT}$ then we will abbreviate $\tau(X)$ as X_τ. The following example illustrates this. Let $\mathcal{DR} = \{e_1, e_2, e_3\}$ be the set of all data resources such that e_1 HasType t_1, e_2 HasType t_2 and e_3 HasType t_2. In this case $\mathcal{DR}_\tau = \tau(\mathcal{DR}) = \{t_1, t_2\}$.

It follows that an element may have more than one type. An example from the domain of data resources illustrates this. Suppose that $E = \{\texttt{1.htm}, \texttt{2.xml}\}$ such that `1.htm` HasType *Html*, `1.htm` HasType *Xml* and `2.xml` HasType *Xml*. In this case we have $E_\tau = \{Html, Xml\}$. This example shows that $\tau(\pi(Html)) \subseteq \tau(\pi(Xml))$. We will get back to this when we discuss subtyping in Section 3.4.5.

We presume that all elements have a type, and that types exist only if they have a population. This must hold in resource *space*. It does not necessarily refer to a specific resource *base*. In a specific resource base, it is quite possible to have a population for a certain type at a certain point in time but not at the next. The proper behavior of typing in our model is enforced by the following two axioms:

Axiom 18 (Total typing) $\tau(e) \neq \emptyset$

Axiom 19 (Existential typing) $\pi(t) \neq \emptyset$

As a consequence we can now prove that an element is in the population of its type and that a type is in the type-set of its population:

Corollary 5 $e \in \pi(\tau(e))$

Corollary 6 $t \in \tau(\pi(t))$

Following the line of reasoning for the total typing axiom, two types are presumed to be equal if their populations are equal:

Axiom 20 (Equal types) $\pi(s) = \pi(t) \;\Rightarrow\; s = t$

This axiom is particularly relevant for the definition of subtyping in Section 3.4.5. The partitioning of elements of resource space (Axiom 14) should be obeyed by their types as well:

Axiom 21 $\mathcal{IR}_\tau, \mathcal{DR}_\tau, \mathcal{RP}_\tau, \mathcal{DV}_\tau, \mathcal{AT}_\tau$, and $\mathcal{RL}_\tau$ form a partition of $\mathcal{TP}$

In our approach we adopt a *types follow population* approach, which constrasts with a typical database approach. In the world of relational databases (See e.g., [Dat03]) one usually starts by defining the schema after which the data is fitted to the schema. That said, we take the opposite approach and assert that a type exists as soon as it has population (see also Axiom 19). This allows for the much needed flexibility to deal with the ever changing nature of resources on the Web.

3.4.3 Typing of connectors

We already explained that we distinguish between two classes of connectors: relations (relating data resources to data resources) and attributions (relating data resources to data values). Furthermore, it seems obvious that there are different kinds of relations and different kinds of attributes. It might be tempting to introduce a Src and Dst at the typing level as follows:

- if we observe $e_1 \overset{c}{\rightsquigarrow} e_2$
- and if we know that $e_1 \,\mathsf{HasType}\, t_1$, $c \,\mathsf{HasType}\, t$, and $e_2 \,\mathsf{HasType}\, t_2$
- then at the typing level we can conclude that $\mathsf{Src}(t) = t_1$ and $\mathsf{Dst}(t) = t_2$

This would, however, be a bad idea because a single connection type may provide connections between instances of different combinations of types. For example, both *Zip*-files and *Tar*-files may have a *Payload* (connector type) consisting of *File*s (data resource type). In one case, *Payload* connects between *Zip* and *File* and in the other it connects between *Tar* and *File*. As a result, there is no functional dependency between connector types and the underlying types that are connected.

The connections between elements of different types that can be made by a connector type are therefore formally represented by the relation:

$$\_ \xrightarrow{-} \_ \subseteq \mathcal{TP} \times \mathcal{CN}_\tau \times \mathcal{TP}$$

If $t_1 \xrightarrow{t} t_2$ then the intuition is that instances of type t_1 can connect to instances of type t_2 via connections of type t. Note that the participation is not mandatory. For example, *Zip* files may connect to *Comment*, denoting the fact that *Zip* files may have an attribution of type *Comment*. Not all *Zip* files have to have this, though. Similarly, not all types have to be connected to each other by means of the same connector type; for example, in case of a type such as *Postal address* it does not make sense to attach it to a connector of type *Payload*. The actual enforcement of the proper behavior of instances will be inforcecd by the axioms introduced below.

Since *types follow population*, if $t_1 \xrightarrow{t} t_2$ holds then there must at least be some instances of these types to make this true; the evidence for this observation:

Axiom 22 (Soundness of $\_ \xrightarrow{-} \_$) $t_1 \xrightarrow{t} t_2 \;\Rightarrow\; \exists_{e_1 \in \pi(t_1), c \in \pi(t), e_2 \in \pi(t_2)} \, [e_1 \overset{c}{\rightsquigarrow} e_2]$

Note that this axiom is very much needed, even though we already had the axiom of existential typing (Axiom 19). Due to total typing we know that if $t_1 \xrightarrow{t} t_2$ then the populations of the types t_1, t and t_2 are all non-empty. There is, however, no way to be sure that any instance of the population of t actually bridges (provides evidence) between the populations of t_1 and t_2. To illustrate this further, consider the case where t_1 is the data resource type *Zip*, t is the connector type *Payload* and t_2 is the type *File*. As we've seen before, $t_1 \xrightarrow{t} t_2$ denotes the fact that *Zip*-files have a payload consisting of files. Axiom 22 states, then, that at least one *Zip*-file exists that has at least one file in its payload.

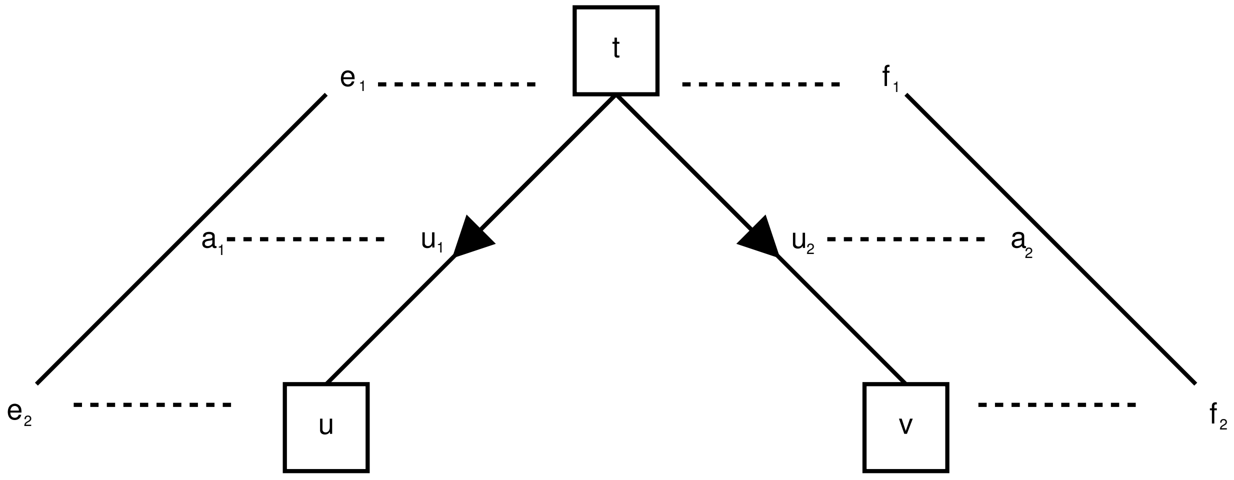


Figure 3.3: Behavior of connectors and their types

Conversely, if a connector c exists such that $e_1 \overset{c}{\rightsquigarrow} e_2$ then this must be reflected at the typing level. In other words, we can conclude that the types of these instances behave as follows:

Axiom 23 (Completeness of $\_ \overset{-}{\rightarrow} \_$) $e_1 \overset{c}{\rightsquigarrow} e_2 \;\Rightarrow\; \exists_{t_1 \in \tau(e_1), t \in \tau(c), t_2 \in \tau(e_2)} [t_1 \overset{t}{\rightarrow} t_2]$

Figure 3.3 illustrates this situation. For any given connector (e.g. a_1 or a_2), there must be instances attached to this connector such that both the connector and the elements it connects have the proper types. A direct result from Axiom 22 is that Lemma 3 can be translated for types as well. Simply put, if the types t_1 and t_2 are connected via a connection type t, then they can not be connection types themselves:

Corollary 7 (No connector nesting) $t_1 \overset{t}{\rightarrow} t_2 \;\Rightarrow\; \{t_1, t_2\} \cap \mathcal{CN}_\tau = \emptyset$

For a given type, the set of connector types that are attached to it can be determined by:

$$\mathsf{Conn}(t_1) \;\triangleq\; \left\{ t \;\middle|\; \exists_{t_2} [t_1 \overset{t}{\rightarrow} t_2] \right\}$$

To show the proper behavior of this relation we must show that if two types have the same outbound connections then this must be reflected by the Conn relation. More formally:

Lemma 4 $\forall_{u,t} [s_1 \overset{u}{\rightarrow} t \Leftrightarrow s_2 \overset{u}{\rightarrow} t] \;\Rightarrow\; \mathsf{Conn}(s_1) = \mathsf{Conn}(s_2)$

Proof:

Let us suppose that $\forall_{u,t} [s_1 \overset{u}{\rightarrow} t \Leftrightarrow s_2 \overset{u}{\rightarrow} t]$. Let $u \in \mathsf{Conn}(s_1)$. From the definition of Conn it follows that there are u, t such that $s_1 \overset{u}{\rightarrow} t$. Under the assumption made it follows that also $s_2 \overset{u}{\rightarrow} t$. From the definition of Conn it then follows that $u \in \mathsf{Conn}(s_2)$. The proof of the inverse is identical due to the symmetry of $\Leftrightarrow$.

□

3.4.4 Complex data resources

As mentioned before, some data resources are composed of other data elements. The main example used for these *accessors* so far was that of *Zip*-files which have a payload. As such, the examples given so far are mostly of this kind. At the instance level, we already introduced the special class of connections called accessors ($\mathcal{AC}$) to signify that a specific data resource should be regarded as being composed of other elements. We will now elaborate on the behavior of these accessors on the typing level.

A type is complex if instances of this type are complex. To put it differently, a complex type has (outgoing) accesor types associated to it. A special class of connection types is that of *accessor types* ($\mathcal{AC}_\tau \subseteq \mathcal{CN}_\tau$). A connection type is an accessor type if its instances are accessors. The set of accessor types that are indeed associated to a type is defined as:

$$\mathsf{Act}(t) \triangleq \mathsf{Conn}(t) \cap \mathcal{AC}_\tau$$

Using the definition of Act we can now define the set of complex types as follows:

$$\mathcal{TP}_c \triangleq \{t \mid \mathsf{Act}(t) \neq \emptyset\}$$

In Figure 3.3 we could, thus, have: $t \in \mathcal{TP}_c$ and $\mathsf{Act}(t) = \{u_1, u_2\}$. Since we know that $\mathcal{AC}_\tau \subseteq \mathcal{CN}_\tau$ we can show that the $\_ \overset{\_}{\to} \_$ must behave property for accessor types as well; similar to what we showed in Lemma 4:

Corollary 8 $\forall_{u,t} [s_1 \overset{u}{\to} t \Leftrightarrow s_2 \overset{u}{\to} t] \;\Rightarrow\; \mathsf{Act}(s_1) = \mathsf{Act}(s_2)$

Thus far we have not associated any semantics to accessors and complex types in terms of their influence on a population in resource space other than state that complex instances are constructed by means of other instances. In other words, if an instance has a complex type then it must have at least one accessor with an instance at its base to avoid 'fake construction':

Axiom 24 (Proper construction) $e_1 \in \pi(\mathcal{TP}_c) \;\Rightarrow\; \exists_{a \in \mathcal{AC}, e_2} [e_1 \overset{a}{\rightsquigarrow} e_2]$

For convenience, the definition of Act is overloaded to include the instance level as follows:

$$\mathsf{Act}(e) \triangleq \bigcup_{t \in \tau(e)} \mathsf{Act}(t)$$

In other words, the accessor types associated to an instance are the union of the sets of accessor types associated to all the types of the instance under consideration. Note again that it may be the case that some of the accessor types in an instance of a complex type are unused. The following example illustrates how the accessor mechanism works in practice.

Example 3.4.2 *Suppose* `x.zip` *is a Zip-file. Its payload consists of three files,* `a.doc`, `b.ps` *and* `c.pdf`*. They can be accessed by their respecitve accessors a_1, a_2 and a_3 which all have the accessor type Payload. Furthermore, there is a comment and a password associated*

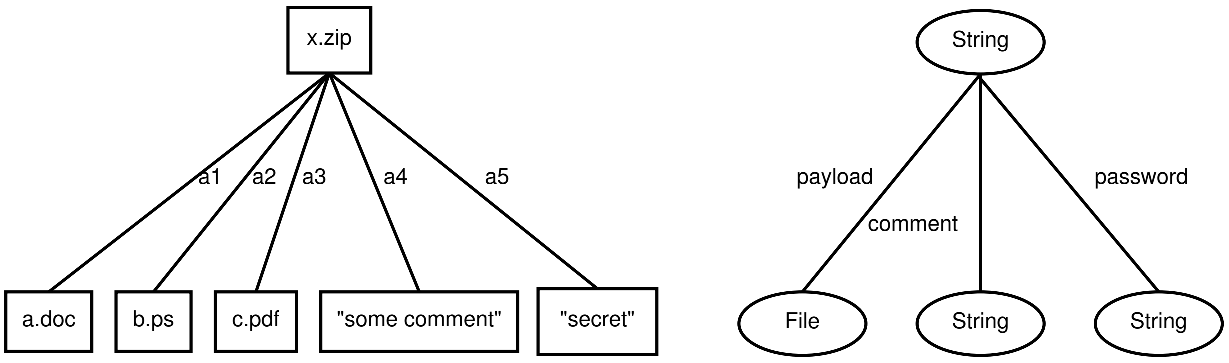


Figure 3.4: Accessors and typing

to the Zip-file. These are accessed via accessors a_4 and a_5 with accessor types Comment and Password, respectively. The accessor types originate from attributions. Formally:

$$\begin{aligned}
\pi(\mathcal{DR}) &= \{\texttt{x.zip}, \texttt{a.doc}, \texttt{b.ps}, \texttt{c.pdf}\}\\
\pi(\mathcal{DR}_\tau) &= \{\mathit{Zip}, \mathit{Doc}, \mathit{Postscript}, \mathit{Pdf}\}\\
\pi(\mathcal{DV}) &= \{\text{"some comment"}, \text{"secret"}\}\\
\pi(\mathcal{DV}_\tau) &= \{\mathit{String}\}\\
\pi(\mathcal{CN}) &= \{a_1, a_2, a_3, a_4, a_5\}\\
\pi(\mathcal{CN}_\tau) &= \{\mathit{Payload}, \mathit{Comment}, \mathit{Password}\}\\
\pi(\mathcal{AT}) &= \{a_4, a_5\}\\
\pi(\mathcal{AT}_\tau) &= \{\mathit{Comment}, \mathit{Password}\}\\
\pi(\mathcal{AC}) &= \{a_1, a_2, a_3\}\\
\pi(\mathcal{AC}_\tau) &= \{\mathit{Payload}\}
\end{aligned}$$

This is illustrated in Figure 3.4. The left-hand side of the figure is at the instance level, whereas the right hand side is at the typing level. Note that for $t \in \tau(\{a_4, a_5\})$ it holds that $\mathit{Zip} \xrightarrow{t} \mathit{String}$. Similarly, for $t \in \tau(\{a_1, a_2, a_3\})$ it holds that $\mathit{Zip} \xrightarrow{t} \mathit{File}$; the actual types *Doc*, *Postscript* and *Pdf* are considered to be subtypes of the type *File*. Subtyping is the topic of the next section.

3.4.5 Subtyping

In many modeling languages (such as for example *Object Role Modeling*, *Entity Relationship Modeling* and UML) a subtyping mechanism is used to denote *is-a* relations between sets of elements. For example, the statement "each woman is a person" has the same connotation as "woman is a subtype of person".

We assume the existence of subtyping for types in our model. Let $\underline{\mathsf{SubOf}} \subseteq \mathcal{TP} \times \mathcal{TP}$ therefore define the subtyping relationship, where $s \, \underline{\mathsf{SubOf}} \, t$ indicates that type s is a subtype of or equal to type t. Conversely, let $s \, \mathsf{SubOf} \, t$ denote proper subtyping. The mapping

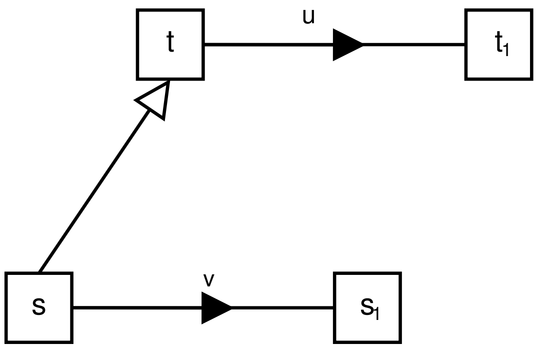


Figure 3.5: Illustrating the behavior of accessors in case of subtyping

to sets of elements is defined as follows:

$$\begin{aligned} s \,\underline{\mathsf{SubOf}}\, t &\triangleq \pi(s) \subseteq \pi(t) \\ s \,\mathsf{SubOf}\, t &\triangleq \pi(s) \subset \pi(t) \end{aligned}$$

From these definitions we can easily prove that:

Corollary 9 $s \,\mathsf{SubOf}\, t \;\Rightarrow\; s \neq t$

Furthermore, we know that SubOf is transitive, irreflexive, and asymmetric and that $\underline{\mathsf{SubOf}}$ is transitive, reflexive, and antisymmetric:

Corollary 10 $\underline{\mathsf{SubOf}}$ is transitive, reflexive and anti-symmetric

Corollary 11 SubOf is transitive, irreflexive and asymmetric

To illustrate what happens with connections (connection types) in case of subtyping, consider the situation as illustrated in Figure 3.5. In this figure the open-ended arrow denotes subtyping and a closed arrow denotes the accessor relation from a complex type to its base. The figure denotes a situation where $s \,\mathsf{SubOf}\, t$, $\mathsf{Conn}(t) = \{u\}$, and $\mathsf{Conn}(s) = \{u, v\}$. Furthermore:

- Suppose we know that $e_1 \overset{a}{\rightsquigarrow} e_2$ such that $e_1 \in \pi(s)$, $a \in \pi(v)$ and $e_2 \in \pi(s_1)$. Because of subtyping we know that $e_1 \in \pi(t)$. Therefore, at the typing level we know that e_1 *may* have a connector with type u leading to an instance of type t_1. Without evidence for this at the instance level, we only know that $s \xrightarrow{v} s_1$.

- Suppose we know that $f_1 \overset{b}{\rightsquigarrow} f_2$ and assume that $f_1 \in \pi(s)$, $b \in \pi(u)$ and $f_2 \in \pi(t_1)$. Based on this evidence we know that $\exists_{c,f_3} [f_1 \overset{c}{\rightsquigarrow} f_3]$ such that $c \in \pi(v)$ and $f_3 \in \pi(s_1)$, otherwise there would be no proof for $f_1 \in \pi(s)$! This follows directly from the definition of $\mathcal{CN}$ and Axiom 22: from the definition of $\mathcal{CN}$ it follows that $s \xrightarrow{v} s_1$ and Axiom 22 states that there must be at least one set of instances to provide evidence for this observation at the typing level. We can now conclude that $s \xrightarrow{u} t_1$ and $s \xrightarrow{v} s_1$.

- Suppose we know that $g_1 \overset{c}{\rightsquigarrow} g_2$ such that $g \in \pi(t)$, $c \in \pi(u)$ and $g_2 \in \pi(t_1)$. We may already conclude that $t \xrightarrow{u} t_1$. If we find evidence for $g_1 \overset{d}{\rightsquigarrow} g_3$ such that $d \in \pi(v)$ and $g_3 \in Pop(s_1)$ then we have $g_1 \in \pi(s)$ and may also conclude that $s \xrightarrow{v} s_1$.

This also has consequences for applications that work on complex instances. Consider the situaton where t is the *Zip*-type and s extends the standard *Zip* with a password. Take an $e \in \pi(s)$ and consider it from the perspective of type t (This is allowed since $s\,\mathsf{SubOf}\,t$ and thus $\pi(s) \subset \pi(t)$). Similar to what we know from *interfaces* in object orientation (See e.g., [BRJ99]), the interface of t will simply ignore the additional specificities of the interface of s. In the real world this means that applications may fail if they implement the t-interface and receive an instance built according to the s-interface.

These insights lead to the following: firstly, connector types and their respective bases are inherited across a subtyping relation. That is, subtypes inherit the connector types associated to their super types (but not vice versa):

Axiom 25 (Completeness of connections)
$s\,\mathsf{SubOf}\,t \;\wedge\; t \xrightarrow{u} v \;\Rightarrow\; s \xrightarrow{u} v$

From this axiom it follows immediately that:

Corollary 12

- $s\,\mathsf{SubOf}\,t \;\Rightarrow\; \mathsf{Conn}(t) \subseteq \mathsf{Conn}(s)$
- $s\,\mathsf{SubOf}\,t \;\Rightarrow\; \mathsf{Act}(t) \subseteq \mathsf{Act}(s)$
- $s\,\mathsf{SubOf}\,t \;\wedge\; t \in \mathcal{TP}_c \;\Rightarrow\; s \in \mathcal{TP}_c$

Following the same line of reasoning, the base of connector types must also be complete. That is, if instances of type s can connect to instances of type t_1 and we know that t_2 is a subtype of t_1 then we know that instances of s can also connect to instances of type t_2:

Axiom 26 (Completeness of base) $s \xrightarrow{u} t_1 \;\wedge\; t_2\,\mathsf{SubOf}\,t_1 \;\Rightarrow\; s \xrightarrow{u} t_2$

For the remainder of the discussion we need to introduce the notion of *type relatedness*. We define that two types are type related if their populations overlap:

$$s \sim t \;\triangleq\; \pi(s) \cap \pi(t) \neq \emptyset$$

We already discussed subtyping which is one form of type relatedness. From e.g., Lisa-D we know that other forms of type relatedness exist as well [HPW93]. For example, assume that a type (e.g., *Zip*) has two distinct subtypes (*Zip* with password and *Zip* with comment), each with its own subtype defining rule. It may be possible to have an instance that is in both the subtypes (i.e., a specific *Zip* file may have both a password and a comment associated to it). Using this intuition we can more easily specify the soundness of the base of accessor types:

Axiom 27 (Soundness of Base) $s_1 \,\mathsf{SubOf}\, s_2 \;\wedge\; s_1 \xrightarrow{u} t_1 \;\wedge\; s_2 \xrightarrow{u} t_2 \;\Rightarrow\; t_1 \sim t_2$

This allows us to prove that if a subtype and a supertype share an accessor, then their bases must at least be type related:

Lemma 5 $s_1 \,\mathsf{SubOf}\, s_2 \;\wedge\; s_1 \xrightarrow{u} t_1 \;\wedge\; s_2 \xrightarrow{u} t_2 \;\Rightarrow\; t_1 \sim t_2$

Proof:

Let $s_1 \,\mathsf{SubOf}\, s_2$. Furthermore, let $s_1 \xrightarrow{u} t_1 \;\wedge\; s_2 \xrightarrow{u} t_2$. From Axion 25 we know that $s_1 \xrightarrow{u} t_2 \;\wedge\; s_1 \xrightarrow{u} t_2$. From Axiom 27 we can now conclude that $t_1 \sim t_2$.

□

Note that we did not include a "soundness of complex" axiom. The following example illustrates why such an axiom is not desirable:

Example 3.4.3 *Let s be the Zip type, t be the E-mail type and v be the Payload type ($v \in \mathcal{AC}_\mathcal{T}$). Furthermore, let $s \xrightarrow{v} u$ and $t \xrightarrow{v} u$. In other words, both Zip and E-mail have an accessor of type Payload that offers access to a Html base.*

If we were to introduce a soundness axiom for complex, we would be forced to conclude that *Zip* and *E-mail* are type related which, in our opinion, is undesirable.

To summarize the discussion so far, we define typed resource space to be defined by the following signature:

$$\Sigma^\tau \;\triangleq\; \langle \Sigma, \mathcal{TP}, \mathsf{HasType} \rangle$$

The distinction between resource *space* and resource *base* is important with respect to the observation that, in our model, types follow instances. Without this explicit distinction, if the last instance of a type would be removed then the type would cease to exist. This can be particularly inconvenient in case of an application that depends on the existence of this type. The world of transformations (see Chapter 4) is a typical example since, as we will show, transformations are described in terms of their input types and output types.

Similar to resource base, a *typed resource base* corresponds to those resources and their types that are available at some moment in time. This set spans a sub-space of a typed resource space, consisting of a resource base and its typing. Given a typed resource space Σ^τ a typed resource base Σ^τ_B therefore corresponds to a substructure of Σ^τ that on itself forms a typed resource subspace. More formally, a typed resource base Σ^τ_B is a substructure of Σ^τ:

$$\Sigma^\tau_B \;\triangleq\; \langle \Sigma_B, \mathcal{TP}_B, \mathsf{HasType}_B \rangle$$

such that:

- the Σ_B is a resource base in resource space Σ
- $\mathcal{TP}_B \subseteq \mathcal{TP}$ and $\mathsf{HasType}_B \subseteq \mathsf{HasType}$
- all the above discussed axioms apply to the sub-structure.

3.5 Language for resource space

In the previous section of this chapter we have presented our model for information supply. In this section we will extend this model with a language which will function as a query and constraint language. We will use this language in the upcoming sections mainly for three reasons:

1. Specify properties that data resources may have at the typing level. For example, we can specify the property that *Html* files may have outgoing hyperlinks.

2. Assert that a certain data resource has a property at the instance level. For example, we can specify that `a.html` has outgoing hyperlinks.

3. Specify queries. For example, we can construct a query that selects data resources that have outgoing hyperlinks.

If we were to state that the former two reasons pertain to *meta data* then we cross the border between the conceptual level of our investigation so far and the level of actual implementations. So far we have presented our view of the information landscape in the form of a conceptual model. If we are to build tools that make use of these models then we will somehow have to gather information to actually populate the model. To this end we could indeed use meta data about data resources such as RDF annotations. We will get back to this discussion later in this dissertation.

In Figure 3.1 we presented the signature of our formalism in the ORM notation. That is, we presented the (names of the) object types and fact types. We have not yet provided verbalizations to navigate over the conceptual schema, however. By adding these names (as well as some abbreviations) we essentially create a LISA-D[2] language that is suitable for our goals as described above. Figure 3.6 does include these verbalizations and therefore is the basis for our language. We will now present the verbalizations as shown in the figure. Observe that we have chosen to display only the query-form of the verbalizations. The asserting form can easily be derived from it. We give an example at the end of this section.

Typing : The typing of data resources can be expressed as follows:

- Data Resource . . . having Type . . .

Aboutness : Expressing the types of data resources is straightforward as we will see shortly. The relationship between data resources and information resources models aboutness. We introduced the concept of representations (a combination of a data resource and an information resources) to be able to specify how a data resource "implements" the information resource. This gives us three possible kinds of expressions:

[2]Appendix B presents an overview of LISA-D.

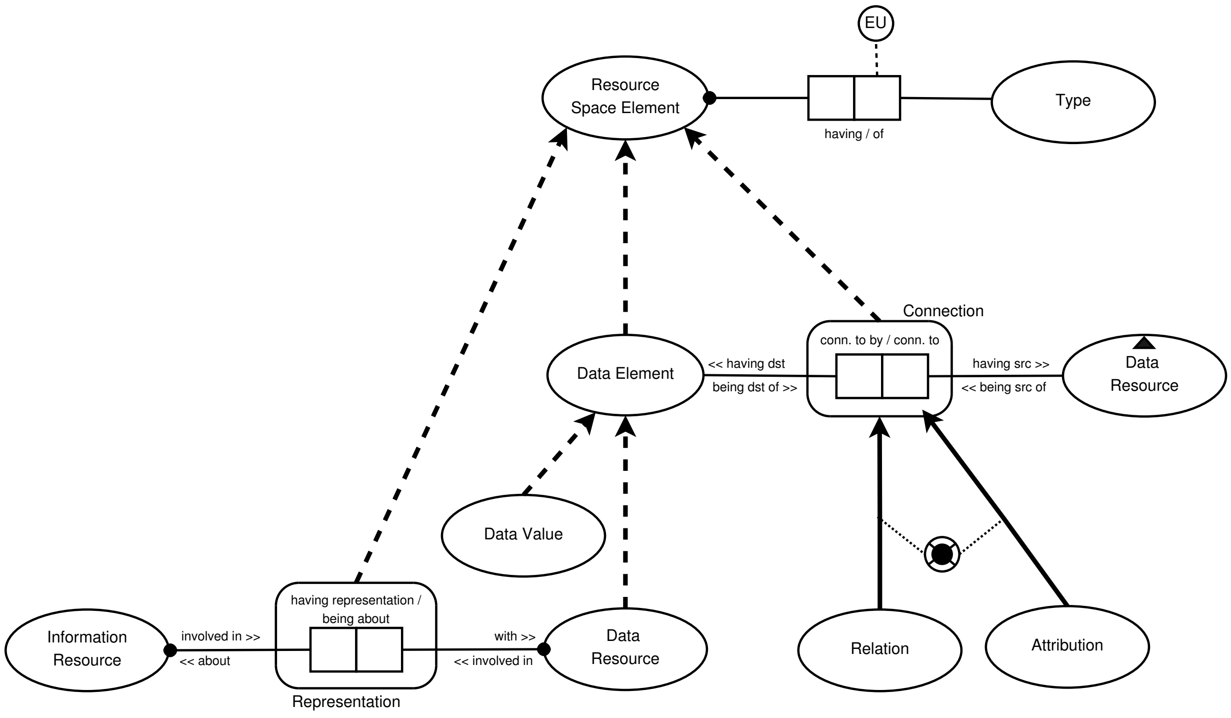


Figure 3.6: Signature of the model for resource space including verbalizations

- Only specifying what a data resource is about:
  Data Resource ... being about Information Resource ...
  Data Resource ... involved in Representation about ...
- Specifying the representation type that the data resource has when it is involved in a representation:
  Data Resource ... involved in Representation of Type ...
- A combination of the above:
  Data Resource ... involved in Representation
  (about ... AND-ALSO of type ...)

Connections : Many things can be asserted about connections (i.e., relations and attributions). Similar to what we saw when we explained the verbalizations for aboutness we can:

- Specify that a data resource is connected to some data element:
  Data Resource ... connected to Data Element ...
  Data Resource ... being source of Connection with destination ...
  By replacing the term Data Element with either Data Resource or Data Value, the expression can be refined. Similarly, the term Connection can be replaced by either Attribution or Relation. As an abbreviation we also introduce ... has

... as an abbreviation for the expression ... being source of ... and ... with ... for the expresson ... with destination ....

- Specify that a data resource is being connected to by another data resource; i.e., expressing that the data resource is the destination of a relation:
  Data Resource ... being connected to by ...
  Data Resource ... being destination of Connection with source ...
- Specify the type of connection:
  Data Resource ... being source of Connection of Type ...
  Data Resource ... being destination of Connection of Type ...
  by replacing the term Connection with either Relation or Attribution, the expression can be refined.
- Combine the above, for example:
  Data resource being source of Relation
  (with destination ... AND-ALSO of Type ...)

These expressions can, in turn, be combined again to make even more complex expressions thus forming a language for specifying requirements (of a searcher) with regard to resource space. A typical example of a query that combines the above would be:

```
Data resource ( having Type “EPS”
   AND-ALSO involved in Representation
      (about “Mona Lisa” AND-ALSO having Type “picture-of” )
   AND-ALSO being destination of Relation having source “davinci.html”)
```

This would find all pictures of the `Mona Lisa` in the *Eps* format that are, somehow, related to the webpage `davinci.html`. We can also use the above verbalizations to express assertions about data resources. An example would be:

```
Data Resource “louvre.html” (has type “HTML”
   AND-ALSO having a Relation with destination “davinci.html”)
```

which asserts that the data resource `louvre.html` which is a *Html* file is the source of a relation that has `davinci.html` as its destination.

3.6 Conclusion

In this chapter we have presented a conceptual model for the information landscape. The entities that actually make up the information landscape are called data resources in our model. Other important building blocks in our model are information resources, representations, data values, attributions, and a strong typing mechanism. With respect to this typing mechanism we adopt a type-follows instance approach as opposed to an instance-follows types approach that is commonly used in database design. The main distinction lies in the fact that we have not specified a rigid and static list of types in our conceptual model. This allows for more flexible implementations.

With respect to this transition from the conceptual level to the implementation level it is important to observe that in this chapter we did not discuss implementation issues yet. Many issues have to be dealt with when designing and implementing real applications when using our model. For example, one has to decide how to get the details on relations, attributions and representations of real data resources. For aboutness (the *informational value domain* as introduced in the previous chapter) one could, for example, decide to use index expressions, for attributions and other aspects pertaining to the *structural value domain* one might decide to rely on RDF annotations and so on. Lastly, for the *emotional domain* one might rely on such things as user modeling (see Section 1.2.7). We get back to these discussions in later chapters when we zoom in on a real search system that tries to achieve aptness-based retrieval.

Finally, the last contribution of this chapter is a language with which we can express properties of data resources (at both the typing and the instance level) which can also be used as a query language. We will use this language throughout the upcoming chapters.

CHAPTER 4 Manipulating information supply

Outline In this chapter we present our framework for *transformations*. By means of transformations we are able to manipulate resources in the information landscape. In our framework we study, among other things, which effects transformations have on resources, after which we show how transformations can be *value adding*. In other words, we also explain how transformations can be used by brokers on the information market as presented in Chapter 2.

This chapter is based on [GPB04, GPBW04, GPBV05, GPBW05].

4.1 Introduction

In the previous chapter we have presented a model for the information landscape built around data resources, aboutness of these data resources, and other aspects (such as relations, attributions, typing and so on) of these data resources. In this chapter we will build on this model and present a framework for *transformations* with which we can manipulate information supply. Two main application areas of transformations are recognized (See also [Läm04]). On the one hand, transformations are essential in system development in general and in transformational specification strategies in particular. On the other hand, when a system has been developed and is being used in a practical context, particular user objects may be available in a heterogeneous environment (e.g., documents in the context of the Web), which introduces the need for transformations. It is exactly this latter aspect that is the main motivation for our work on transformations. Our definition of a transformation is as follows:

Definition 4.1.1 (Transformation) *A system that transforms data resources (of a certain type) into other data resources.*

Note that we do not limit ourselves to database transformations only (see earlier work on transformations in e.g., [BKM94, Pro97]). We also include transformations such as abstract generation, conversions between data resource types, etcetera.

We hypothesize that search systems can benefit a great deal from having a set of transformations available, especially from a searcher point of view. Consider the following motivating example:

Example 4.1.1 *While on the road, a searcher uses his Pda to search the web for information on his stock. The Pda is equipped with software for reading pdf files and html files only. Using his favorite search engine he finds a spreadsheet. In this case it would be very useful if the search engine automatically transforms the spreadsheet to one of the formats which he can access on his Pda. Without such transformation the data resource would be completely useless at this point.*

In this case, the transformation can be said to be *value adding.* Even more, a broker that is capable of executing transformations on demand is considered to be value adding, as long as it executes "useful" transformations. We will get back to this discussion later. Transformations can have an effect on certain properties of data resources. In the given example, the transformation would have an effect on the data resource type. Other examples of effects would be: changing the resolution of an image, the document length, etcetera. The usefulness of a transformation thus depends on the effects that it has and the properties that the searcher desires.

The remainder of this chapter is organized as follows. We will start by presenting a formal framework for transformations in Section 4.2. With this framework we specify what transformations are and how they behave. After that we extend this framework with complex transformations (such as composed transformations, or transformations on instances of complex types; see also Section 3.4.4) in Section 4.3. After that we study the effects that transformations may have, as well as present the formal relation between transformations, resource space and resource base. This is done in Section 4.4. Next, in Section 4.5, we will present some practical aspects that are relevant when implementing a system using such transformations. This includes aspects such as learning the effects of transformations and automatic transformation selection. Last but not least, we will present some experiments with transformations in Chapter 5.

4.2 Transformations

Let $\mathcal{TR}$ be a set of transformations. The semantics of a transformation specify what this transformation actually *does.* The semantics of a transformation is given by the function:

$$\mathsf{SEM} : \mathcal{TR} \rightarrow (\mathcal{DR} \rightarrow \mathcal{DR})$$

In other words, transformations transform one data resource into another. As an abbreviation we use $\overrightarrow{T}$ to denote $\mathsf{SEM}(T)$. In our model, transformations do not change the aboutness of data resources. In other words, if a data resource is associated to a certain information resource then all data resources that can be generated from this data resource using transformations will be associated to the same information resource. We model this as follows. In logic, $M \models A$ is often used to denote the fact that M is a model for A. Here we use the symbol to denote the fact that a data resource is associated to an information resource:

$$i \models d \triangleq \exists_r \left[\mathsf{IRes}(r) = i \ \wedge \ \mathsf{DRes}(r) = d\right]$$

Since we consider data resources to be an implementation for an information resource, we can also consider an information resource to be a model for a data resource. The fact that transformations do not modify the aboutness of data resources can now be expressed as follows:

Axiom 28 ($\mathcal{TR}$ neutral transformations) $i \models d \ \Rightarrow \ i \models \overrightarrow{T}(d)$

Any given transformation has a fixed input and output type for which it is defined, similar to the notion of mathematical functions having a domain and a range. In our formalism we model this using $\mathsf{Input}, \mathsf{Output} : \mathcal{TR} \rightarrow \mathcal{DR}_\tau$. As an abbreviation we introduce:

$$t_1 \xrightarrow{T} t_2 \triangleq \mathsf{Input}(T) = t_1 \ \wedge \ \mathsf{Output}(T) = t_2$$

to express that transformation T transforms data resources of type t_1 into data resources of type t_2. In our formalism, a transformation is identified by its semantics:

Axiom 29 (Identity of transformations) $\overrightarrow{T_1} = \overrightarrow{T_2} \ \Rightarrow \ T_1 = T_2$

Observe that transformations are defined at the typing level. We will now describe the relation with the instance level. Recall that a transformation is only defined for instances of the correct input type, and that it only produces instances of the specified output type. If a transformation is applied to a data resource which is not of its input type then this data resource will not be changed. The proper behavior of transformations at the instance level is enforced by the following axioms:

Axiom 30 (Output of transformations) $e \in \mathsf{Input}(T) \ \Rightarrow \ \overrightarrow{T}(e) \in \mathsf{Output}(T)$

Axiom 31 (Input of transformations) $e \notin \mathsf{Input}(T) \ \Rightarrow \ \overrightarrow{T}(e) = e$

Transformations may also be applied to sets of data resources. Let E be such a set and T a transformation, then the application of T to E results in a new set of data resources:

$$\overrightarrow{T}(E) \triangleq \left\{\overrightarrow{T}(e) \mid e \in E\right\}$$

This means the following. If a transformation T is applied to a set of data resources E then the transformation will transform all resources for which it is defined (Axiom 30). The instances in E that are not in its input type are left untouched (Axiom 31).

Another property of transformations is the fact that they are closed under composition. Transformations can be composed by performing one after the other. We therefore assume $\circ$ to be a binary operator on $\mathcal{TR}$ such that $\overrightarrow{T_1 \circ T_2} = \overrightarrow{T_1} \circ \overrightarrow{T_2}$ denotes transformation composition in terms of mapping composition. We can now prove the following:

Lemma 6 $\circ$ is an associative operator for transformations.

Proof:

As mapping composition is associative we may conclude this property from Axiom 29.

□

Note that we do not require transformations to have an inverse. The following example illustrates the composition of transformations.

Example 4.2.1 *Let* $t_1 \xrightarrow{T_1} t_2$ *and* $t_3 \xrightarrow{T_2} t_4$ *be two transformations such that* $t_4 \neq t_2$. *Let* T *denote a transformation with* $\overrightarrow{T} = \overrightarrow{T_1 \circ T_2}$. *If* T *is applied to a single instance then either one of two things can happen: (1) nothing happens; this is the case when e is not in the input types of* T_1 *and* T_2. *(2) e is actually changed; this is the case when the type of e is either the input type or* T_1 *or the input type of* T_2. *Similarly, if* T *is applied to a set of data resources then the above holds for each of the data resources in this set.*

4.3 Complex transformations

In the previous section we have discussed the basic properties of transformations, and showed how transformations can be composed by sequencing them using the $\circ$ operator. In this section we discuss more complex ways of composing transformations, relying heavily on the accessor types presented in Sections 3.2 and 3.4.4. We call a transformation to be complex if:

- it operates on (instances of) a complex type
- it has an effect on the instances that were used to construct the complex instance (that is, the instances at the *base* of the instance of a complex type).

We distinguish several classes of complex transformations such as transformations that remove accessors of a certain type, transformations that transform instances at the base of a certain accessor type, and type casting transformations. We will introduce the semantics of these complex transformations in the subsequent sections.

4.3.1 Transformations that remove an accessor type

The first complex transformation is used to *remove* accessors of a certain type, as well as the instances at their base. For example, it may be desirable to remove a *Comment* from a *Zip* file, or to remove an attachment from an *E-mail*. Such a transformation:

- takes as its input a data resource and an accessor type,
- removes the accessors from the data resources that have the specified accessor type as well as the data resources at their base,

- leaves other accessors (and their bases) untouched.

We use ϱ_u to denote the semantics of such a transformation. Let e be a complex instance and u the type of one of its accessors. Then, the transformation identified by $\varrho_u(e)$ removes from e all accessors of type u as well as the instances at the base of these accessors:

$$\varrho_u(e) \ltimes u = \emptyset \;\;\wedge\;\; \forall_{a,d}\,[e \overset{a}{\rightsquigarrow} d \;\;\wedge\;\; a \notin \pi(u) \;\;\Leftrightarrow\;\; \varrho_u(e) \overset{a}{\rightsquigarrow} d]$$

In the above definition we have used the following notation: let e be a complex instance and u an accessor type, then $e \ltimes u$ denotes the instances at the base of the accessor from e with type u. More formally:

$$e \ltimes u \;\triangleq\; \{d \mid e \overset{a}{\rightsquigarrow} d \;\;\wedge\;\; a \in \pi(u)\}$$

The intuition behind this shorthand is that $e \overset{a}{\rightsquigarrow} d$ retrieves all data elements that are used in constructing a complex data resource e via accessors of type t. This type of transformations can be performed on each instance with a complex type, since such an instance *must* have at least one accessor. If the last accessor of an instance is removed then ϱ_u is said to destruct the instance. The existence of transformations with semantics as specified is enforced by the following axiom:

Axiom 32 (Existence of ϱ_u) $u \in \mathsf{Act}(t) \;\Rightarrow\; \exists_T\,[t \xrightarrow{T} t \;\;\wedge\;\; \overrightarrow{T} = \varrho_u]$

4.3.2 Deep transformations

The second class of complex transformations does a little more work; they are *deep* transformations in the sense that instances at the base of a complex type are transformed. For example, all *Doc* files in a *Zip* archive may be transformed to *Pdf*. These transformations:

- take a data resource with a complex type as input as well as a transformation and an accessor type,
- transform the data elements at the base of accessors of the specified accessor type,
- leave other accessors (and their bases) untouched.

We use δ to denote the semantics of such a transformation. Let e be a complex instance and u the type of one of its accessors. Furthermore, let T be a transformation. Then, the transformation identified by $\delta_{u:T}(e)$ transforms the data elements at the base of all accessors with type u, leaving the other accessors and their bases untouched:

$$a \in \pi(u) \;\;\Rightarrow\;\; (e \overset{a}{\rightsquigarrow} d \;\;\Leftrightarrow\;\; \delta_{u:T}(e) \overset{a}{\rightsquigarrow} \overrightarrow{T}(e))$$

$$a \notin \pi(u) \;\;\Rightarrow\;\; (e \overset{a}{\rightsquigarrow} d \;\;\Leftrightarrow\;\; \delta_{u:T}(e) \overset{a}{\rightsquigarrow} d)$$

Transformations with such semantics thus loop over the accessors of the specified type and execute the specified transformation on their bases. The transformations on these bases need not have an effect, for example because the data elements at the base do not match the input type of the transformation.

Axiom 33 (Existence of $\delta_{u:T}$) $u \in \mathsf{Act}(t_1) \;\Rightarrow\; \exists_T\,[t \xrightarrow{T} t \;\;\wedge\;\; \overrightarrow{T} = \delta_{u:T}]$

4.3.3 Transformations for type casts

So far we have shown different kinds of transformations between (instances of) types. Even more, we have introduced the notion of subtyping in Section 3.4.5. In most programming languages (some) *type casts* are executed implicitly by the compiler. For example, the integer 13 is automatically cast to the float 13.0 in a context where a float is required. We explicitly name and call type casts in our framework for transformations. As such, type-casting transformations are considered to be first class members of $\mathcal{TR}$. Using them we can *lift* an instance to its supertype; i.e., consider an instance using the interface of its supertype. The point is that the data resource itself does not change. The semantics of such a transformation are, thus, void:

$$\iota_t(e) = e$$

These transformations are applicable for all instances of all types as long as the types have a supertype:

Axiom 34 (Existence of ι_t)

$$s \,\underline{\mathsf{SubOf}}\, t \;\;\Rightarrow\;\; \exists_T \, [\overrightarrow{T} = \iota_t]$$

4.3.4 Example

In this section we will present a small example that illustrates the composition of transformations, transformations that remove an accessor (Axiom 32), and deep transformations (Axiom 33). Consider the following situation. Let `backup.zip` be a *Zip* archive. Two files (`report.doc` and `letter.doc`) form the *Payload* of this archive. Also, a *Comment* ("Backup") and a *Password* ("Secret") are associated to it. In other words:

$$\begin{array}{rcl}
\tau(\texttt{backup.zip}) & = & \mathit{Zip} \\
\mathsf{Act}(\texttt{backup.zip}) & = & \{\mathit{Payload}, \mathit{Comment}, \mathit{Password}\} \\
\texttt{backup.zip} \ltimes \mathit{Payload} & = & \{\texttt{report.doc}, \texttt{letter.doc}\} \\
\texttt{backup.zip} \ltimes \mathit{Comment} & = & \text{“Backup”} \\
\texttt{backup.zip} \ltimes \mathit{Password} & = & \text{“Secret”}
\end{array}$$

Now, let T_1 be a transformation with $\mathsf{Input}(T_1) = \mathit{Doc}$ and $\mathsf{Output}(T_1) = \mathit{Pdf}$. Then, $\delta_{\mathit{Payload}:T_1}$ is a transformation that transforms the documents in the payload of any *Zip* file to *Pdf*. Also, let $\varrho_{\mathit{Password}}$ be a transformation that removes the password of a *Zip* archive. If we want to transform `backup.zip` such that the documents in the payload are transformed to *Pdf* and its password is removed then we can achieve this by composing a transformation as follows:

$$\begin{array}{l}
T = \delta_{\mathit{Payload}:T_1} \circ \varrho_{\mathit{Password}} \\
\overrightarrow{T}(\texttt{backup.zip}) = \texttt{new.zip}
\end{array}$$

The result of this transformation is a new archive *new.zip* such that:

$$\begin{aligned}
\tau(\texttt{new.zip}) &= \mathit{Zip}\\
\mathsf{Act}(\texttt{new.zip}) &= \{\mathit{Payload}, \mathit{Comment}\}\\
\texttt{new.zip} \ltimes \mathit{Payload} &= \{\texttt{report.pdf}, \texttt{letter.pdf}\}\\
\texttt{new.zip} \ltimes \mathit{Comment} &= \text{“Backup”}
\end{aligned}$$

4.4 Effects of transformations

Transformations can be applied to data resources. The key point of transformations is that the data resources on which they are applied somehow change. For example, the data resource type or resolution of the input instance is altered. In the previous section we have presented the details of the transformation framework. We now shift the focus to the effects of transformations. The study of specific properties of transformations has been conducted in other contexts as well. As an example we mention properties concerning clustering (e.g., [SWL$^+$02]) and performance (e.g., [RCDT01]).

The remainder of this section is organized as follows. First we will extend our formalism with the notion of properties of data resources relying heavily on the language presented in Section 3.5. This will allow us to assert whether a data resource has a certain property (or not). The next step is to study the possible classes of effects by transformations on properties of data resources both at the type level and the instance level. This is the topic of Section 4.4.2 and 4.4.3. Finally we will go into detail on the relation between transformations, resource space and resource base in Section 4.4.4.

4.4.1 Properties

The word *property* has many different meanings in different fields. For example, “a thing belonging to someone”, “a building and the land belonging to it” but also “a characteristic of something”. In this section we are primarily interested in the latter meaning of the word. More specifically, we wish to express the properties (in the sense of characteristics) that data resources may have.

We consider any predicate over Σ^{τ} to be a property, as long as the first argument of the predicate concerns a data resource. For example, the data resource type that a data resource may have is a property, as well as the fact that it may have hyperlinks. We will present more formal examples shortly. To model properties formally, we will presume a property to be represented in general as $\varphi(e, W)$ where e is a data resource and W is a sequence of (zero or more) resource space elements or types. The intended meaning of $\varphi(e, W)$ is that e has the property φ with resource space elements W as evidence. In other words, W is the support for the assertion that e has property φ. Note that W must be a sequence, as opposed to an unordered set, since φ resembles the notion of a *predicate* in which the order matters.

We can express these predicates either in terms of our formalism, or using the language as introduced in Section 3.5. Consider the following examples in which we firstly give the intuition behind the property, then present it in terms of our formalism and finally present it in terms of the language from Section 3.5:

Example 4.4.1

- *Let φ_p denote the property that a data resource is a Pdf file:*

$$\begin{aligned}\varphi_p(e,[\,]) &\triangleq e \,\mathsf{HasType}\, \mathit{Pdf}\\ \varphi_p(e,[\,]) &\triangleq \text{e has Type “Pdf”}\end{aligned}$$

- *Let φ_t denote the property that a data resource has a certain type:*

$$\begin{aligned}\varphi_t(e,[t]) &\triangleq e \,\mathsf{HasType}\, t\\ \varphi_t(e,[t]) &\triangleq \text{e has Type “t”}\end{aligned}$$

- *As a more complex example, consider the property where a data resource is of a certain representation type (i.e., it implements an information resourse via a certain representation type) and also is of a certain encoding:*

$$\varphi_c(e,[x,y]) \triangleq$$
$$\exists_{r\in\mathcal{RP},a\in\mathcal{AT}}\,[\mathsf{DRes}(r)=e \;\wedge\; r\,\mathsf{HasType}\,x \;\wedge\; e \overset{a}{\rightsquigarrow} y \;\wedge\; a\,\mathsf{HasType}\,\mathit{Encoding}]$$
$$\varphi_c(e,[x,y]) \triangleq$$
e (involved in Representation of Type “x” AND-ALSO
  involved in Attribution (of type “Encoding” AND-ALSO having Data Value “y”))

Given the fact that the latter expression tends to be more readable we propose to use these for expressing properties. In the remainder, let Φ denote the language in which properties can be expressed.

Given a resource base, a specific data resource e and a property φ, one may wonder whether this property holds for e or not. An equally interesting problem is finding the support for data resource e having some property φ. As an example of the latter consider the expression $\varphi_c(e,[\mathit{Keyword\text{-}list}, \mathit{UTF\text{-}8}])$ which uses the definition of φ_c from Example 4.4.1. To be able to evaluate this truth assignment of a φ for a given instance e we use the property support function $\Gamma : \mathcal{DR}\times\Phi \to \wp(\mathcal{RE}^+)$ with the intended meaning:

$$\Gamma(e,\varphi) \triangleq \{W \mid \varphi(e,W)\}$$

In other words, $\Gamma(e,\varphi)$ returns the set of sequences of resource space elements W for which φ is true, given a data resource e. If $\Gamma(e,\varphi)$ returns $\emptyset$ then $\varphi(e,W)$ apparently does not hold. Continuing the previous example where we defined φ_c, if:

$$\Gamma(e,\varphi_c) \supset \{[\mathit{Keyword\text{-}list}, \mathit{UTF\text{-}8}]\}$$

then the support for the assertion that e has property φ_c is the given sequence of resource space elements. This implies that data resource e indeed is a *Keyword-list* in the *UTF-8* encoding. We end this discussion of properties, their definitions and truth assignments with two small examples of how they can be used in practice:

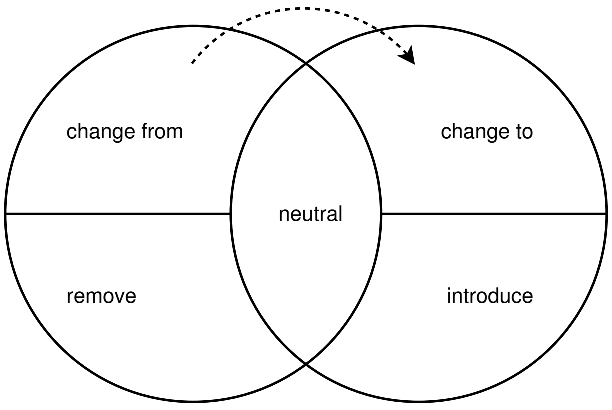


Figure 4.1: Effect classes of transformations

- Let $\varphi_t(e, [t]) \triangleq$ e has Type t denote the typing property, and e a data resource with $\tau(e) = \{t_1, t_2\}$. Then, the support for e having this property is provided by these two types.

- Let $\varphi_t(e, [\,]) \triangleq$ e has Type "Pdf" denote the property *is of type Pdf*, and let e be a data resource with $Pdf \in \tau(e)$. Then, the support for e having this property is the empty sequence which implies that e indeed has this property. Similarly, let f be a data resource with $Pdf \notin \tau(f)$. In this case, the support for f having this property is void which implies that this property for f is not supported.

4.4.2 Classes of effects

Now that we have properly introduced the notion of properties we can shift our attention to the possible effects that a transformation may have on properties of a data resource. We discern four classes of effects when a single data resource is concerned. These effect classes are illustrated in Figure 4.1. The left circle depicts the input type of a transformation and the right circle depicts its output type. Some properties may be left unchanged, which is depicted by the intersection of both circles. Also, some properties may be changed, which is depicted by the *from* and *to* section of the circles. Finally, properties may be removed or introduced which is depicted by the remaining sections. Observe that this line of reasoning is for the *instance level*. To see why this matters, consider a transformation that adds a version number to a data resource if it is not yet present. This transformation will *introduce* the property for certain instances, and be *neutral* with respect to other instances. Therefore, let $\mathcal{EC}_i = \{neutral, alter, remove, introduce\}$ be the set of effect classes at the instance level.

At the typing level, the reasoning is slightly more complex. In this case, the effect of a transformation on instances of a certain data resource type with a certain property is studied. In other words, the conclusion that a transformation has a certain effect at the instance level must be generalized to the type level. In the next subsection we will show how this can be done. The effect classes at the type level are $\mathcal{EC}_t = \{\textit{neutral}, \textit{remove}, \textit{introduce}, \textit{hybride}\}$.

4.4.3 Effects: instance and type level

Using the Γ relation, it is straightforward to find out (i.e., learn) the effect class of a transformation with regard to a specific property. Let $\mathsf{Effect} : \mathcal{TR} \times \mathcal{DR} \times \Phi \rightarrow \mathcal{EC}_i$ be the function that finds the effect of a transformation $T \in \mathcal{TR}$ on a data resource $e \in \mathcal{DR}$ with respect to a property $\varphi \in \Phi$. The actual effect can established by comparing the support of a property before and after transformations:

- If the support sets are equal, then for this (input) instance, the transformation is neutral with respect to this specific φ.
- If the input set is a subset of the output set, then the transformation, for this (input) instance, apparently is introducing with respect to this property.
- Similarly, if the output set is a subset of the input set, then the transformation is removing for this instance with respect to this specific property.
- If neither of the above applies then, for this instance, the transformation is said to be altering with regard to this specific property.

The above can be formalized as follows. The effect of a transformation T on data resource e with respect to property φ is:

$$
\begin{array}{l}
\mathsf{Effect}(T, e, \varphi) \triangleq \\
\quad \begin{array}{llll}
\textbf{if} & \Gamma(e,\varphi) = \Gamma(\overrightarrow{T}(e), \varphi) & \textbf{then} & \textit{neutral} \\
\textbf{elif} & \Gamma(e,\varphi) \subset \Gamma(\overrightarrow{T}(e), \varphi) & \textbf{then} & \textit{introduce} \\
\textbf{elif} & \Gamma(e,\varphi) \supset \Gamma(\overrightarrow{T}(e), \varphi) & \textbf{then} & \textit{remove} \\
\textbf{else} & \textit{alter} & &
\end{array}
\end{array}
$$

A similar line of reasoning explains how the definition of Effect may be generalized to the typing level such that $\mathsf{Effect} : \mathcal{TR} \times \mathcal{DR}_\tau \times \Phi \rightarrow \mathcal{EC}_t$. It may seem that $\mathcal{DR}_\tau$ is superfluous. However, this is not the case since transformations are defined for all data resource types. $\mathsf{Effect}(T, t, \varphi)$ is *neutral* for $t \notin \mathsf{Input}(T)$.

At the typing level, the effect that a transformation has on a certain property must be analyzed for *all instances* of a certain type

- If a transformation is neutral with regard to a property for all instances of a given data resource type then, at the typing level, we conclude that the transformation is neutral with regard to this specific property.

- It seems obvious that, at the type level, a transformation is removing for a given property if the transformation is removing for every instance of this type. A simple example shows that the situation is slightly more complex. Consider the property characterized by the expression $\varphi(e, [\,]) =$ e has Relation of Type "hyperlink" and a transformation T. Furthermore, let e be an instance that, indeed, has hyperlinks and is of the proper type. Then, $\mathsf{Effect}(T, e, \varphi) = \mathit{remove}$. If the transformed instance $\overrightarrow{T}(e) = e'$ is transformed again, using the same transformation, then $\mathsf{Effect}(T, e', \varphi) =$ *neutral* because e' did not have any hyperlinks in the first place.

  Therefore, the rule must be: if a transformation is removing with regard to a property for at least one instance and neutral for all others, then, at the typing level the transformation is said to be removing with respect to this specific property..

- For similar reasons, if a transformation is introducing with regard to a property for at least one instance and neutral for all others, then at the typing level the transformation is said to be *introducing* for this specific property.

- Again, it may seem that at the typing level a transformation is altering with regard to a property if it is altering for all instances of this type. However, this is not the case since other situations may occur also. For example: a transformation may be introducing for one instance and altering for another. This can occur, for example, when a transformation sets the version attribute to a specific value regardless of the fact that a data resource already had a version attribute. If it did, then the transformation is likely to be altering for this property. If it did not, then the transformation would be introducing. In this case, we are indecisive about the effect that this transformation has and assert that it is *hybride* at the typing level.

Summarizing, the effect of a transformation T with regard to a property φ considered at the *type level* is the following:

$$\begin{array}{l}
\mathsf{Effect}(T, t, \varphi) \triangleq \\
\quad \textbf{if} \quad \forall_{e \in \pi(t)} [\Gamma(e, \varphi) = \Gamma(\overrightarrow{T}(e), \varphi)] \quad \textbf{then} \quad \mathit{neutral} \\
\quad \textbf{elif} \quad \forall_{e \in \pi(t)} [\Gamma(e, \varphi) \subseteq \Gamma(\overrightarrow{T}(e), \varphi)] \quad \textbf{then} \quad \mathit{introduce} \\
\quad \textbf{elif} \quad \forall_{e \in \pi(t)} [\Gamma(e, \varphi) \supseteq \Gamma(\overrightarrow{T}(e), \varphi)] \quad \textbf{then} \quad \mathit{remove} \\
\quad \textbf{else} \quad \mathit{hybride}
\end{array}$$

4.4.4 Transformations, resource space, and resource base

In this section we will discuss how the above can be applied to the resource space and resource base as described in Chapter 3. In this context, the key distinction between them is that in the resource base we can observe the effect of a transformation on a specific instance, whereas in the resource space we can observe the total effect a transformation may have.

We will start at the resource base level and then generalize to the resource space level. Recall that a population can be expressed in terms of the signature of typed resource base as presented on page 52. Consider the following population, which exemplifies a resource base where a person named John Doe has a web page in *Html* format, where the current version of the web page is 2.4. Furthermore, this web page does not have any links to other sites:

$$
\begin{array}{lcl}
\mathcal{IR} & = & \{\text{John Doe}\} \\
\mathcal{RP} & = & \{r\} \\
\mathcal{DR} & = & \{\texttt{john-doe.html}\} \\
\mathcal{RL} & = & \emptyset \\
\mathcal{AT} & = & \{a\} \\
\mathcal{DV} & = & \{2.4\} \\
\mathcal{AC} & = & \emptyset \\
\mathsf{IRes} & = & \{\langle r, \text{John Doe}\rangle\} \\
\mathsf{DRes} & = & \{\langle r, \texttt{john-doe.html}\rangle\} \\
\mathsf{Src} & = & \{\langle a, \texttt{john-doe.html}\rangle\} \\
\mathsf{Dst} & = & \{\langle a, 2.4\rangle\} \\
\mathcal{TP} & = & \{\textit{Webpage-of}, \textit{Html}, \textit{Version}\} \\
\mathsf{HasType} & = & \{\langle r, \textit{Webpage-of}\rangle, \langle \texttt{john-doe.html}, \textit{Html}\rangle, \langle a, \textit{Version}\rangle\}
\end{array}
$$

The effect that a specific transformation has on *this* particular instance may be measured by transforming the instance and comparing the two signatures. For example, let $T \in \mathcal{TR}$ and $\mathsf{Input}(T) = \textit{Html}$, $\mathsf{Output}(T) = \textit{Ascii}$ such that $\overrightarrow{T}(\texttt{john-doe.html}) = \texttt{john-doe.txt}$. The following signature denotes this new data resource and its properties:

$$
\begin{array}{lcl}
\mathcal{IR} & = & \{\text{John Doe}\} \\
\mathcal{RP} & = & \{r\} \\
\mathcal{DR} & = & \{\texttt{john-doe.txt}\} \\
\mathcal{RL} & = & \emptyset \\
\mathcal{AT} & = & \{a\} \\
\mathcal{DV} & = & \{2.4\} \\
\mathcal{AC} & = & \emptyset \\
\mathsf{IRes} & = & \{\langle r, \text{John Doe}\rangle\} \\
\mathsf{DRes} & = & \{\langle r, \texttt{john-doe.txt}\rangle\} \\
\mathsf{Src} & = & \{\langle a, \texttt{john-doe.txt}\rangle\} \\
\mathsf{Dst} & = & \{\langle a, 2.4\rangle\} \\
\mathcal{TP} & = & \{\textit{Document-about}, \textit{Ascii}, \textit{Version}\} \\
\mathsf{HasType} & = & \{\langle r, \textit{Document-about}\rangle, \langle \texttt{john-doe.txt}, \textit{Ascii}\rangle, \langle a, \textit{Version}\rangle\}
\end{array}
$$

The effects of the actual transformation can now be studied by comparing these two signatures. More specifically, the effects can be found by computing $\mathsf{Effect}(T, \texttt{john-doe.html}, \varphi)$ for different φ:

- The input and output instance have different data resource types. More specifically, we transform an instance from type *Html* to type *Ascii*. If $\varphi(e,[t]) \triangleq \textsf{e has Type t}$ then $\textsf{Effect}(T, \texttt{john-doe.html}, \varphi) = \mathit{alter}$.

- Conform Axiom 28, both input instance and output instance of the transformation are attached to the same information resource; they have the same aboutness. However, their representation types differ. The relevant property in this respect is given by:

  $$\varphi(e,[\,]) \triangleq \textsf{e involved in Representation of Type t}$$

  Since the support for this property is different for the two data resources we can conclude that $\textsf{Effect}(T, \texttt{john-doe.html}, \varphi) = \mathit{alter}$.

- Both the input instance and output instance have an attribute of type *Version*. The attributed value is 2.4 in both cases. Hence the following property:

  $$\begin{aligned}\varphi(e,[v]) \triangleq\ & \textsf{e involved in Attribution} \\ & \textsf{(of Type ``Version'' AND-ALSO with Data Value v)}\end{aligned}$$

  It is easy to observe from the above signatures that the support for this property is the same for both data resources. Therefore, we know that $\textsf{Effect}(T, \texttt{john-doe.html}, \varphi) = \mathit{neutral}$.

Instances in the resource *space*, by definition, reflect the full extent of a given data resource type in the sense that every possible property of a type has an instance. The total effect of a transformation can be derived by locating the instance in resource space that has every possible property (that is compatible with its type), applying a transformation on this instance and comparing the two signatures.

4.5 Towards implementation

So far the focus of this chapter has been on the conceptual level in the sense that we have presented a fully formalized reference architecture for transformations in the world of information supply. In this section we will shift the focus to the implementation level which results in many interesting problems (an overview of these different levels in the context of databases can be found in e.g., [PGPR02]). In line with the overall topic of this dissertation we will focus on (aspects related to the) implementation of our reference architecture in the context of searching on the Web.

The first question that arises when considering transformations in a practical setting on the information market is: where do transformations come into view? Who will provide transformations or transformational services? Who executes the transformations? In our reference architecture we have explained how transformations can be value adding. Therefore it seems to be the case that the transformation task should be delegated to brokers on the information market (Section 2.2).

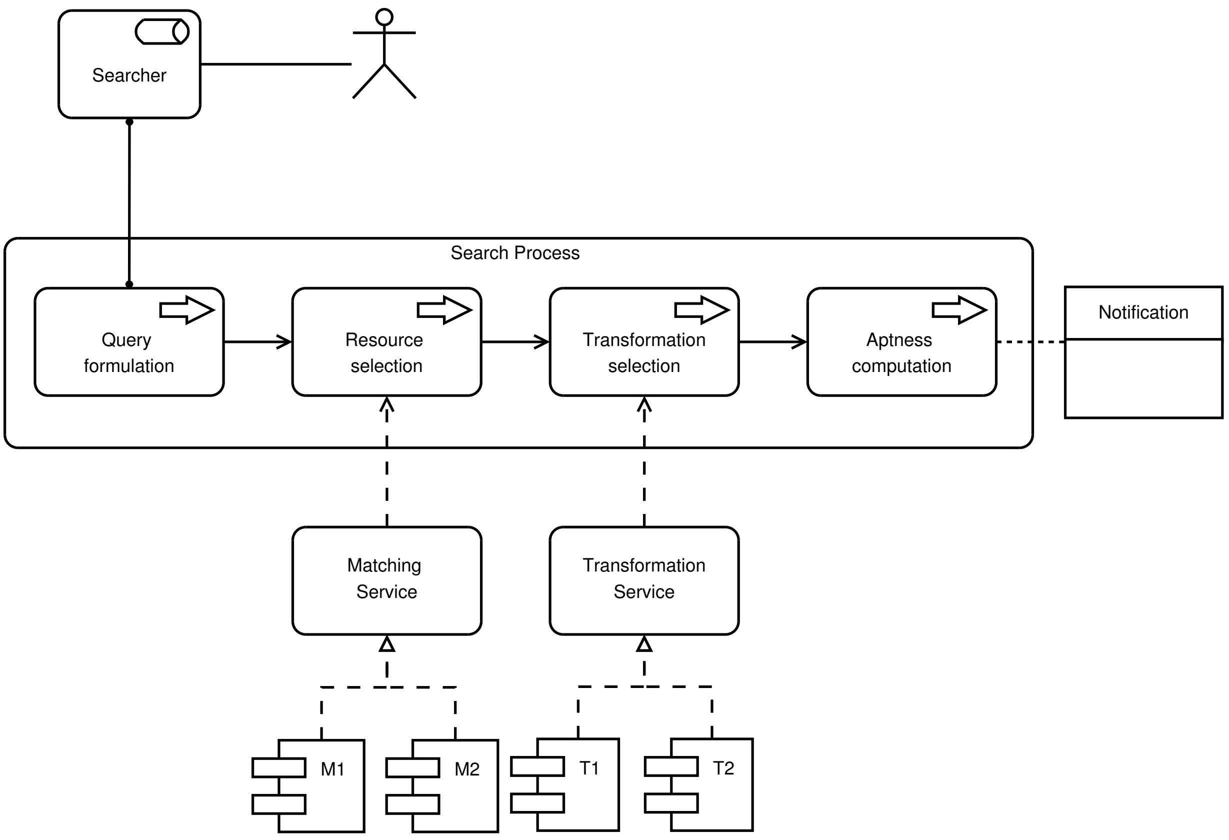


Figure 4.2: Reference model for the search process

In terms of [Ull89], the information landscape can be seen as a large *extensional* database. Use of transformations yields a practically infinite *intensional* database. The extensional web thus consists of the resources that are directly available, whereas the intensional web also includes the resources that can be derived from the former resources using transformations. In an ideal situation, searchers would be able to search the intensional web. This is, however, not feasible. Therefore we adopt another strategy based on the idea of push-down selection as used in database query optimization.

In query optimization this means that (expensive) join operations are executed *after* selection, which is more efficient than performing the join first and the selection later. In our approach it means that we firstly select those resources that are topically relevant. After this step has been completed we select and execute transformations and as such we rely heavily on Axiom 28 which states that transformations do not alter the aboutness of data resources. Finally the data resources must be re-ranked according to their aptness with respect to the information need by the searcher (aptness computations are discussed in Chapter 6). This is illustrated in Figure 4.2 which uses the ArchiMate notation for enterprise modeling (See e.g., [Lan05] for details on this language). The figure shows that the *searcher* role is performed by some actor with a specific information need. Query formulation is a collaboration between the actor and the system. After the query has been formulated, resource selection takes place. Our architecture is such that we can use different components ($M1$ and $M2$ as long as we can tap into their matching service). This also holds for the next step: transformation selection. Last but not least, aptness (which is, for the time being, defined as a generic measure for usefulness) must be computed and the results are presented to the actor performing the searcher role.

Topics such as user interface design and (support for) query formulation are, unfortunately, beyond the scope of this dissertation. The provider (player on the information market) of the matching service can be selected dynamically depending on the application domain. Our goal is not to devise yet another application for topical matching. Instead we'd rather tap into the service offered by e.g., *Google*. In the remainder of this section we discuss some of the issues pertaining to the transformation service. More specifically, we briefly discuss how effects of transformations can be learned (Section 4.5.1) and describe the problem of transformation selection (Section 4.5.2).

4.5.1 Learning effects

When building an application that actually uses our reference architecture for transformations, it is important to gather as much information about the transformations that are actually used. In many cases pre-existing tools. Since we consider transformations as "black boxes" (that is, we do not study them at the source-code level), we have to somehow *learn* their effects. The mechanisms described in Section 4.4.3 can be used to this end:

- Initially, it is assumed that a transformation is *neutral* with respect to every property; similar to the notion of being innocent until proven otherwise.

- After a transformation is performed on an instance (from resource base), the properties of the input instance and the output instance are compared to study the effect of the transformation:

  - We may discover a new property of a type. For example: before the transformation was executed we did not have a single instance of the *Pdf* type with a price, but after the transformation is finished we do. This implies that we have to take this additional knowledge into account the next time we compose a transformation involving the *Pdf* type.
  - We may discover that a transformation is not neutral with regard to some property. For example, it may *alter*, *remove*, or *introduce* certain properties. The transformation from *Html* to *Postscript* may, for example, remove all *Hyperlinks*.

The above implies that the performance (in terms of accuracy of transformation selection, which is the topic of Section 4.5.2) will be poor initially. Even more, it is likely to improve as more and more transformations are actually executed.

4.5.2 Transformation selection

The goal of this section is to present several issues related to a transformation selection algorithm. The goal of such an algorithm is to select one or more transformations that somehow improve data resources with respect to the wishes of a searcher. In other words, the transformation selection algorithm tries to select those transformations that increase the aptness of a data resource. As such, we now focus on increasing the aptness of data resources under the following strict assumptions:

- Everything that we need to know for aptness computations can be expressed in terms of our property language Φ.
- The entire information need is captured by the query as specified by the searcher.
- The desirability of properties (i.e., which properties must be supported, which properties must not be supported) as well as a prioritarization are included in the query as well.

As such, the 'aptness computations' that we propose here are somewhat limited. In Chapter 6 we will go into more detail on aptness (computations) and the impact on transformation selection.

At a high level of abstraction, the transformation selection scheme boils down to comparing the properties that a data resource is expected to have after a transformation with the properties as desired by the searcher. Therefore, the first step in this process is to determine the expected support for properties of a data resource after it has been transformed (possibly by a composed transformation).

Recall that the support for a property, given a data resource, is a sequence of resource space elements. Ergo, for a given data resource the support for every possible property is given by the function $\Psi : \mathcal{DR} \rightarrow \wp(\Phi \times \wp(\mathcal{RE}^+))$ with the intended meaning:

$$\Psi(e)(\varphi) \triangleq \Gamma(e, \varphi)$$

In other words, Ψ assigns to each data resource its property support. The following example illustrates the use of this relation.

Example 4.5.1 *Let Φ be the language for properties such that there are exactly two properties φ_1 and φ_2. Furthermore, let e be a data resource of type t. The support for the first property and e is the resource space element w_1 but also the element w_2. The support for the second property is the sequence of resource space elements $[w_3, w_4]$. Then, we know that:*

$$\Psi(e) = \{\langle\varphi_1, \{[w_1], [w_2]\}\rangle, \langle\varphi_2, \{[w_3, w_4]\}\rangle\}$$

This shorthand notation allows us to easily decide on the effects of a transformation by comparing the support for properties of a data resource with the support for these properties of the transformed data resource. We now continue the above example:

Example 4.5.2 *Let T be a transformation with* $\mathsf{Input}(T) = t$ *such that*

$$\Psi(\overrightarrow{T}(e)) = \{\langle\varphi_1, \{[w_1]\}\rangle, \langle\varphi_2, \{[w_3, w_5]\}\rangle\}$$

Then we can conclude that $\mathsf{Effect}(T, e, \varphi_1) = \textit{removing}$ *and that* $\mathsf{Effect}(T, e, \varphi_2) = \textit{altering}$.

Combined with the scheme for learning effects presented in Section 4.4.3, Ψ provides the machinery for estimating the expected support for properties of a data resource after it is transformed.

The second step is to compare the expected support for properties of the transformed data resource to the wishes of the user with regard to these properties. This can be modeled in several ways. For example, one may choose to model user preference using acceptable ranges for property support, or use a lattice-like structure to express the releative importance of properties. We have chosen to model these wishes as follows. Let $\mathcal{U} : \Phi \to \mathbb{R}$ denote a preference assignment (as a real number) with regard to properties. A positive assignment indicates that the property must be present, whereas a negative assignment indicates that the property must not be present. A zero-assignment denotes indifference. For our running example:

$$\mathcal{U} = \{\langle\varphi_1, 10\rangle, \langle\varphi_2, -5\rangle\}$$

denotes the preference where φ_1 must be present and φ_2 may not be present. Even more, the magnitude of the preference assignment is such that the searcher finds his preference with respect to φ_1 twice as important as his preference with respect to φ_2. This step is completed by comparing the support for properties of every possible transformation to $\mathcal{U}$. To illustrate how this works we end this section with an example of an aptness calculation under the restrictions we mentioned above.

Example 4.5.3 *Let $e, f \in \mathcal{DR}$ be data resources. Furthermore, $Doc, Pdf \in \mathcal{DR}_\tau$ are data resource types. Also, $r_1, r_2 \in \mathcal{RL}$ are relations and $T \in \mathcal{TR}$ a transformation such that $e \overset{r_1}{\leadsto} f$ and $\overrightarrow{T}(e) \overset{r_2}{\leadsto} r$. These two relations are typed. Therefore, let $Hyperlink, Reference \in \mathcal{RL}_\tau$. The language for properties is such that:*

- *Typing property:*
  $\varphi_t(e, [t]) \triangleq \mathsf{e\ HasType\ t}$
- *More specific typing property that tests for being a Doc file:*
  $\varphi_s(e, [\,]) \triangleq \mathsf{e\ HasType\ “Doc”}$
- *Property for having (outgoing) hyperlinks:*
  $\varphi_r(e, [r]) \triangleq \mathsf{e\ being\ source\ of\ Relation\ of\ Type\ “Hyperlink”}$

Given the support for properties, as well as a vector of preference assignments $\mathcal{U}$ we can now come to a decision with regard to the aptness of e and $\overrightarrow{T}(e)$ for this $\mathcal{U}$:

- $\Psi(e) = \{\langle\varphi_t, \{[Doc]\}\rangle, \langle\varphi_s, \{[\,]\}\rangle, \langle\varphi_r, \{[\,]\}\rangle\}$
  In other words, e is of type Doc and indeed has an outgoing relation of type Hyperlink.

- $\Psi(\overrightarrow{T}(e)) = \{\langle\varphi_t, \{[Pdf]\}\rangle, \langle\varphi_s, \{[\emptyset]\}\rangle, \langle\varphi_r, \{[\emptyset]\}\rangle\}$
  In other words, $\overrightarrow{T}(e)$ is of type Pdf and has no outgoing hyperlinks (instead, the relation can, for example, be of type Reference).

- $\mathcal{U} = \{\langle\varphi_t, 0\rangle, \langle\varphi_s, -10\rangle, \langle\varphi_r, 5\rangle\}$
  In other words, the user refuses data resources of type Doc and would prefer them to have outgoing hyperlinks.

The aptness computation is now relatively straightforward. The aptness of e $= -10 + 5 = -5$. *Since* e *is of type Doc, it receives a penalty of* -10 *and since it has outgoing hyperlinks it receives a positive score of* 5. *Similarly for* $\overrightarrow{T}(e)$ *the aptness would be* $10 - 5 = 5$. *Apparently, the transformation increases the aptness for this specific searcher.*

4.5.3 An abstract view on transformation selection

In our view, the transformation selection problem can be seen as a variation on the shortest path problem as presented in [Dij59]. Firstly, one must realize that transformations bascially form a labelled directed graph where the nodes represent data resource types and the edges represent transformations. An edge thus represents a tranformation from the input type to the output type. The problem can now be formulated as follows:

- We start with a data resource e of which we know the property support $\Psi(e)$.
- In the retrieval setting, the query specifies a (virtual) data resource e' of which we know the desired property support $\Psi(e')$
- Because typing is mandatory (Axiom 18) so we know the start type $t \in \tau(e)$ and from $\Psi(e')$ we can derive a goal type t'
- The selection task is to find a transformation T such that $\Psi \circ \overrightarrow{T}(e) = \Psi(e')$

This can be considered a varation on Dijkstra's algorithm because we are mainly trying to find an optimal path through the transformation graph. However, the weights of the edges in the graph are not fixed, they differ from selection problem to selection problem. More specifically, the length of an edge is determined by the usefulness of the transformation in composing a transformation that generates e'. More specifically, it depends on the cost (i.e., runtime) of a transformation and the desirability of $\mathsf{Effect}(T, \varphi, \mathsf{Input}(T))$ which can be derived from $\Psi(e')$.

Similarly the selection problem can also be considered an A^* problem (see e.g., [HNR68, Wik06]) which is a best-first search algorithm for graphs. In this case the graph is much more complex. A possibility would be to create nodes for all possible combinations of properties. With n possible properties this would mean 2^n nodes in the graph. The edges still represent transformations. Both the input instance and the desired instance as specified by the query now are nodes in the graph. Even though the A^* algorithm is arguably the computationally most efficient alogirhm, finding an actual sollution in a large space (many different properties) may be computationally hard to say the least. Robert Helgesson[1] therefore suggested another strategy using a two-step approach:

> It might be good to consider the transformation between types and transformations within a type as two different things. So, in the first step you find a path between the data resource type and the target data resource type. Then you expand the target "cluster" and do another search within that. So you do two separate A^* searches.

Considering the transformation selection problem as a graph traversal problem, thus, provides interesting insights which should be taken into account when designing real tools. The larger the graph, the more efficient algorithms must be used.

4.6 Conclusion

The main goal of this chapter was to answer the question: how can (data resources in) information supply be meaningfully manipulated. As such, this chapter builds upon the formal framework for information supply as presented in Chapter 3.

We started this chapter with a discussion of basic properties of transformations and gradually progressed to more complex forms of transformations. The main idea behind our formalism is that transformations have an input type and an output type. Even more, the semantics of a transformation (what it actually does) can only be applied to instances (data resources) of the proper input type. Even more, the semantics of transformations can be combined to form composed or complex transformations.

We also studied the effects that transformations may have on properties of data resources. To this end we firstly introduced the notion of properties of data resources and explained how the effects of transformations on these properties can be learned. Knowledge about these effects can be used to reason about transformations. More specifically, it can be used to aid us in the problem of transformation selection.

Transformation selection, in the context of this dissertation, can be loosely defined as selecting those transformations that somehow increase the aptness of data resources, where aptness can be seen as a metric for usefulness (aptness computations are discussed in Chapter 6). The main assumption behind this selection scheme is that the usefulness of data resources depends on the properties of data resources.

Several issues have to be resolved before transformations can be used in a practical setting. Many of these issues pertain to compiling a database of transformations, learning their effects, deciding which properties one wants to take into account, and designing an algorithm for doing actual selection. In Chapter 5 we will present some small experiments related to these issues.

[1]personal communication, 2006

CHAPTER 5 Transformations in practice

Outline In this chapter we present two small experiments that we have conducted to test the applicability of our framework in practice. The scope of these experiments is somewhat limited in the sense that we do not rigorously test all aspects involved with using transformations in a practical setting. In a first experiment (see Section 5.1) we have experimented with two different algorithms for finding a path through the transformation graph. In the second experiment (see Section 5.2) we have experimented with a search system for scientific papers that actually uses our transformations.

5.1 Path finding

In this experiment we try to devise an algorithm that selects a suitable transformation path. At first sight it may seem sensible to do a depth-first exhaustive search and simply select the shortest path. This is, however, not a feasible solution when the transformation graph grows large. Even more, as soon as the graph is cyclic then we need to somehow make sure that the search process terminates at some point.

Our task is, thus, not to select all possible paths, but the acceptable paths. We will use a penalty-mechanism for this. The configuration of the penalty mechanism can be tweaked to formulate the desired properties of the transformation paths (for example, the algorithm can be tweaked to return exactly one path). We have devised a naive path finding algorithm and a somewhat more advanced method which will be presented subsequently. In the remainder of this Section we will use the transformation graph in Figure 5.1 as running example. This figure shows 10 types and 21 possible singleton transformations. We will search for transformations from Rtf to Ps.

In the simplest case we search for all possible paths from input type to output type (i.e., perform a depth first exhaustive search) with the only exception that we prevent a transformation being part of the transformation path twice. This prevents endless loop-

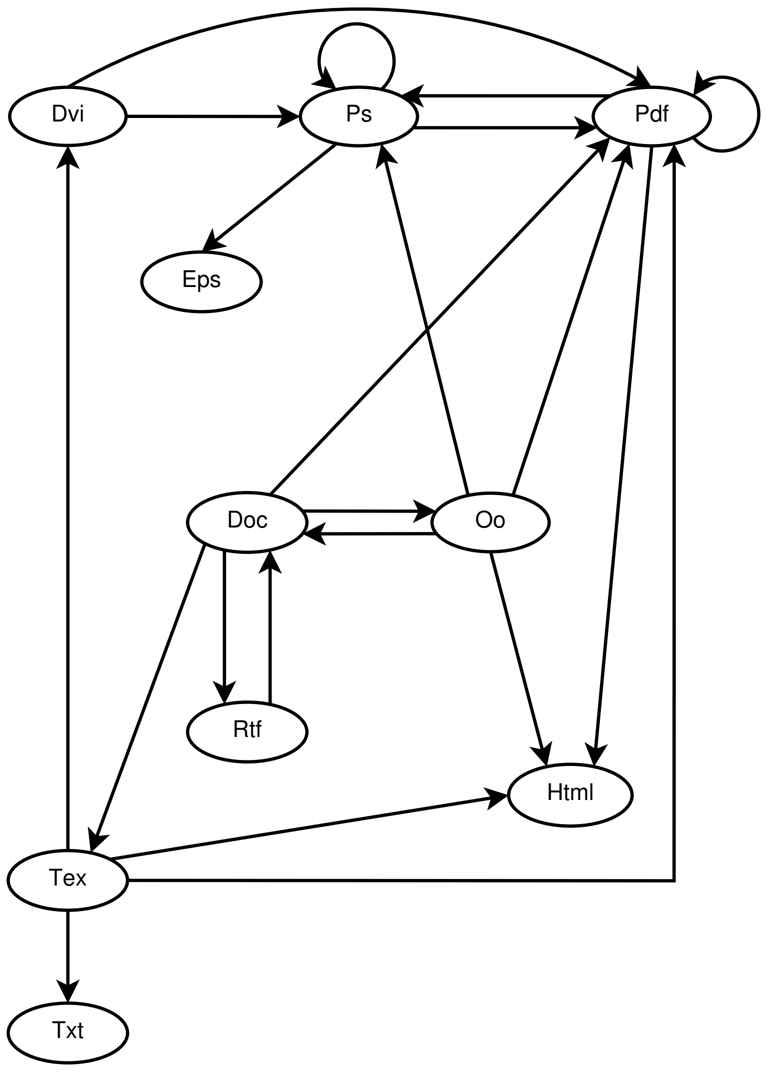


Figure 5.1: Example transformation graph

```
function getPath(from, to, currentPath)
begin
  // findTransformationsFrom finds all
  // transformations starting with input
  // type from
  candidates := findTransformationsFrom(from);
  results := new List();

  foreach transformation in candidates
  begin
    if transformation in currentPath then
      break;
    if transformation.resultType = to then
      results.append(currentPath + transformation)
    else
      results.append(getPath(transformation.resultType,
                  to, currentPath + transformation));
  end;

  return results;
end;
```

Figure 5.2: Pseudo code for path finder

ing and makes sure the selection algorithm terminates at some point. The steps of this algorithm are as follows:

1. Take the start type and take all transformations that have this type as its input type.

2. Loop over these transformations and check if this transformation has been performed already.

3. If the transformation has *not* been performed yet, check if the target type is reached with the current transformation. If it has been reached then we have found a transformation path. If it has not been reached, take the current output type and start with step 1 again to recursively find the target type.

This algorithm is illustrated in pseudo code in Figure 5.2. Performing this algorithm on the running example leads to the following transformation paths:

1	$Rtf \rightarrow Doc \rightarrow Oo \rightarrow Ps$
2	$Rtf \rightarrow Doc \rightarrow Oo \rightarrow Pdf \rightarrow Ps$
3	$Rtf \rightarrow Doc \rightarrow oo \rightarrow Pdf \rightarrow Pdf \rightarrow Ps$
4	$Rtf \rightarrow Doc \rightarrow Doc \rightarrow Tex \rightarrow Dvi \rightarrow Ps$
5	$Rtf \rightarrow Doc \rightarrow Tex \rightarrow Dvi \rightarrow Pdf \rightarrow Ps$
6	$Rtf \rightarrow Doc \rightarrow Tex \rightarrow Dvi \rightarrow Pdf \rightarrow Pdf \rightarrow Ps$
7	$Rtf \rightarrow Doc \rightarrow Tex \rightarrow Pdf \rightarrow Ps$
8	$Rtf \rightarrow Doc \rightarrow Tex \rightarrow Pdf \rightarrow Pdf \rightarrow Ps$
9	$Rtf \rightarrow Doc \rightarrow Pdf \rightarrow Ps$
10	$Rtf \rightarrow Doc \rightarrow Pdf \rightarrow Pdf \rightarrow Ps$

With all possible transformation paths known, it is straightforward to figure out what happens to properties during transformations. If the effect that each transformation has on a given property is known then this knowledge should be used: follow the path and, in each node, determine if the property still holds or not.

However, if the effects that some transformations have on a given property are unknown, the only way to be absolutely sure which path should be selected would be to perform every transformation path (and thus learning the effects on this property for future use too).

We consider the composition of transformations. A sequence of transformations may compose a new 'overall' transformation. This raises the question of transformation performance, since several different transformation sequences may transform a given input type into a given output type. Again, one could use a shortest path-like scheme. For a full treatment of web transformation performance as found in other areas of transformation (see e.g., database transformation in [RCDT01]). Note that in our shortest path view we do not necessarily require a single shortest path to be found. Rather, we aim at a reduction of the possible paths in order to yield a selected set of candidate transformation compositions. We have successfully exploited reduction in transformations in earlier projects, such as database transformation (see e.g., [BW92a]).

The approach we have described so far has some serious disadvantages. First of all, as the number of types and singleton transformations grow, the number of possible paths through the transformation graph is likely to explode. Determining all possible paths from a given input type to an output type at runtime will take an increasing amount of time. The situation is even worse if properties may be composed dynamically at runtime: after finding the possible paths, they must all be executed to determine what happens with the newly composed properties.

A similar problem exists in the world of (relational) databases: performing a join *before* doing a selection is computationally heavier than performing the join *after* doing the selection. Therefore, a push-down selection scheme should be adopted (See e.g. [Ull89]). Translated to our problem of walking through the transformation graph: determining which transformation paths are not feasible should be done as soon as possible as opposed to removing the unwanted paths after figuring out all possible paths. Simply put: figure out which paths are likely to be infeasible *while finding all possible paths through the graph.* As soon as it is likely that following a path will lead to no good, that path should be abandoned and a new one tried; i.e., break the current loop and go on with the recursive search. This will not only lessen the time it takes to perform the search but, hopefully, will lessen the number of paths that are found.

We have not yet elaborated on the criteria that should be used to estimate the likelihood that a path will not be feasible. To this end we propose to use a penalty-based approach:

- Short paths are likely to be better (for example: faster in terms of execution time) than long paths. Therefore, each step through the graph is penalized. This is particularly apparent when, for example, execution time plays a role: every step takes extra time (if, in a certain situation, the execution time does not play a role than this penalty can be set to 0). However, this is not the only reason. Transformations

```
function getPath(from, to, maxPenalty, currentPath)
begin
  // findTransformationsFrom finds all
  // transformations starting with input
  // type from
  candidates := findTransformationsFrom(from);
  results := new List();

  foreach transformation in candidates
  begin
    if ((transformation in currentPath)
        or (transformation.penalty > maxPenalty)) then
      break;
    if transformation.resultType = to then
      results.append(currentPath + transformation)
    else
      // penalty is a function that calculates the
      // penalty of this transforamtion
      results.append(getPath(transformation.resultType,
              to, maxPenalty - penalty(transformation),
              currentPath + transformation));
  end;

  return results;
end;
```

Figure 5.3: Pseudo code for penalty based path finder

may also reduce the "quality" of the input resource which can also be a reason to increase the penalty for this transformation.

- If a property must be retained during transformation, removing it along the way will result in a penalty. If the property is added along the way, this will result in a negative penalty. Similarly, if a property must be removed during transformation, adding it will lead to a penalty and removing it will lead to a negative penalty.

- As soon as the current penalty for a path surpasses a certain boundary then it is assumed that this path is likely to be not feasible, therefore, it will no longer be followed and a new path must be tried.

The pseudo code in Figure 5.3 shows the outline of this implementation and exemplifies the algorithm.

We extended the above mentioned example with penalties such that every transformation has a penalty of 0.1 because of execution time. However, the transformations *Dvi* $\rightarrow$ *Pdf*, *Pdf* $\rightarrow$ *Pdf*, *Oo* $\rightarrow$ *Ps* and *Oo* $\rightarrow$ *Pdf* have a penalty of 0.2 and *Tex* $\rightarrow$ *Pdf* has a penalty of 0.3 because these are presumed to be heavier in terms of computation. Also, we know that the transformations *Oo* $\rightarrow$ *Html* is *removing* with regard to a certain property φ and *Doc* $\rightarrow$ *Tex* is *introducing* for this same property. Since we wish to retain this property, the former transformation receives a negative penalty of 0.2 and the latter receives a positive penalty (bonus) of 0.2.

Running this algorithm and, thus, taking into account the above mentioned penalties results in the following paths:

number	**path**
1	$Rtf \rightarrow Doc \rightarrow Oo \rightarrow Ps$
2	$Rtf \rightarrow Doc \rightarrow Tex \rightarrow Dvi \rightarrow Ps$
3	$Rtf \rightarrow Doc \rightarrow Tex \rightarrow Dvi \rightarrow Pdf \rightarrow Ps$
4	$Rtf \rightarrow Doc \rightarrow Tex \rightarrow Pdf \rightarrow Ps$
5	$Rtf \rightarrow Doc \rightarrow Pdf \rightarrow Ps$

Selecting the "optimal" path from these transformations still needs to be done. It is tempting to simply select the path with the lowest penalty, but this may not always be the best path because the total effect that the transformation(path) has on the properties must be taken into account. For the above example, the penalties and effects are the following:

number	**penalty**	**effect**
1	0.4	*neutral*
2	0.2	*introducing*
3	0.4	*introducing*
4	0.4	*introducing*
5	0.3	*neutral*

If the effect that a composed transformation has on properties is taken into account, as well as the penalty this transformation receives then the second path is to be selected since:

1. It is introducing for a property that we wish to retain, so we're 100% sure that the property will hold after this transformation path is executed on any given input instance.

2. It has the lowest penalty.

Our running example has only a few types, properties and a few transformations. In practice, though, these numbers are likely to be much higher. An example of a typing framework is the MIME framework (See e.g. [BF92]). There are dozens of different MIME-types to represent textual documents, applications, audio, images, video, etcetera. Therefore we have also tested our algorithm against a larger transformation graph which is shown in Figure 5.4. For clarity, the names of the transformations have been omitted. Note that we did not include "transformations to self". For purposes of this experiment this does, at least conceptually, not make a difference. In this graph:

- There are 100 types.

- We're looking for a transformation from type t_{12} to type t_{89}.

- We assume the existence of three properties: p_1, p_2 and p_3. The output instance must have properties p_1 and p_2, but may not have property p_3.

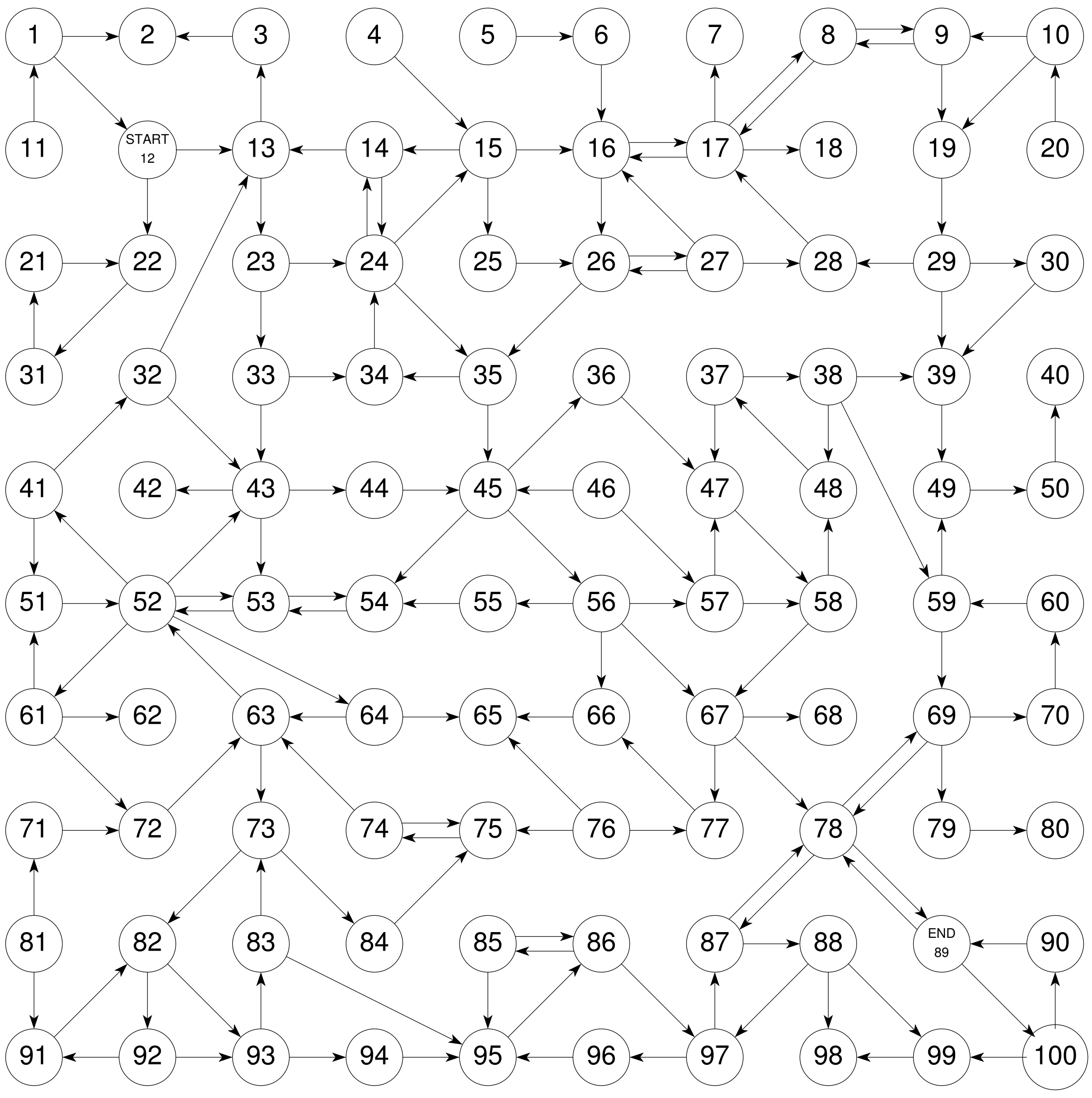


Figure 5.4: Larger example of a transformation graph

- There are 161 singleton transformations.
- Every singleton transformation will result in a penalty of 0.1.
- For 59 transformation-property combinations we know the effect (i.e., there are 59 statements in the form: Transformation t has effect e for property p), spread out over 46 transformations.
- If we don't know the effect of a transformation on a property, we will assume that it is neutral.
- The average penalty (either positive or negative) is 0.207.
- The maximum penalty that a transformation path may have is set to 2.5.

After running both the naive path finding algorithm and the more complex penalty based approach we observe the following:

- The path finder algorithm finds a total of 915 *possible* paths through the graph. Using the penalty based approach, this is reduced to 82 *acceptable* paths.
- The average length of a path for the path finder algorithm is approximately 27, whereas the average length in the penalty based approach is approximately 18.
- For this particular example, the penalty based approach is approximately 6 times faster than the naive algorithm.

The above suggests that, at least for this example, the penalty based algorithm performs better in terms of execution speed as well as in the number / length of the paths that it returns.

5.2 Publication searching

As a second experiment with transformations, we have implemented a prototype search system along the lines of the architecture presented in Figure 4.2. The goal of this experiment was to show the feasibility of integrating transformations in the search process. To this end we chose a domain of (relatively) low complexity: searching in a collection of scientific papers. To start with we have:

- A collection of 3016 scientific papers for which we have at least the bibliographic data available in *BiBTeX* format.
- For 2872 of these papers we have a *Pdf* version of the article available and for 119 papers we have a *Html* version available.
- For 135 of the papers we also have LaTeX sources available.

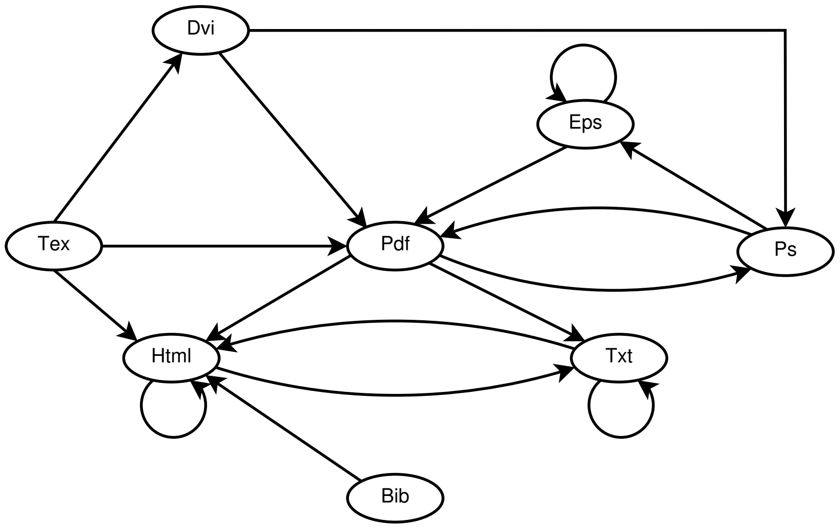


Figure 5.5: Transformation graph for the publication searching system

- In our prototype implementation we have 8 Data Resource Types (*Tex*, *Dvi*, *Pdf*, *Html*, *Bib*, *Ps*, *Eps*, and *Txt*) and 18 singleton transformations. This is illustrated in Figure 5.5.
- We distinguish between three Representation types: full-text, abstract, and bibliography.
- All transformations are *neutral* with respect to the property support for representationtypes except for the transformation *Txt* $\rightarrow$ *Txt* which is *introducing* for the abstract property, and *removing* for the full-text property.

When implementing the system, the first hurdle to be taken was indexing the collection such that we can both search for topics (i.e., the informational dimension) as for property support (i.e., the structural dimension). For example, in a query we want to be able to specify that the topic of the paper is to be x, the author has to be y and that it was published in journal z. To achive this we decided to use the tool `swish-e`[1] in combination with a set of tools with which *Html* and *Pdf* files can be easily converted to *Ascii* such that they can be indexed.

The second step in implementing our system was to decide on the user interface. For our prototype implementation it seemed overkill to implement a full-blown parser which is able to handle all LISA-D formulated queries. Designing and implementing parsers is a research field on its own. Therefore we have chosen another strategy. In our user interface the user can explicitly specify several query aspects:

- The *aboutness* is specified using a textfield.

[1]See `http://swish-e.org/`

- Property support for data resource types is specified using a pull-down menu in which the searcher can specify exactly one data resource type.
- Property support for representation types is also specified using a pull-down menu.
- Property support for other properties stems from bibliopgraphic data. These can be specified using textfields per property type (such as author, year of publication and so on)

From this data we can easily construct the LISA-D query which is a nice feature to check if the semantics of the actual query are indeed what the user intended to specify. For example, a LISA-D query generated by the system would be:

```
Data resource involved in Representation of Type “FullText”
    AND-ALSO of Type “PostScript”
    AND-ALSO having Atribution (with value “van Gils” AND-ALSO of Type “author”)
```

Our system searches for papers that match the criteria as well as possible. For individual results it offers the searcher the possibility to (manually) select a transformation from a list of possible transformations selected by the path finding algorithm (See Figure 5.2). After this step is completed, the transformation is executed and the result sent back to the searcher.

The interesting aspects of this prototype lie not so much in its practical use, especially since transformation selection is still a manual process. However, some interesting lessons can be learned from it. First of all, characterization of resources is not as straightforward as it may seem, especially if support for properties must be taken into account. In our domain we were “lucky” in the sense that meta data was available in the form of the bibliographic information. In other domains this may turn out to be a much bigger problem.

A second problem pertains to user interface design. Even though it is conceptually elegant to allow the user to specify his/her query in the form of a LISA-D / restricted language representation (see e.g., [Hop03] for a discussion on restricted language). Some important issues have to be dealt with when such a strategy is selected. For example, one must choose / implement a parser for this language. A more important issue is the following: searchers may specify properties for which we are unable to test the support for a given data resource. For example, a searcher may specify that the third letter of a data resource must have the color blue. It seems highly unlikely that the implementation of a search system has software readily available to test the support for this particular property.

Finally, we encountered a third problem that relates to the execution of transformations. The problem occured when we tried to convert LaTeX sources to another data resource type such as *Pdf*. These sources often consist of more than one file (i.e., a LaTeX file, a file for the bibliographic material, a file for each image etcetera) and may also generate more than one file. Presently, the only way to deal with this is to add building instructions to each source in our collection. A more elegant solution would be to amend our theory with an additional layer which is capable of handling collections / sets of data resources which belong together.

5.3 Conclusion

In this chapter we have experimented with several aspects of transformations. Our motivation behind these experiments was to test the applicability of transformations in a retrieval setting. We have found that, in theory, transformations can indeed be applied in retrieval systems elegantly and efficiently. However, many practical issues will have to be dealt with first. These, in turn, require more theory.

CHAPTER 6

Quality on the information market

Outline Aptness can be seen as a quality metric. Quality is therefore the focus of this chapter. Since the notion of quality is used in so many different fields we start this chapter with an extensive literature survey. After that we present a formal model for quality. This is the basis for our aptness metric as well as studying the quality of transformations.

In this chapter we start with an extensive survey of the literature on quality. From this we synthesize an abstract view on quality which is the basis for a formal model for quality. We will show that the measurement of qualities of artifacts (i.e., data resources on the Web) plays an important role in quality assessment. Therefore we also elaborate on the topic of measurements in this context. Last but not least we present our view on the quality of (value adding) transformations.

6.1 Introduction

In previous chapters we have already observed that the amount of information available to us has increased, the way it is presented to us has evolved, the importance of this information has increased, etcetera. This lead us to believe that it is increasingly important to find *apt* resources on the Web, rather than resources that are “only” topically relevant. Aptness can, as such, be seen as a metric for quality and it is the main theme of this chapter. This chapter thus brings together the ideas and models as presented in previous chapters. Relevant questions that we try to answer in this chapter are: What is the quality of the characterization of resource space? What qualities do resources have? What is the quality of a query? How well is it formulated and how accurately does it describe the searchers information need? What is the quality of a search engine/ match maker? What are its qualities?

To be able to answer these questions we firstly study the notion of quality as used in different fields by surveying literature. This survey is presented in Section 6.2 and suggests that there are two aspects of quality: qualities in the sense of attributes that artifacts may have and quality in the sense of desirability. More specifically, when an actor assesses the quality of an artifact then this assessment is based on some of the qualities that this artifact has. This synthesis of our survey leads to a formalized reference model for (both aspects of) quality, which is presented in Section 6.4.

It seems impossible to find a domain in which we can *express* quality in the sense of desirability. It does not make sense to state something like: “The quality of this artifact is 10”. Quality of an artifact only makes sense in comparison with other (similar) artifacts. As such quality provides an ordering similar to the value notion that we presented in Chapter 2. Observe, however, that we (humans) may associate a judgment (reasonable quality, poor quality) to this comparison based on the properties that an artifact has. The measurement of (the support for) these properties is, thus, an important issue for our work. It is often difficult to (automatically) measure which properties an artifact has, or which values it has to support a property. For example, different people may classify the *color* of an artifact different (red versus orange, blue versus green). Measurements and related topics are discussed in Section 6.5.

Last but not least, in Section 6.6 we specialize our notion of quality and tailor it specifically for the information market. This specialized quality notion can be used as an aptness metric. Finally in Section 6.7 we discuss the quality (or: value addition) of transformations.

6.2 Quality: a survey

In this section we present different views on quality such as the dictionary definition, philosophy, e-commerce, operations management, software engineering, the world wide web community and finally from the field of library information systems. From this survey we will distill a generic view on the concept “quality” which will be the basis for a formal model in the next section.

6.2.1 Dictionary

The *Webster’s third new international dictionary, unabridged* (1981) has an extensive entry detailing quality. The noteworthy headings in the entry are:

> peculiar and essential character; a distinct, inherent feature; degree of excellence; inherent or intrinsic excellence of character or type social status; a special or distinguishing attribute the character in a logical proposition of being affirmative or negative something that serves to identify a subject of perception or thought in respect in which it is considered something from the possession of which a thing is such as it is manner of action

The *Concise Oxford Dictionary* is, as its name implies, more concise. It states that quality is: “(1) the standard of something as measured against other things of a similar kind. (2) a distinguishing characteristic or characteristic.”. The *Wikipedia*[1] relates the notion of quality to different fields and states that:

> The term quality is used to refer to the desirability of properties or characteristics of a person, object, or process. In the case of a person this is considered in a particular context, such as worker, student, sports person, etcetera. The term is often used in opposition to quantity. In science, the work of Aristotle focused on measuring quality; whereas, the work of Galileo resulted in a shift towards the study of quantity.

It also describes that in manufacturing, the notion of quality relates to making a product fit for a purpose with the fewest possible defects (see also the ISO 9000 standard which specifies requirements for a Quality Management System overseeing the production of a product or service). Finally, quality can historically have four different interpretations: conformance to specifications, fitness for use, must-be / attractive quality and value to some person.

6.2.2 Philosophy

The notion of quality has a long history. It was already studied extensively by the ancient philosophers. For example, in his work on *the Philosophy of Nature* Aristotle used the notion of quality (See e.g., [IEP06]). In his work, Aristotle suggests that quality is the category according to which objects are said to be like or unlike.

Other great philosophers such as Descartes, Bacon, Newton, and Galileo oppose to Aristotle’s view on (the quality of) matter. In their view “the real or objective qualities of matter are extension in space, figure, number and motion wherever color, taste, smell, bitter or sweet are no more than mere names so far as the object in which we place them is concerned” [Eus87]. In other words, a distinction must be made between the objective qualities of matter and its largely subjective qualities.

6.2.3 E-Commerce

In [TLKC99] the problem of *quality uncertainty* is discussed. This problem boils down to the observation that in E-Commerce (loosely defined as doing business via the Web) customers often have difficulty accepting products or services from ‘strange vendors’ that may not even have a bricks and mortar back office. Two methods to deal with this problem are mentioned: *provide free samples* and *return if not satisfied.* The former, however, is difficult in case of digital products since they are consumed when they are viewed by customers.

[1] `http://en.wikipedia.org/wiki/Quality`

In [LASG02] the notion of quality of information is related to E-Commerce. The (lack of) quality of information about assets which can be either products or services can pose a risk for web-consumers. For example: a financial risk, a performance risk, risk for loss of time/convenience. In the description of a real-life situation (selling insurance via the Web), it is stated that "Controlling the information quality dimension is more challenging as quality can be addressed through either process or outcome measures." In other words, a distinction can be made between process quality and the quality of the actual outcome too.

6.2.4 Operations management

Quality is an important notion in operations management. In [Har96] an entire chapter is devoted to the topic of managing quality, involving concepts such as total quality management, quality of service, etcetera. The key dimensions of quality are defined to be: product attributes, product performance, service characteristics, warranty, service availability, and total price.

In the context of operations management, the question "How can operations contribute to delivering a quality product?" must be answered with: operations management is concerned with deciding on the most suitable production process through job design, production planning and control, obtaining resources for production, and with quality control in the sense of ensuring that products leaving the workplace conform to specifications. *Conformance to specification* is the central theme in operations research, especially when Total Quality Management (TQM) is considered. TQM emphasizes, at every link in the production chain, the need to arrive at agreement on performance requirements, supplier capability, timing, cost, and the monitoring of changing needs. To put it in the words of [LL96]: TQM is a concept that makes quality the responsibility of all people within an organization.

The conformance to specification approach is criticized in [LL96] for its sole focus is the supplier perspective. The consumer perspective is, according to the authors, more concerned with *value for the dollar* (i.e., getting your money's worth). This includes both characteristics of the product/service that is bought and psychological aspects such as how knowledgeable the support staff is, courtesy of the staff, etcetera.

The focus in [Pij94] is on the ex-post evaluation of quality of information in organizations. The ISO-8402 definition of quality:

> The totality of features and characteristics of a product, process or service that bear on its ability to satisfy stated or implicit goals.

is used as a starting point. The author makes several observations: Firstly, any conceptual quality model should take account of the importance of the production process. Secondly, the quality of a product or service has to be considered in the light of the use that is made of it. Last but not least, quality is described in terms of a series of features and characteristics of a product, service or process.

In his quality model, Van der Pijl proposes a dual view on quality: the *causal point of view* deals with the quality of information, seen as the result of the quality of the process in which it is produced. In the *teleological point of view* the quality of information is seen as the degree to which it satisfies stated or implicit needs, derived from the situation in which it is used.

6.2.5 Software Engineering

In the field of software engineering the notion of quality plays two important roles: the quality of the software itself on the one hand and quality of the software engineering process on the other.

[Som89] discusses quality in the chapter on *Quality management*. The author starts with the observation that the classical notion of quality (conformance to specification, see Section 6.2.4) is difficult to apply to software systems because:

- The specification should be oriented towards the characteristics of the product that the customers wants. However, the development organization may also have requirements which are not included in the specification.
- We do not know how to specify certain quality characteristics in an unambiguous way.
- It is very difficult to write complete software specifications. Therefore, although a software product may conform to its specification, users may not consider it to be a high-quality product.

It is emphasized that the quality of a software system can only be assessed in terms of *quality attributes* (such as safety, security, reliability, resilience, robustness, learnability, etcetera) and that software quality management can be structured into three principle activities: quality assurance, quality planning, and quality control. Standards (product standards, process standards, documentation standards) play an important role in these activities. Last but not least, *software metrics* can be used to make quality measurable:

> Software measurement is concerned with deriving a numeric value for some attribute of a software product or a software process. By comparing these values to each other and to standards which apply across an organization, it is possible to draw conclusions about the quality of software or software processes.

Also in [DO85], where quality is defined as excellence or fitness, it is proposed to measure the quality of information systems by means of characteristics such as complete data, accurate data, relevant output, meaningful output, etcetera. A highly systematic approach to measuring quality attributes of a system is needed if measurement using attributes is to succeed, for faulty measurements lead to an incorrect assessment of the quality of the system under consideration. In the classic work [BJ87] the author indeed corroborates that quality (of software products) can only be achieved with discipline and systematic software quality controls.

Even more, in [McC04] it is explained that the attempt to maximize certain aspects of quality inevitably conflicts with the attempt to maximize other aspects. At a certain level of abstraction this can be interpreted as: increasing quality conflicts with increasing quality. In practice this is dealt with by prioritizing the different quality characteristics and maximize within certain bounds (i.e. a budget). In [Gil88] the focus is on the process of software engineering where determining attribute specification is one of the difficult problems. Attributes are in two kinds:

Resources (people, time, money): are almost always limited

Qualities or benefits (performance, reliability): we always want more than we can afford.

Three principles that relate to this particular problem are:

The principle of unambiguous quality specification : asserts that all quality requirements can and should be stated unambiguously;

Kelvin's principle : asserts that when you can measure what you are speaking of and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind;

Shewhart's measurable quality principle : states that the difficulty in defining quality is to translate future needs into measurable characteristics so that a product can be designed and turned out to give satisfaction at a price the user will pay.

6.2.6 Quality on the Web

In [GMSS04] a discussion on the quality of data on the Web is presented. This discussion starts off with the observation that:

> Well-founded and practical approaches to assess or even guarantee a required degree of the quality of data are still missing.

According to the authors, data quality comprises more than the format of the data (text, multi-media, streaming data); it has to do with how fit (*apt*) data is for consumers. Relevant keywords in this respect are accuracy, completeness, timeliness, and consistency. As such, data quality can not be studied in isolation. The data producers, custodians (entities that provide and manage data), as well as consumers have to be taken into account as well. The authors propose that a *quality algebra* be used for assessing/ dealing with data quality on the Web. Factors to take into account when designing such algebra are: data quality assessment, data quality interpretation, and data quality dynamics.

The above mentioned custodians are called information intermediaries in [VW99]. The authors pose that user concerns about (their perception of) the quality of information on the Web continues to be a strong incentive for "the emergence and success of information intermediaries." They can play an important role in the trust relationship between suppliers and consumers, as well as in quality/price control:

> Quality in information products is a complex and elusive phenomenon. It can be described on the basis of outcomes for their users and potential increase in the efficiency for the tasks they perform. A broader understanding of quality can comprise not only quality properties but also additional parameters of clearly qualitative nature.

The central observation in [Orr98] is that data quality is the measure of the agreement between the data view presented by an information system and some data in the real world. The authors propose the following six data quality rules for maintaining the quality of data in (web) information systems:

1. Unused data cannot remain correct for very long
2. Data quality in an information system is a function of its use, not its collection.
3. Data quality will be no better than its most stringent use.
4. Data quality problems tend to become worse as the information system ages.
5. The less likely some attribute (element) is to change, the more traumatic it will be when it finally does change.
6. Laws of data quality apply equally to data and meta-data.

6.2.7 Library Information Systems

Another field where the notion of quality plays an important role is the library and information science community. An interesting starting point in this respect is [DES05] which has a "comprehensive list of possible selection criteria" in the context of information gateways (brokers on the information market). This list can be summarized as follows;

Relating to the internal quality of resources	Relating to quality in the subject gateway context
Content criteria	Scope criteria
Form criteria	Collection criteria
Process criteria	

It is interesting to observe that the authors make a distinction between quality aspects pertaining to the resources themselves and quality aspects pertaining to the process of locating / accessing the resources. In [HC05] the focus is on the latter, and gives an extensive list of efforts geared towards achieving a high quality of service. The authors observe that:

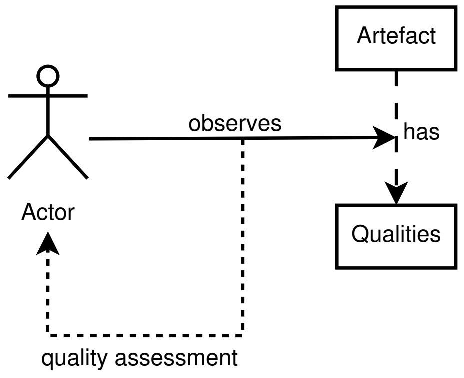


Figure 6.1: Two views on quality

> Most researchers in library and information science have concentrated on the perspective of services quality as meeting and/or exceeding expectations.

The authors then elaborate on the several models for service quality such as LibQUAL and WebQUAL[2]. These approaches all have in common that they focus on quality attributes of the offered services. As an example, the LibQUAL approach lists the following *dimensions of library services quality*: reliability, affect of service, ubiquity of access, comprehensive collections, library as place and self reliance.

6.3 Synthesis

In the previous section we have studied several fields in which the notion of quality plays an important role. From this study we conclude that there are two views on quality as illustrated in Figure 6.1:

Property : the 'qualities of something'. At some level of abstraction this view on quality can be considered objective. However, deciding whether something has a property or not can also lead to philosophical discussions. It remains to be seen if an 'objective reality' exists or not.

Desirability : has to do with 'how good' something is (in comparison to other things). This is a subjective view on quality.

These two views on quality are, obviously, related. The intuition is that the quality (desirability) of an artifact is high if it has the right qualities (properties) for a specific actor. In the fields that we presented previously, both views on quality play a role but usually one of them is dominant:

- In the field of philosophy, at least the work cited here, the focus is mainly on the property-aspect of quality.

[2]See `http://www.libqual.org` and `http://www.webqual.net` respectively

- In the field of e-commerce the focus is on optimzing the quality (desirability) of products / services. This quality is dependent of the qualities of the products / services under consideration.
- A similar line of reasoning holds for operations management where the goal is to deciding on the most suitable (high quality) production process.
- etcetera.

In the next section we will present a formalism for reasoning about quality. The goal of this formalization is twofold. On the one hand we want to express more precisely (a) what it means when we assert that an artifact has a certain quality (property), and (b) study the relationship between the two views on quality further. The formalism will be the basis for our aptness metric.

6.4 A reference model for quality

In the previous section we presented a detailed survey on the literature of quality in different fields. From this we learn that there are two main views on quality: quality in the sense of properties and quality in the sense of desirability. In Section 6.4.1 we firstly zoom in on qualities in the sense of properties. In Section 6.4.2 we will shift the focus to quality in the sense of desirability and, thus, present quality metrics. Observe that in this section our focus is on a reference model for quality. Therefore our examples will be from the domain of physical artifacts such as mugs, cars, etcetera. In the subsequent sections we will present aptness metrics as well as examples more tailored to the Web domain.

6.4.1 Quality & Properties

We will present our model for quality in two steps. Firstly, the intuition behind our model has to be presented. We will use motivating examples for this. Secondly, we will present a formalism. Figure 6.2 shows our model using the Object Role Modeling (ORM) notation[3] which provides the signature for the formalism. In the remainder of this section we will use the terminology introduced in this Figure.

The first observation that we must make is that the artifacts can play different roles. For example, a mug can be seen as a device from which you can drink tea; it can be seen as an art object or even as a place to store pens in. The quality of some artifact depends on which role this artifact plays. Continuing the above example: a mug can be great as a drinking device but be horrible as an art object. We will model this as follows: Let $\mathcal{AF}$ be a set of artifacts that may have certain qualities (properties) and let $\mathcal{RO}$ be a set of role types that these artifacts can fulfill [4].

[3] See Appendix B for an overview of this notation.

[4] We use the words 'role types' and 'roles' interchangably since they can be disambiguated from the context.

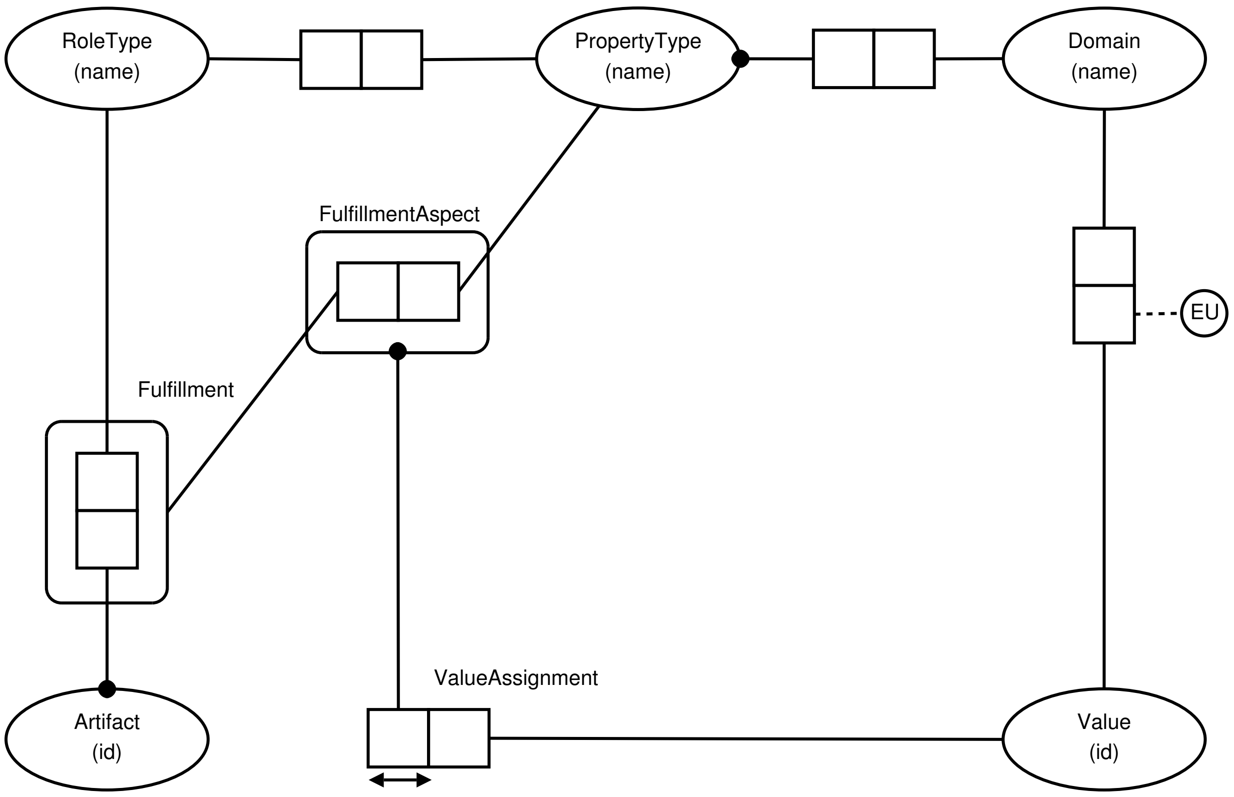


Figure 6.2: Properties of artifacts

The combination of an artifact and a role is dubbed a *fulfillment* (i.e., a fulfillment denotes an artifact in a role): $\mathcal{FL}$. The artifacts and roles that participate in a fulfillment can be found using the functions $\mathsf{Artifact} : \mathcal{FL} \to \mathcal{AF}$ and $\mathsf{Role} : \mathcal{FL} \to \mathcal{RO}$ respectively. Since a fulfillment denotes an artifact in a role, we know that an artifact and a role combination uniquely determines a fulfillment:

Axiom 35 (Unique fulfillment)

$$\mathsf{Artifact}(e_1) = \mathsf{Artifact}(e_2) \ \wedge \ \mathsf{Role}(e_1) = \mathsf{Role}(e_2) \ \Rightarrow \ e_1 = e_2$$

For convenience of notation we introduce the following abbreviation for a fulfillment;

$$\langle a, r \rangle \ \triangleq \ e \text{ such that } \mathsf{Artifact}(e) = a \ \wedge \ \mathsf{Role}(e) = r$$

This allows us to write $\langle \text{MyMug}, \text{drinking device} \rangle$ for a specific fulfillment. The following example illustrates the use of artifacts, roles and fulfillments in our model.

Example 6.4.1 *Let* Mug *(denoted by a) be an artifact that can play two roles. It either plays the role of type:* something to drink from *(denoted by r_1) or the role of type:* art object *(denoted by r_2). Both $e_1 = \langle a, r_1 \rangle$ and $e_2 = \langle a, r_2 \rangle$ are artifacts such that:*

$$\begin{array}{ll} \mathsf{Artifact}(e_1) = a & \mathsf{Role}(e_1) = r_1 \\ \mathsf{Artifact}(e_2) = a & \mathsf{Role}(e_2) = r_2 \end{array}$$

Recall that the quality (desirability) of an artifact depends on its qualities (properties). Furthermore, observe that properties should not be coupled to artifacts as such, but to the roles that these artifacts play. To see why this is the case one only needs to realize that, for example, all mugs have a volume; that all vehicles have a maximum speed; that all storage devices have a capacity, etcetera. Furthermore, properties such as speed and capacity can be expressed in different domains. For example, consider the property type color. This can be expressed in the domain *RGB color* but also as *CMYK color.*

We model this as follows: Role types can have properties, the value of which are expressed in a property domain. Let $\mathcal{PT}$ be the set of property types and $\mathcal{PD}$ be the set of property domains. The properties that can be played by a certain role type are given by the relation $\mathsf{Props} \subseteq \mathcal{RO} \times \mathcal{PT}$ and the domain in which (values of) a property can be expressed is given by the function $\mathsf{PrDom} \subseteq \mathcal{PT} \times \mathcal{PD}$. We continue the above mentioned example to illustrate the use of our model further.

Example 6.4.2 *Role type* art object *(r_2) can have the property type* color *(denoted by p) which can be expressed in the domain* RGB-colors *(denoted by d_1) and the domain* CMYK-colors *(denoted by d_2) such that:* $\mathsf{Props}(r_2) = \{p\}$ *and* $\mathsf{PrDom}(p) = \{d_1, d_2\}$

Note that property types and domains are at the typing level, similar to what we did in our model for information supply (see Figure 3.2 on page 43). We still need to assign values to artifacts having a certain property type. The first step to achieve this is to create a link between $\mathcal{PD}$ and the values from this domain. The set $\mathcal{VL}$ consists of sets of values for a certain domain. In other words, an element from $\mathcal{PD}$ is the *names* of a certain domain and an element of $\mathcal{VL}$ consists of its values. In the ORM-schema (Figure 6.2) the extensional uniqueness constraint denotes the fact that the values uniquely determine the domain(name). The functions $\mathsf{Value} : \mathcal{PD} \to \mathcal{VL}$ and $\mathsf{VlDom} : \mathcal{VL} \to \mathcal{PD}$ are used to find the values of a domain or the name of a set of values respectively. For example:

Example 6.4.3 *The domain* RGB-colors *(d) for colors has the range of values* $v = \{\#000000 \ldots \#FFFFFF\}$*. More specifically:* $\mathsf{Value}(d) = v$ *and* $\mathsf{VlDom}(v) = d$

Last but not least we should introduce a notation for expressing the fact that a fulfillment has an associated value for a certain property. For example, we should be able to express that a mug has a volume of $20cc$. The property type of a fulfillment is denoted in our model by a *fulfillment aspect.* The set of these fulfillment aspects is denoted by $\mathcal{FA} \triangleq \mathcal{FL} \times \mathcal{PT}$ such that

$$\langle f, p \rangle \in \mathcal{FA} \;\Rightarrow\; p \in \mathsf{Props}(\mathsf{Role}(f))$$

The intended meaning is as follows:

Example 6.4.4 *Let* $f = \langle \textit{mug}, \textit{drinking device} \rangle$ *denote the fulfillment of a mug in its role as drinking device and let color* $\in \mathcal{PT}$ *be a property type. Then* $\langle f, \textit{color} \rangle$ *is a fulfillment aspect denoting the color of mugs in their role as drinking device.*

This notion of fulfillment aspects may seem somewhat unnatural. We introduce this concept here mainly to make the remainder of our formalization more elegant. In our model we will use the predicate $\mathsf{ValAss} : \mathcal{FA} \to \mathcal{VL}$ to denote the observation that a fulfillment has a certain value for a property type. Continuing our example:

Example 6.4.5 *The fact that the mug (a) as an art object (r2) has the color (p) red (#FF0000) is expressed as:* $\mathsf{ValAss}(\langle a, r_2 \rangle, p) = \#FF0000$

In our model we have to ensure that the observations on the instance level do not conflict with the typing level, something that is 'obvious' in the real world. For example, if a fulfillment is said to have a value assignment for a property then, obviously, one of the roles of this fulfillment must at least have this property. Similarly, consider the observation: $\mathsf{ValAss}(\langle \text{mug}, \text{drinking device} \rangle) = 20cc$. To be able to make this observation, the value $20cc$ must be in $\mathcal{VL}$ and it must be of the correct domain. That is, it must be of the domain in which the property type can be expressed. The following axiom enforces that the typing level and instance level stay in sync. Let f be a fulfillment, p a property type and v a value:

Axiom 36 (Conformance)

$$\mathsf{ValAss}(f, p) = v \;\Rightarrow\; p \in \mathsf{Props}(\mathsf{Role}(f)) \;\wedge\; \mathsf{PrDom}(p) = \mathsf{VlDom}(v)$$

In order to be able to operationalize this model for quality properties, a measuring method has to be developed for:

- measuring the roles that an artifact can play
- measuring the property types that exists
- measuring the value assignment of a fulfillment

The fact is that devising such a measuring method is a problem in itself. In [Ald02]. Ken Alder writes "Our methods of measurement define who we are and what we value." In his book, Alder describes the quest or a universal measure for distance in the late 1790's by two astronomers. Their task was to establish a new measure (the meter) as one ten-millionth of the distance from the North Pole to the equator. This is, obviously, by the standards deployed in these days, as well as by modern standards, a daunting task to say the least. However, given the many different "standards" for measuring distances that were used in that time, something had to be done. For example, different interpretations of *un pied* (a foot) seriously hampered commerce between different regions in France.

As this example illustrates: agreement of stakeholders is important. Sufficiently many people involved should agree on the roles that an artifact can play, the properties that exist, etcetera. For example, if two stakeholders can not agree on the color(s) of a mug or the roles that this mug can play: what good will a system be, then? Note that, in essence, there are two ways of measuring systems

objective : some value assignments can be measured objectively. For example: the number of characters in a file, or the weight of an artifact,

subjective : other value assignments are, really, dependent on humans. For example: is an artifact expensive, or is it pretty?

We will return to this issue in the upcoming sections. We will now zoom in on the second view on quality: quality in the sense of desirability.

6.4.2 Quality & Desirability

To be able to assess the quality (in the sense of desirability) of an artifact for a user, his/her actual desires must be made explicit. We already touched upon this subject in Sections 2.2.3 where we discussed the goals of actors in the context of valuing, and Section 4.5.2 where we discussed user preferences with respect to properties of data resources. The question is how to do this. One of the main problems is to choose a domain in which quality is expressed. To be more precise, it doesn't seem to make sense to say: "The quality of this artifact is 24." The notion of quality is abstract and can be used to compare artifacts.

Quality, in the sense of desirability, depends on the desires of people (actors). However, these actors are not always aware of their desires, or may not know how to express them. Such issues also arise in other fields such as:

- Software engineering: stakeholders have to, somehow, express requirements with regard to a system. See e.g., [KG03, Som89, Bev99]

- Search on the web: searchers must try to specify their information need. See e.g., [BBWW98, Gro00, HPW96]

Furthermore, a distinction must be made between *hard* and *soft* desires with regard to artifacts. These can be compared, to some extent, to functional and non-functional requirements or hard goals and soft goals in requirements engineering (See e.g., [DB04]). In requirements one often tries to *make soft goals hard.* In our opinion, a goal/ requirement is considered to be *soft* if a human opinion is needed for the value assignment. Otherwise, it is considered to be *hard.* In other words, hardness or softness of a requirement depends on the way of measurement. The following are examples of hard goals and soft goals:

hard goals : Price may not exceed €20. Volume equals 25 liters. Made of stainless steel.

soft goals : Cheap. Pretty. Low. Hard. Strong.

Quality in the sense of desirability depends on the *requirements* of an individual with respect to qualities. More specifically: these requirements have to do with value assignments; the quality of some fulfillment increases if properties have "the right value". Putting it differently, value assignments are *constrained.* Consider the following examples of a requirement for a fulfillment:

Example 6.4.6

- The price may not exceed €10
  In this example, price *is a property type which is expressed in the domain* €'s. *Furthermore, 10 is a value and* may not exceed *is a constraint.*

- The price in euros must be as low as possible
  In this example, price *is a property type which is expressed in the domain* €'s. *Furthermore,* must be as low as possible *is a constraint.*

- The price in euros may not exceed the price of cup c
  In this example, price *is a property type which is expressed in the domain* €'s. *Furthermore,* may not exceed the price of cup c *is a constraint involving an assignment.*

Observe that the frst requirement has a property type, a constraint and a value and the other two requirement do not specify a value. We model this as follows: Let $\mathcal{RQ}$ be the set of requirements and $\mathcal{CS}$ be the set of constraint operators[5]. A requirement adheres to a property type (mandatory), a constraint (mandatory) and possibly an *expression* (optional).

Expressions can either be values or value assignments, as illustrated by the above examples. In the first example the expression is a value whereas in the last two examples the expression is another value assignment. Traditionally, expressions are often modeled in

[5]In the following text we will abbreviate "constraint operator" with the simpler, and more readable "constraint".

terms of base expressions (literals) which can be combined by operators and possibly some logical connectors. Consider example, the expression $P(x) \ \wedge \ Q(x,y)$. This expression has a unary operator P and a binary operator Q. Even more, the expressions are coupled using a logical and. In terms of our model we need only a subset of this full approach. Therefore we model expressions to be values, whether they are atomic (i.e., just another value) or the result from a value assignment. Let $\mathcal{EX}$ be the set of expressions. Furthermore, let $\mathsf{Prop} : \mathcal{RQ} \rightarrow \mathcal{PT}$, $\mathsf{Constr} : \mathcal{RQ} \rightarrow \mathcal{CS}$, and $\mathsf{Expr} : \mathcal{RQ} \rightarrowtail \mathcal{EX}$. We introduce the following shorthand notation:

$$\begin{aligned} r_1 = \langle p, c, e\rangle &\triangleq \mathsf{Prop}(r_1) = p \wedge \mathsf{Constr}(r_1) = c \wedge \mathsf{Expr}(r_1) = e \\ r_2 = \langle p, c\rangle &\triangleq \mathsf{Prop}(r_2) = p \wedge \mathsf{Constr}(r_2) = c \end{aligned}$$

The previous examples can now be written more formally as follows (we include a verbalization of the constraint, an expression in terms of our formalism and a LISA-D expression[6]):

Example 6.4.7

- The price may not exceed €10
  $\langle \mathit{price}, <, \text{€}\mathit{10}\rangle$
  Requirement on Property Type "Price" by Constraint Operator "may not exceed" is Value "10 euro"

- *The price in euros must be as low as possible*
  $\langle \mathit{price}, \min\rangle$
  Requirement on Property Type "Price" is Constraint Operator "minimize"

- *The price in euros may not exceed the price of cup c*
  Letting g denote the fulfillment of cup c in some role:
  $\langle \mathit{price}, <, \mathsf{ValAss}(c, \mathit{price})\rangle$
  Requirement on Property Type "Price" by Constraint Operator "may not exceed" is the Value of Artifact "c" with respect to Property Type "price"

Figure 6.3 illustrates how requirements are positioned in our quality-model. Note that a requirement with respect to a fulfillment is of a certain actor/ individual. Let $\mathcal{AC}$ be the set of actors and $\mathsf{Req} : \mathcal{AC} \times \mathcal{FL} \rightarrow \wp(\mathcal{RQ})$ denote the requirements of an actor with regard to a fulfillment. For example:

$$\mathsf{Req}(a, f) = \{r_1, r_2\}$$

denotes the observation that actor a has requirements r_1 and r_2 with regard to fulfillment f. Last but not least we will point out the relation between quality assessment and choice. To this end, consider the following example situation in which you want to buy a mug (in its role of a 'drinking device'):

Example 6.4.8 *The decision space is summarized by:*

[6]Appendix B

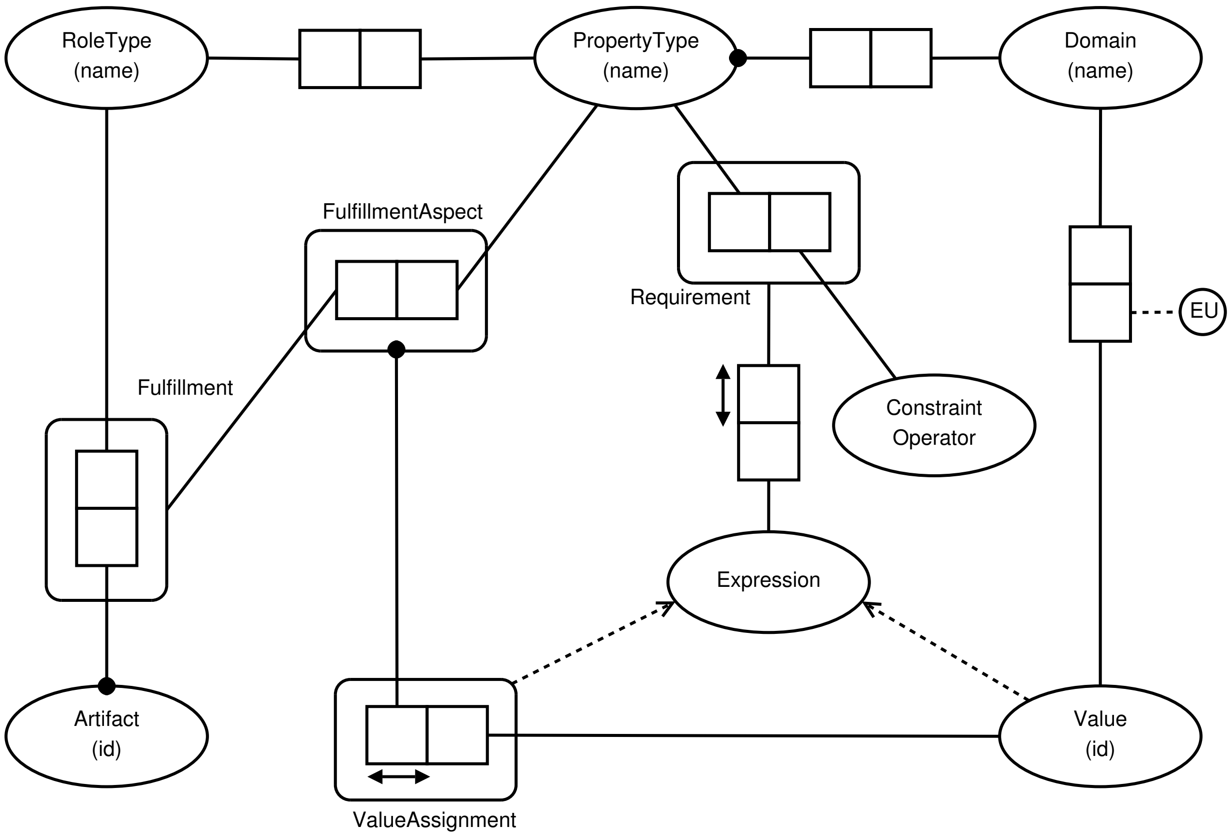


Figure 6.3: Requirements & constraints

	property type		
	color	***volume***	***price***
m_1	*red*	20*cc*	*€3*
m_2	*red*	25*cc*	*€3*
m_3	*blue*	25*cc*	*€2*

Which mug is best (i.e., has highest quality for an actor a) depends on the requirements of the actor. Let f denote the fulfillment of a mug artifact in its role as a drinking device and $\mathsf{Req}(a, f) = \{r_1, r_2, r_3\}$ *where* $r_2 = \langle color, =, red\rangle$, $r_3 = \langle volume, \geq, 25cc\rangle$ *and* $r_1 = \langle price, \leq, e3\rangle$. *In this case, it seems apparent that* m_1 *is not feasible: for this actor it is over priced and too small.* m_2 *and* m_3 *seem equally feasible for 2 out of 3 requirements are matched. Furthermore, if the price attribute is more important than the color then* m_3 *will be chosen, if color is more important then* m_2 *will be chosen.*

Literature suggests numerous ways to deal with these kinds of selection/ optimization problems such as Operations Research [KA97, Tah92] and multi-objective decision making [Diw03, Bom95]. For example, one may opt to model this using a relative prioritization of the requirements, weighing of the requirements or using several objective functions. Discussing these approaches in detail is beyond the scope of this paper. Observe that it is important to decide what kind of problem is under consideration: finding the fulfillment which conforms to all constraints is a completely different problem than finding a fulfillment that is best, given these constraints.

The above *selection problem* may be solved adding weights to the requirements.

Example 6.4.9 *Suppose that the following weights are added to the requirements:*

requirement	***weight***
r_1	*0.4*
r_2	*0.3*
r_3	*0.3*

It is easy to verify that a considers m_2 *to be of the highest quality (color is more important than price).*

This concludes our exploration of the (general) notion of quality. In the upcoming sections we will firstly zoom in on measurements and uncertainty related to these measurements. After that we present our view on quality on the information market and thus present our aptness metrics.

6.5 On the measurement of qualities

Assessing the quality (desirability) of some artifact for some actor is tricky, to say the least. As we have explained in the previous section, actors make quality assessments based on goals/ constraints. These constraints are, usually, a *linguistic statement* such as: *I will assess the quality of this car to be high if its top speed is high*, where it is unclear how

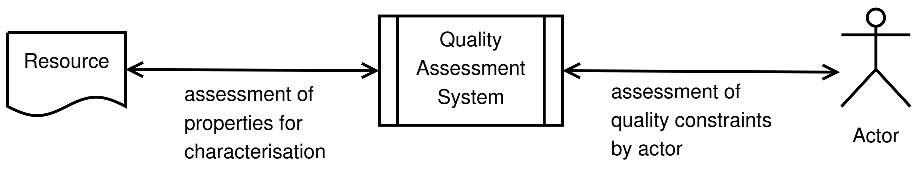

Resource
Quality Assessment System
Actor
assessment of properties for characterisation
assessment of quality constraints by actor

Figure 6.4: Uncertainty in quality assessment

high is to be interpreted. In other words, the quality assessment system has to deal with uncertainty about the constraints posed by the searcher. A second kind of uncertainty has to do with the observations/ measurements made by the system. For example:

- The fact that a resource has (outgoing) hyperlinks can be be measured with near 100% certainty.
- The language of a resource is more difficult to measure. For example, consider the subtle differences between American English and British English, or between Dutch and Flemish, for that matter. It is very well possible that a quality assessment system can only establish the language of a resource with only 90% certainty.

In other words, the quality assessment system has to take different kinds of uncertainty into account as illustrated by Figure 6.4. Quality assessment systems have to somehow deal with the uncertainty involved with measuring whether or to what degree a resource has a certain property, as well as determine the constraints that the actor uses for quality assessment. In order to come to a quality assessment of an artifact for an actor, the quality assessment system has to somehow combine the 'hard' (often numerical) measurements made with the 'soft' (and linguistic) classifications made by actors. The concept of a *linguistic variable* provides an elegant way to model the fuzzy assessments made by actors. In Section 6.5.1 we will introduce the concept of a linguistic variable based on [Zad75a, Zad75b, Zad75c, Zad02]. In our discussion of linguistic variables we will adopt the same notation as used in Zadeh's papers. In Section 6.5.2 we will present our view on quality which uses the fuzzy concept of a linguistic variable. We will illustrate how this can be used to come to an actual measurement for the quality of a resource to a searcher by means of an extensive example.

6.5.1 Linguistic Variable

In this section we introduce the concept of a *linguistic variable*. First, we briefly study the notion of a "normal" variable which can be described as follows:

> In computer science and mathematics, a variable is a symbol denoting a quantity or symbolic representation. In mathematics, a variable often represents an unknown quantity; in computer science, it represents a place where a quantity can be stored. Variables are often contrasted with constants, which are known and unchanging.

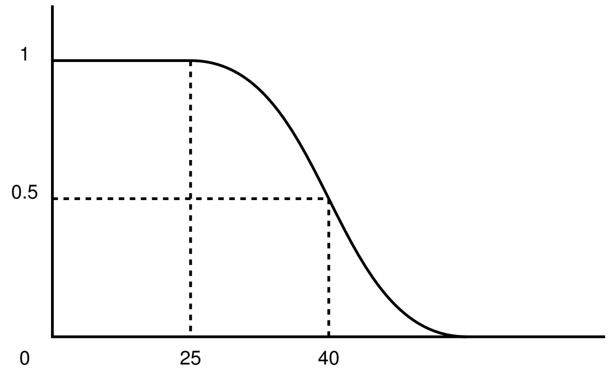


Figure 6.5: Membership function for *young*

In other words, a variable x has an associated domain $\mathsf{Dom}(x) = D$ from which it can be assigned values. For example if $\mathsf{Dom}(x) = \mathbb{N}$ then we can assign natural numbers to x (but not, say, reals). Further more, x either has a certain value or it doesn't.

The main distinction between fuzzy variables and non-fuzzy variables lies in the membership function. In case of fuzzy variables, the assignment of a value to a variable has a *membership degree* which expresses to what extent a variable has a certain value. Formally, a fuzzy variable is characterized by a triple $\langle X, U, R\rangle$ where X is the name of the variable, U is the universe of discourse, and R if the fuzzy restriction on U characterized by a membership function. Consider the following example. Let y denote the fuzzy variable *young*. The underlying domain U is age in years and the fuzzy restriction is characterized by the membership function as shown in Figure 6.5. In the given example $\mu_y(40) = 0.5$ indicates that an age of 40 years has membership degree 0.5 for the fuzzy variable young.

Finally, we can turn our attention to the concept of linguistic variables which differ from normal, numerical, variables in that its values are not numbers but words, or sentences in some language. This makes the concept of a linguistic variable of a higher order than a fuzzy variable, in the sense that a linguistic variable takes fuzzy variables as its values. For example, the linguistic variable *age* might take *young*, *not young*, *old* or *not very old* as its values.

In Zadeh's formalism for fuzzy logic (e.g., [Zad75a]), a fuzzy variable is characterized by a quintuple $\langle \mathcal{X}, T(\mathcal{X}), U, G, M\rangle$ in which

- $\mathcal{X}$ is the name of the variable,
- $T(\mathcal{X})$ (or simply T) denotes the term-set of $\mathcal{X}$, that is, the set of *linguistic values* associated to $\mathcal{X}$,
- U is a concrete (numerical) domain which is linked to the linguistic values by means of membership functions,
- G is a syntactic rule (which usually has the form of a grammar) for generating the linguistic values of $\mathcal{X}$,

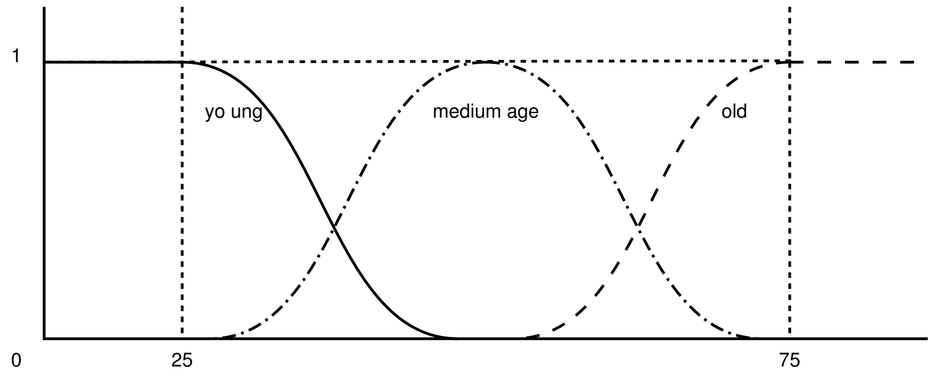


Figure 6.6: The linguistic variable *Age*

- M is a semantic rule for associating with each X its meaning.

In more recent approaches the syntactic and semantic rules are often not included. The following example illustrates how linguistic variables work without going into details on these rules.

Example 6.5.1 *Let* $\mathcal{X} = \text{age}$ *be a linguistic variable with* $U = [0, 100]$*. I.e., we assume that people do not get older than* 100 *years. In this case* young *is considered to be a linguistic value of* $\mathcal{X}$*.*

More specifically, if $T(\mathcal{X}) = \{\text{young}, \text{medium age}, \text{old}\}$ *then Figure 6.6 illustrates the possible value assignments with their respective membership functions. In this example everyone below* 25 *years of age has membership degree* 1 *for the fuzzy variable* young *and everyone over* 75 *years of age has membership degree of* 1 *for the fuzzy variable* old*.*

Frequently, the syntactic rule G that generates the terms in T is a context-free grammar. In other words, $T(\mathcal{X})$ is described by G. For example:

$$\begin{array}{rcl} T & \rightarrow & \textit{young} \\ T & \rightarrow & \textit{very } T \end{array}$$

The above example G is capable of generating terms such as *young*, *very young* but also *very … very young.* To compute the meaning of such term one only needs the meaning of the term *young* (i.e., μ_{young}) and the meaning of the term *very.* The former is a *primary term*, that is, a term whose meaning must be specified as an membership function. The latter is a *linguistic hedge*, that is, a modifier of the meaning of its operand. These can be specified as function that operates on the membership function. The example membership function given in [Zad75b] for the variable *young* is as follows:

$$\begin{array}{rcll} \mu_{\ young} & = & 1 & \text{for } 0 \leq u \leq 25 \\ & & \left[1 + \left(\frac{u-25}{5}\right)\right]^{-1} & \text{otherwise} \end{array}$$

Even more, if the interpretation of the hedge *very* is the square of the term to which it belongs then the interpretation of *very old* is the square of the above function.

Last but not least, the interpretations of the *fuzzy and*, *fuzzy or* and *fuzzy not* have to be defined. These are fairly straightforward and similar to their logical counterparts. Let $\sqcap$, $\sqcup$ and $\neg$ denote the fuzzy and, fuzzy or and fuzzy not. Furthermore, assume we have a linguistic variable $\mathcal{X}$ with underlying domain U and restriction R. Let X_1 and X_2 be two linguistic values of this variable (i.e. $X_1, X_2 \in T(\mathcal{X})$) such that for a given object o we have:

$$\mu_{R(X_1)}(o) = p_1 \text{ and } \mu_{R(X_2)}(o) = p_2$$

Then for this object we have the following membership degrees:

- $X_1 \sqcap X_2 = \min(p_1, p_2)$
- $X_1 \sqcup X_2 = \max(p_1, p_2)$
- $\neg X_1 = 1 - p_1$

6.5.2 Measurements of qualities

In the previous subsections we have explained the two kinds of uncertainty that play a role in quality assessments: uncertainty related to the interpretation of measurements and uncertainty related to the actual measurements. Furthermore, we have introduced the concept of a linguistic variable to model the former. In this section we will zoom in on the latter: measurements of qualities.

Firstly we must define what it means if we assert that we measure some property of an artifact (to have a certain value) with some degree of certainty. An important observation in this respect is that measurements depend on the situation in which they are done. For example, measuring the weight of an artifact depends on the location (on the moon, versus earth). Furthermore, the measuring device is another cause for concern. For example, one thermometer may be less accurate than another. To model this we introduce $\mathcal{SI}$ to be a set of possible situations and $\mathcal{MD}$ to be a set of measuring devices.

Two additional observations are relevant to our discussion here. First of all, two different kinds of measurements can be done:

1. One can attempt to measure the value of some property of an artifact
2. One can attempt to verify whether the value associated to a property of an artifact equals some value

This implies that a measurement always results in some value. In the first case it is the value that is measured but in the second case it would be a boolean true/false. Let $\mathcal{MV}$ be the union of possible value domains. A measuring device $R \in \mathcal{MD}$ can now be modeled as a function that maps object-situation combinations into values:

$$R = [\mathcal{AF} \times \mathcal{SI}] \rightarrowtail \mathcal{MV}$$

Furthermore, we can denote a specific measurement with $\mathsf{M}(a, s, d) = v$ where a denotes the artifact under consideration, s the present situation, d the measuring device and finally v the actually observed value. The following example illustrates how this may be used.

Example 6.5.2 *Let a be the car or John Doe. At a certain point in time, John is driving down the highway somewhere in Europe. Let s denote his situation, i.e. his current point in the space-time continuum. John happens to be so fortunate to drive past a police officer who uses a certain device d which checks the speed of cars. The observation that John is driving at a speed of* 125km/h *is expressed as:* $\mathsf{M}(a, s, d) = 125\text{km/h}$

A remaining, yet very important, issue is: what about the accuracy of measurements? In this context one must realize that (values of) measurements are expressed in a domain and that there are standards for expressing them. For example, speed can be measured in terms of kilometers per hour, weight can be measured in terms of grams, distances in terms of meters and so on. It is possible / likely that a measurement is not 100% correct in the sense that the measuring devices may not be 100% correct. A measurement, thus, should be seen as a strong indication for the actual value that one wanted to measure.

Standards bodies (department of weights and measures) govern these standards. By comparing an actual measurement to the measurement by a standards body (we dub this the standard measurement) one obtains a metric for determining the accuracy of a measurement device. To continue the above example:

Example 6.5.3 *Let d_s be an 'approved' measuring device for speed. I.e., it measures exactly according the department of weights and measures. This means that a measurement executed with this device is always* 100% *correct. If* $\mathsf{M}(a, s, d) = \mathsf{M}(a, s, d_s)$ *then we know that John was indeed driving exactly at* 125km/h.

In many cases there will be a small deviation between the two measurements. As such, in practical settings, one should either work with intervals (we measured value v so the real value should be $v \pm \delta$ where δ is a very small number representing the deviation of measurements with this device.

In many cases such a (very) small deviation of measurement can be allowed when comparing an actual measurement to a standard measurement. To put it differently, when determining whether an actual measurement is equal to a standard measurement one tests if they are *sufficiently equal.* We define $\doteq$ to be an operator that measures if a measurement is sufficiently equal to a standard measurement[7]. In other words, a measurement is accurate (sufficiently equal to a standard measurement) if $\mathsf{M}(a, s, d) \doteq \mathsf{M}(a, s, d_s)$.

Last but not least, we can relate the above discussion to the uncertainty involved with measurements. This uncertainty is caused by two things: the accuracy (or, if you wish, the quality) of the measurement devices and the many possible situations in which they are used. The following illustrates what we mean by this. Let d be a measurement device and d_s be a standard measurement device for the same domain. This measurements of device d

[7]In a more elaborate theory it would be interesting to parameterize the $\doteq$ to be able to specify the allowable deviation, the tolerance. This is, however, beyond the scope of this dissertation.

can be tested against d_s in many (but not necessarily all) situations $S \subseteq \mathcal{SI}$. The accuracy of d is defined to be the average deviation of that device with respect to the situations in which it is tested:

$$\mathsf{Acc}(d) = \frac{\sum_{s \in S} \mathsf{M}(a,s,d) \overset{\circ}{=} \mathsf{M}(a,s,d_s)}{|S|}$$

Observe that in this treatment we have a single accuracy value for a measuring device. This may be a crude measure for some situations. For example, to measure very small volumes an extremely sensitive measuring device may be necessary. However, in other situations a few *cc* more or less won't matter. Our treatment of accuracy of measuring devices is similar to *Similarity theory* in general and the local-global principle in particular (see e.g., [AR99, MR01] for details).

Accuracy is the basis for defining the measurement uncertainty. That is, if we assert that (the value of) a property can be measured with a degree of certainty n then we mean that measurements done with this device are correct in $n\%$ of the situations.

The uncertainty involved with interpreting measurements is modeled similarly and makes use of linguistic variables. Let $\langle \mathcal{X}, T(\mathcal{X}), U, G, M \rangle$ be a linguistic variable. In the running example for this section, $\mathcal{X}$ represents the variable *volume of a mug* with term-set $T(\mathcal{X}) = \{\text{big}, \text{medium}, \text{small}\}$. We interpret the membership-degree for these linguistic values as the degree of certainty that we have in this specific interpretation of the actual measurement. Let $\mu_t : U \rightarrow [0 \ldots 1]$ denote the membership degree for the terms t in the term-set. To set the stage, consider the following running example:

Example 6.5.4 *In our example, the linguistic variable $\mathcal{X}$ denotes volume with term-set $\{$small, medium, big$\}$. The domain U represents the volume in* cc*'s. The membership function for the linguistic value 'big' is given by:*

$$\mu_b(u) = \begin{cases} 0 & u \leq 15 \\ \frac{1}{15}u - 1 & \textit{otherwise} \\ 1 & u \geq 30 \end{cases}$$

and is drawn in Figure 6.7. For ease of computation we have chosen the membership function to be linear.

In the running example we wish to answer the following question:

> Suppose I measure the volume of a mug to be $25cc$. What are the odds that this mug is considered to be big?

The answer to this question depends on the (accuracy of) measurements as previously described, but also on the interpretation of the linguistic value 'big'. The trick is to interpret the membership degree as certainty of interpretation. This requires a conversion of the (graph of the) membership degree function to a probability distribution.

By examining the increase of the surface under this membership function we get a cumulative probability distribution, provided that for each linguistic value v it holds that

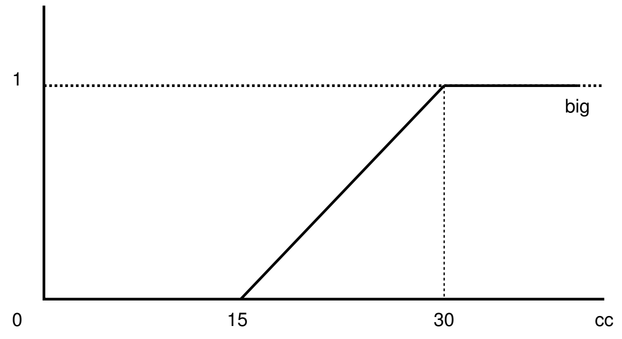


Figure 6.7: Probability profile for linguistic value 'big'

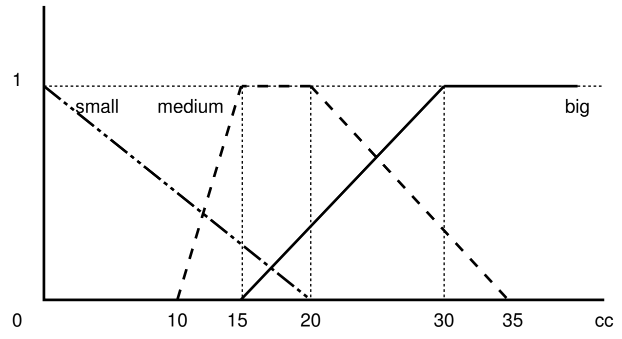


Figure 6.8: Probability profile for the values of the linguistic variable 'volume'

Axiom 37 $\int\limits_0^\infty \mu_v(u)\,du = 1$

In our example it is straightforward to verify that this indeed the case. The certainty for our interpretation given measured value u and linguistic value v is given by:

$$P_v^i(u) = \int\limits_0^u \mu_v(u)\,du$$

In our case, $P_b^i(25) = \frac{2}{3}$ indicates that we are approximately 67% certain that the contents of the mug will be assessed as 'big' and, consequently, that the quality of the mug will be 'high'.

The question that remains is: how can these probabilities be combined to calculate the certainty of our quality assessment? Continuing the previous example:

Example 6.5.5 *We use measuring device d to determine the contents of mug a in situation s. The accuracy of measurement $\mathsf{Acc}(d) = 0.9$. Let P^m denote this accuracy. The observed volume of this mug is $\mathsf{M}(a, s, d) = 25$cc. Before we can compute $P(25 = big)$ we must define the membership functions for the linguistic values 'small' and 'medium'. We* *presume these to be:*

$$P^i_s(u) = \begin{cases} 1 - \frac{1}{20}u & u \leq 20 \\ 0 & \mathit{otherwise} \end{cases}$$

$$P^i_m(u) = \begin{cases} 0 & u \leq 10, u > 35 \\ \frac{1}{5}u - 2 & 10 < u \leq 15 \\ 1 & 15 < u \leq 20 \\ \frac{35}{15} - \frac{1}{15}u & 20 < u \leq 35 \end{cases}$$

respectively. The membership functions are illustrated in Figure 6.8. It is easy to verify that:

- *the certainty that measured volume is indeed interpreted as 'big':* $P^i_b(25) = 0.67$*,*
- *the certainty that measured volume is indeed interpreted as 'medium':* $P^i_s(25) = 0.67$
- *the certainty that measured volume is indeed interpreted as 'small':* $P^i_m(25) = 0$

We still have to combine the uncertainty involved with measurements and uncertainty as a result of interpretations in order to compute the certainty with which we can assess that an artifact is of high quality for an actor. This is computed by multiplying the P^m with $P^i_v(u)$. For our toy example this would mean:

Example 6.5.6 *The certainty which we can assess that our mug is of high quality is:* $0.9 \times 0.67 = 0.6$*.*

6.6 Quality and the information market: aptness

In the dissertation so far we have mentioned the word *aptness* several times already. After presenting formal models for the information market, the information landscape and quality we are finally able to define what we mean by this term:

Definition 6.6.1 (Aptness) *The aptness of a data resource is a quality metric which expresses how suitable this data resource is for a searcher with a specific information need.*

This definition may appear fuzzy still, but given the material presented in previous chapters it is actually quite concrete. One must firstly realize that this metric is mainly used by (search) brokers on the information market (Chapter 2). As such the searcher is always known to this broker.

In Section 4.5.2 we already presented an example of aptness calculations under the assumption that queries issued by searchers capture their entire information need. In real-life situations this, obviously, is not always the case. Search brokers can also make use of such things as user profiles, past queries, search history and so on. One way or the other, however, this information about the searcher is available to the search broker.

In Section 2.3 we presented a taxonomy for value on the information market: structural value, informational value and emotional value. In our earlier examples of aptness computations we assumed that all aspects of a searcher's information need can be captured in terms of properties (Section 4.4.1). It appears to be straightforward for informational aspects and structural aspects of the information need but not so for emotional aspects. An example of an open research question in this respect would be: Which property would capture the fact that a searcher is *happy*? Answers to this, and similar, question(s) may come from the adaptive hypermedia / user modeling community (a brief overview of research in this field is presented in Section 1.2.7) and is unfortunately beyond the scope of this dissertation. However, we argue that explicitly taking non-informational aspects into account is the key to aptness based retrieval.

How can this be achieved, then? Using our model for quality as starting point one needs the following ingredients. The artifacts under consideration are *data resources.* To be able to measure properties of data resources one needs tools that measure (to what degree) the property support for data resources. In practice this means that search brokers can only deal with properties for which it has proper measuring equipment (i.e., software tools). For example, a search broker may not know how to measure whether a *Pdf* file has outgoing hyperlinks or not.

The domains in which properties can be expressed conform to the typing mechanism from *typed resource space* (Section 3.4). Requirements can be expressed as properties; this is an assumption we have to make to be able to do aptness computations. If fuzzy variables are used for properties then one must somehow learn membership function for these variables. In some cases one may even wish to personalize these membership functions. In other cases, averages per group of searchers may be sufficient.

An aptness computation, then, boils down to measuring which properties a data resource has and comparing these to the desired properties by the searcher. The remainder of this section is an extensive example that illustrates how aptness computations may work in practice.

The setting of this example is as follows. A quality assessment system (from now on: the system) is assigned the task to assess the quality of the newsletter of an online news site. The role of this site is 'informative medium'. In terms of our formalism: $n \in \mathcal{AF}$ denotes the newsletter and $r \in \mathcal{RO}$ denotes the role played by this site. Furthermore, $f = \langle n, r \rangle$ is the fulfillment for this newsletter.

The assessment has to take place for a certain actor $a \in \mathcal{AC}$. We know that the actor has three requirements with regard to this artifact: $\mathsf{Req}(f) = \{r_1, r_2, r_3\}$ which are verbalized as follows:

r_1: Data resource involved in Representation having type "newsletter"
r_2: Data resource having type "Pdf"
r_3: Data resource having attribution
  (with value "high" AND-ALSO having type "importance")

These requirements translate to our formalism as follows:

$r_1 = \langle p_1, c_1, e_1 \rangle$	where p_1 is the property type 'representation type', c_1 is the equality constraint and e_1 is the value expression 'newsletter
$r_2 = \langle p_2, c_2, e_2 \rangle$	where p_2 is the property type 'data resource type', c_2 also refers to the quality constraint and e_2 is the value expression 'Pdf' (which is a data resource type in the model for resource space in Chapter 3)
$r_3 = \langle p_3, c_3, e_3 \rangle$	where p_3 is the property type 'importance', c_3 again is the equality constraint and e_3 the value 'high'. Note that in this case the system must use a linguistic variable to represent this constraint since 'high' is a soft value. The underlying 'hard' domain for importance is chosen to be the *PageRank* metric.

To be able to make a quality assessment the system uses three measuring devices $d_1, d_2, d_3 \in \mathcal{MD}$, one for each constraint. The three measurements will be done in parallel; in other words, in one situation $s \in \mathcal{SI}$. Based on previous experiences and tests the system knows that:

d_1: is a software tool that is designed with the sole purpose of determining whether a given artifact is a newsletter or not. Furthermore, $\mathsf{Acc}(d_1) = 0.95$ which means that the system is able to correctly judge whether a given artifact is actually a newsletter in 95% of the situations.

d_2: is a tool that checks the (data resource) types of artifacts. This general purpose tool has been trained extensively on all known types and therefore $\mathsf{Acc}(d_2) = 1$ means that assessments are always correct.

d_3: is a highly complex tool. It assumes that the PageRank is a good measure for importance of artifacts but knows that this need not always be a 100% correct assumption; hence: $\mathsf{Acc}(d_3) = 0.9$.

As stated previously, the system uses a linguistic variable to express the values of the constraints. For r_1 and r_2 the membership function is straightforward; 1 if the condition is met and 0 if it isn't met. However, for r_3 the situation is a little more complex. The term-set for this variable is $\{low, average, high\}$ and the underlying domain $U = [0 \ldots 10]$ the domain for expressing PageRank. After careful consideration of the user-profile of a the system decides the following membership function for the linguistic value 'high':

$$\mu_{\text{high}}(u) = \begin{cases} 0 & 0 \leq u \leq 6 \\ \frac{1}{4}u - 1\frac{1}{2} & 6 < u \leq 10 \end{cases}$$

Finally, in situation s the system makes the following measurements:

$\mathsf{M}(n, s, d_1) = \textit{true}$:	which means that the system suggests that s is indeed a newsletter. Hence, the membership degree is 1.
$\mathsf{M}(n, s, d_2) = \textit{Pdf}$:	which means that the system suggests that s is a *Pdf* file. Hence, the membership degree is 1.
$\mathsf{M}(n, s, d_3) = 9$:	which means that the observed PageRank for n is 9. The membership degree, then, is 0.75.

Last but not least we can compute the certainty with which the system can assert that n is of high quality to a:

- $P_{r_1} = 0.95 \times 1 = 0.95$
- $P_{r_2} = 1 \times 1 = 1$
- $P_{r_3} = 0.9 \times 0.75 = 0.675$

Finally the total quality is the multiplication of these three certainties which results in 0.64. This should be interpreted as: the system is able to assert with 64% certainty that newsletter n is of high quality to actor a.

6.7 Quality of transformations

Using the aptness definition from the previous section it is relatively straightforward to study and define the quality of transformations. An interesting dichotomy is that of the internal quality of a transformation (how well does it perform its task) and the external quality of a transformation (how does the user perceive the effects of the transformations). A similar distinction is made in *recommender systems* where one distinguishes between the *internal* and *perceived* quality of recommendations.

Since we adopt a black-box approach to transformations, we are mainly interested in the external quality of transformations and the aptness metric enables us to compute it as follows:

Definition 6.7.1 (Quality of a transformation) *Quality of a transformation is measured by the expected increase of aptness of a data resource after this transformation has been applied to it. A positive score implies that the transformation is expected to increase the aptness of the data resource, whereas a negative score implies the inverse.*

In other words, to be able to compute the (external) quality of transformations we need to know both the wishes of the searcher, the aptness of the data resource (Section 6.6) and the effects of transformations (Section 4.4). In the remainder of this section we present a small example that illustrates the computation of the quality of transformations.

Let $e \in \mathcal{DR}$ be an artifact, r a role such that $f = \langle e, r \rangle$ a fulfillment. Furthermore, the requirements of a searcher are $\mathsf{Req}(f) = \{r_1, r_2\}$ such that:

$r_1 = \langle p_1, high \rangle$	p_1 a property represented by a linguistic variable with term-set $\{low, medium, high\}$ and an underlying domain of real numbers
$r_2 = \langle p_2, high \rangle$	p_2 a property represented by a linguistic variable with term-set $\{low, medium, high\}$ and an underlying domain of real numbers

The membership functions for the linguistic values "high" of both variables are respectively

$$\mu_{p_1,high}(u) = \begin{cases} 0 & 0 \leq u < 5 \\ \frac{1}{5}u - 1 & 5 \leq u < 10 \\ 1 & 10 \leq u \end{cases}$$

$$\mu_{p_2,high}(u) = \begin{cases} \frac{1}{15}u - 1 & 0 \leq u < 15 \\ 1 & 10 \leq u \end{cases}$$

Furthermore, let d_1 and d_2 be two perfect measuring devices with $\mathsf{Acc}(d_1) = \mathsf{Acc}(d_2) = 1$ and s be the situation in which measurements take place. The measurements and aptness computations are as follows:

$$\begin{aligned}
&\mathsf{M}(e, p_1, d_1) = 7 \text{ such that } \mu_{p_1,high}(7) = \tfrac{2}{5} \\
&\mathsf{M}(e, p_2, d_2) = 8 \text{ such that } \mu_{p_2,high}(8) = \tfrac{8}{15} \\
&P_{r_1} = 1 \times \tfrac{2}{5} = \tfrac{2}{5} \\
&P_{r_2} = 1 \times \tfrac{8}{15} = \tfrac{8}{15} \\
&\text{Aptness } = \tfrac{2}{5} \times \tfrac{8}{15} = \tfrac{16}{75} \approx 0.213
\end{aligned}$$

Assume that two transformations (either singelton or composed) exist to transform this artifact: $T_1, T_2 \in \mathcal{TR}$. For the first transformation:

$$\begin{aligned}
&\mathsf{M}(\overrightarrow{T_1}(e), p_1, d_1) = 10 \text{ such that } \mu_{p_1,high}(10) = 1 \\
&\mathsf{M}(\overrightarrow{T_1}(e), p_2, d_2) = 2 \text{ such that } \mu_{p_2,high}(2) = \tfrac{2}{15} \\
&P_{r_1} = 1 \times 1 = 1 \\
&P_{r_2} = 1 \times \tfrac{2}{15} = \tfrac{2}{15} \\
&\text{Aptness } = \tfrac{2}{15} \approx 0.133
\end{aligned}$$

Even though this transformation drastically improves the situation with respect to requirement r_1, it also seriously hampers the situation with respect to requirement r_2 which results in a lower aptness score. The quality of this transformation can now be computed as the relative increase in aptness score which equals $-\frac{3}{8}$. This negative score implies that this transformation is rejected since it only lowers the aptness score. For the second transformation we have:

$$\begin{aligned}
&\mathsf{M}(\overrightarrow{T_2}(e), p_1, d_1) = 8 \text{ such that } \mu_{p_1,high}(8) = \tfrac{3}{5} \\
&\mathsf{M}(\overrightarrow{T_2}(e), p_2, d_2) = 10 \text{ such that } \mu_{p_2,high}(10) = \tfrac{2}{3} \\
&P_{r_1} = 1 \times \tfrac{3}{5} = \tfrac{3}{5} \\
&P_{r_2} = 1 \times \tfrac{2}{3} = \tfrac{2}{3} \\
&\text{Aptness } = \tfrac{3}{5} \times \tfrac{2}{3} = \tfrac{2}{5} = 0.4
\end{aligned}$$

In this case the transformation improves upon the original data resources with respect to both requirement r_1 and r_2. In this case the quality of the transformation is $\frac{7}{8}$. The fact that this magnitude is positive implies that the transformation does increase the aptness of the original data resource

6.8 Conclusion

In this chapter we have studied the notion of quality in the context of the Web. Our main goal in this endeavor is to answer the research question "how can quality on the Web be measured". Even more, we have discussed several issues related to implementing this quality notion in practice.

From a literature study on quality we have learned that there are two main aspects to quality: quality in the sense of attributes (of artifacts), and quality in the sense of desirability. The relation between these two seems obvious; if qualities of an artifact are "just right" for a certain actor then this actor will judge the artifact to be of high quality. This idea can also be applied to resources on the information market which leads to the notion of aptness.

In other words, in order to compute the aptness of a resource for a specific actor we must know / have several things. First of all, we must know the requirements of the actor (searcher) with respect to this resource. In Chapter 2 we presented a taxonomy for values consisting of emotional value, structural value, and informational value. The first value seems tricky to work with to say the last. Results from the field of user modeling may help in this respect, but this is beyond the scope of this dissertation. The language for expressing properties as presented in Chapter 3 can be used to express requirements with respect to the other two value domains.

Secondly, we must know the support for properties of this data resource. To this end, measuring instruments are needed. In the computation of the aptness of resources, the accuracy of these instruments must be taken into account. Last but not least it may be the case that the requirements of the searcher with respect to properties are expressed using soft, linguistic values. In this case, membership functions (on an underlying hard / concrete value domain) for these linguistic values must be elicitated as well. These can either be specified for individual searchers, or learned for groups of (similar) searchers.

CHAPTER 7 Conclusion

In this dissertation we focussed on a new, generic view on retrieval on the Web which we dubbed "aptness based retrieval". In a nutshell, our approach is inspired by economic theory; the aptness of a data resource should be considered as a generic metric for expressing the usefulness of data resources on the Web for a searcher in a specific situation. As such there seems a clear relation with (micro) economic theory and the literature on quality. As such, the aptness calculations are similar to calculating the value of assets on economic markets. This observation was the main inspiration for formulating our research questions in Section 1.4. Our central research question was:

> What is aptness-based search on the Web and how can it be achieved?

To be able to answer this goal we have formulated 5 research questions. In this concluding chapter we will summarize our findings by zooming in on these questions one at a time. For each question we firstly summarize our findings after which we formulate a more concise answer to the question under consideration. After these concluding remarks in Section 7.1 we will relate our findings to current developments on the Web in Section 7.2. Finally, we will present suggestions for future research, most prominently including (experimental) validation of our work in Section 7.3.

7.1 Research questions

What is the information market? – This first question was inspired by the observation that "search on the web" can, to some extent, be seen as an information market. Our ambition was to explore this parallel and present a formal model for the information market.

Our first step was to explore (economic) markets in general. In our opinion the relevant concepts for our purposes are: players, transactions between players and the notion of value. More precisely, players exchange assets because they expect to gain something from this exchange; the axiom of rational behavior. This *gain* is in terms of the value of the assets being exchanged. In our model, the value of an asset to a player is something abstract; the value of assets can be compared but there does not seem to be a concrete domain in which value can be expressed. After all, money is just another asset that can be exchanged. Even though this domain is abstract, we *do* know that the value of some assets can be compared. In that respect the $>$ operator provides a (partial) order on the value of assets to a certain player.

We have also introduced a model concept called *transactor* which, in essence, represents the view of a single player in a transaction. For example, the fact that a player p exchanges asset a for b in a transaction is a transactor. This transactor view is particularly convenient for modeling the information market.

With the market model in place, we proceeded to apply this model to the specific situation of search on the Web using the transactor view as a starting point. In case of the information market then, an important class of assets are *data resources* which, hopefully, convey information for searchers. From the searcher point of view, transactions on the Web involve an investment of such things as time, effort, and possibly money to receive resources which may convey the information s/he needs. In order to assess if the resource is, to this searcher, more valuable than the investment we need a complex value mechanism.

Our value mechanism bares similarity to the three aspects of architecture as formulated by the Roman architect Vitruvius; *utilitas* corresponds to our informational aspect of value, *firmitas* corresponds to our structural aspect of value, and *venustas* corresponds to the emotional aspect of value. This complex value domain can be used to study transactors. Indeed, on the information market the axiom of rational behavior still has to hold. This implies that players expect to gain something when they publish resources on the Web, or search the Web for information. This brings us to the issue of transactions on the information market.

In our opinion, most transactions on the information market are facilitated by a (search) broker. Even more, transactions usually consist of transferring a *copy* of a data resource from the publisher to the searcher. More specifically, we observed that there is a time aspect to transformations (there may be a large period of time between the moment of publishing a resource and the moment it is actually consumed by a searcher) and that transactions are one-to-many (because many copies of that resource can be made).

One of the most important tasks of a search broker is to locate the proper resources for a searcher, given the query of this searcher. In economic sense, this means that the broker must value resources for searchers. As such, the broker must take the informational aspects, structural aspects and emotional aspects of valuation into account. In essence, search brokers determine the value of resources based on the query of searchers. This query is an extensional representation of the actual information need of the searcher (i.e., the actual intension). We hypothesize that this query can be modeled as a set of constraints on the properties of the ideal resource that the searcher desires. This leads us to the second research question in which we study, amongst other things, these properties. In short, the answer to our first research question is:

The information market is a generic term to denote the process of (information) exchange on the Web. The transactions on these markets tend to be facilitated by brokers whose task it is to value resources for searches based on their query which is an explicitation of their information need. The value of a resource to a player is personal and can only be expressed in its comparison to the value of other resources. Even more, we distinguish three aspects / dimensions of value: informational value, structural value and emotional value.

What is information supply? – The rationale behind our second question was the following observation: if we understand exactly what the information landscape looks like then we will be able to create better characterizations of the resources in this landscape. This, in turn, caters for a richer query language and more clues for brokers on the information market to value resources for searchers. Similar to our first research question, our ambition is to gain a deeper understanding of what the information landscape looks like. As such we adopt a modeling approach.

An interesting perspective on this field of research is provided by the Semantic Web community in general, and the open standards (such as RDF and OWL) maintained by the World Wide Web consortium (W3C) in particular. In our view these open standards are particularly useful for designing and implementing (brokering) tools on the Web.Even more, they provide valuable clues for our model for information supply. Even though we "borrow" ideas, especially from the RDF standard, we have chosen to present our model as an axiomatic theory rather than a RDFS schema or an OWL ontology. Note, however, that the concepts in our model can easily be used to construct such ontology or schema at implementation time.

Our model is based on the following observations. Firstly, all data resources on the Web are about something. These "somethings" are dubbed *information resources* in our model. The aboutness relation is used to assess the informational value aspects from our model for information markets. In practice this relation can be implemented in different ways. In Section 2.4 we have presented an infon algebra which shows a theoretical perspective on aboutness relations in which we assume the existence of information particles called *infons* and a specialization operator. Also, the observation that a data resource is about an information resource is dubbed a *representation* in our model. An example representation, thus, is the observation that a document is about a certain person, or that a picture is about flowers. There are, obviously, different kinds of representations (details on the typing mechanism follow). Examples of representation types include: image of, document about, song about, etcetera. This allows us to give meaning to how a data resource "implements" information resources.

The second observation behind our model is the fact that *connections* play an important role on the Web. We distinguish between two kinds of connections: *relations* connect data resources to data resources. The most prominent example of relations on the Web is of course the hyperlink mechanism but other forms of relations exist as well. The second type of connection is the *attribution* which connects data resources to *data values*. These data values can be considered literals which have no meaning by themselves. Typical examples would include €25 and 2.4. This distinction between literals and entities on the Web is also used in e.g., the RDF standard.

Data resources, data values, representations, relations and attributions are referred to as resource space elements. These resource space elements form the basis for a complex typing mechanism which is tightly interwoven with the rest of our model. As such the typing mechanism should be considered the icing on the cake. One of the most interesting aspects of the typing mechanism is the treatment of complex types. Simply put, a data resource type is considered to be complex if instances (data resources) of this type are constructed by means of other data resources or data values. For example, a *Zip* file is a complex data resource and, therefore, *Zip* a complex type.

Summarizing, information supply is, in our view, a highly connected, heterogeneous collection of data resources which may have many different properties. These properties can be expressed using the aboutness (information resources, representations) concept, connections and typing. To that end we have presented a language that builds on these concepts in Section 3.5. This language can both be used for expressing the properties of data resources and for querying information supply. In summary, the answer to this research question is:

> *Information supply (on the Web) is the totality of data resources that may provide information to searchers with a certain information need. We assume that the resources are always about something and may have different properties. These properties can be formulated in terms of (typed) resource space elements: representations, attributions / attributed data values and relations.*

How can information supply be manipulated? – This question was inspired mainly by two observations. First of all, many players (brokers) on the information market already manipulate resources in their interactions with searchers as the examples in Section 7.2 will show. Also, in our model for markets we observed that brokers are value adding by definition. In our opinion, meaningful manipulation of resources on the Web can be value adding especially if it is done automatically.

Our ambition with the *how*-question was, again, to gain a deeper understanding of what transformations on the Web are. In studying these transformations we adopted a black box modeling approach. That is, we did not study transformations at the source-code level. Instead we set out to describe what transformations are and how they behave in general.

A first observation in this respect is the fact that transformations are described at the type level, but are executed at the instance level. For example, a transformation is said to transform *Pdf* files to *Html* (type level). When this transformation is executed then actual data resources (instance level) are involved. The first step in describing transformations is therefore the typing level. In our view, transformations have an input type and an output type. As such it is possible to construct a directed labelled graph in which the nodes are data resource types and the edges denote transformations. Another aspect of transformations that is relevant in this respect is the observation that they can be combined by executing one after the other. In the graph analogy this means that transformations are constructed by graph traversals. Important transformation classes are type casting transformations and complex transformations (i.e., transformations that operate on complex data resources) such as deep transformations and removing transformations.

Transformations may have an effect on *properties* of the data resources on which they operate. Examples of such properties include the typing property (i.e., a transformation may alter the type of a data resource), the representation type (i.e., a transformation may change a *Textual description* to a *Summary*) etcetera. These properties are expressed in a language that is based on our model for information supply. By comparing the properties of a data resource before and after it has been subjected to a transformation we may learn whether the transformation *alters*, *introduced*, *removes* or is *neutral* with respect to the support for this specific property.

In this case we must also make a distinction between the type level and the instance level. It may be that a transformation *alters* the support for a property in one case, yet it is *neutral* in another case. This is, for example, the case when a transformation sets the resolution of an image to a specific value. This complicates the reasoning about transformations. In terms of our graph analogy it means that we must somehow combine the facts that we learn while actually executing transformations to be able to annotate the edges in the graph with the knowledge about effects on properties. In Section 4.5.1 we have presented an algorithm to learn these effects. Even more, in Section 4.5.2 we show how transformations can be selected/ composed run-time to achieve a certain effect. The assumption for this algorithm is that the user specifies his query as a set of constraints on the properties of the desired data resource. By carefully examining the properties we may be able to select a series of transformations that increase the aptness of the data resource under consideration.

In a practical situation there may be many different types and transformations. This means that our transformation graph will have a high level of complexity. To put it differently: transformations may be combined in many (possibly: infinitely many) different ways. A simple depth-first exhaustive search may therefore not be feasible. In Chapter 5 we have presented some small experiments concerning transformation selection and its application in a retrieval setting. All in all, the answer to our question is:

> *Transformations are systems (or, in practical terms, pieces of software) that manipulate property support of data resources. The knowledge about transformations can be modeled as a labelled directed graph where nodes are data resource types and edges are transformations. These edges can be annotated with the knowledge that we have about the effects of the present transformation on properties. Transformation selection, then, is comparable to a variant of the shortest path algorithm.*

How can quality on the Web be measured? – This question is inspired by the observation that, in essence, *aptness* of a resource to a search can be seen as a quality metric. Our ambition with this question was twofold. Firstly we wanted to get to grips with the notion of quality itself as it is used in many different fields. As such we wanted to conduct a thorough literature survey and come up with a formalism in which we could express what the quality of an artifact / asset is to an actor. Secondly we wanted to come up with a metric for the quality of an artifact to an actor. Such a metric can then, obviously, also be applied to Web resources.

Our survey on the notion of quality included many different fields ranging from philosophy and library information systems to e-commerce and operations management. In Section 6.3 we synthesized the following observation about the quality notion; there are two views on quality: quality in the sense of properties (the qualities of an artifact) and quality in the sense of desirability (how good is an artifact for a player). This duality is also reflected in our model.

In our view, the quality of an artifact to a player depends on the role of the artifact. Observe that quality assessment is highly personal, similar to the concept of value. For example, a mug may be very beautiful and thus of high quality as an art object. On the other hand, it may be useless, and thus of low quality, as a drinking device. Secondly, the quality (desirability) of an artifact depends on the qualities (properties) of this artifact. In case of the mug example, these qualities can be such things as the volume of the mug, color, etcetera. In case of Web resources, these qualities translate to the notion of *properties* which we mentioned previously (and introduced formally in Section 4.4.1). In our model we take these roles and qualities into account when assessing quality.

To be able to make this quality notion computable we assume that players / actors have (implicit or explicit) requirements which they use to come to a quality assessment. Examples of such requirements are: "the mug must have a large volume", "the mug must have a volume of at least 20 *cc*", "the mug must have a volume that is at least as big as the volume of my neighbor's mug". Quality is computed by comparing the actual qualities that an artifact has with the requirements on them. This computation can be refined by, for example, allowing the requirements to be weighted.

The measurement of qualities / properties of artifacts is a problem in itself. These measurements involve two types of uncertainty: uncertainty pertaining to the measurement itself and uncertainty pertaining to the interpretation of the requirements. The former refers to issues pertaining to the accuracy of measuring devices, the latter with semantic issues such as the question "what does the actor mean with *high* volume?". The former is modeled using a measurement deviation, the latter is modeled using the concept of linguistic variables. Both uncertainties have to be taken into account as well. In summary:

> *Quality of artifacts to actors expresses how good an artifact is according to a specific actor. Several factors have to be taken into account to make quality assessments computable: the requirements of the actor with respect to the properties of the artifact, measurement uncertainty of these properties, and interpretation uncertainty.*

What is aptness-based search and how can it work in practice? – The ambition for this last question is to provide an answer to our research goal. As such, we wanted to combine the answers to the previous research questions as well as describe the issues that have to be dealt with when implementing aptness-based search systems in practice.

In our model for the information market we explained that the task of search brokers is to value resources for searchers with respect to their information need. In traditional (Web) retrieval systems, the main (or some times: sole) criterium for valuing is typicality. We argued that this *informational dimension* is only one aspect for valuing resources on the Web. The two other dimensions in our value model are the *informational dimension* and the *structural dimension.* In other words, we feel that "topical aspects" are not enough for valuing resources. Even more, brokers have to add value, one way or the other. One way to do this is by deploying transformations.

Valuing resources is based on a set of (explicit or implicit) requirements that an actor has with respect to a resource. To be able to model and represent these requirements in a uniform manner we developed a conceptual model for information supply. Using this we can describe exactly what the constraints are. This leaves us with the problem of computing the value of resources to searchers. In Section 2.2.3 we observed that there is no concrete domain in which we can express the value of assets (and thus: resources). The notion of value can be seen as a partial order on resources. However, in Chapter 6 we presented a model for quality based on the observation that the quality (how good it is to an actor) of a resource depends on its qualities (properties). We showed how the quality of an artifact can be computed by comparing its property support to the requirements by the actor, taking into account the uncertainty related to measurements and interpretation of the constraints. This quality metric, applied to resources on the Web, is an *aptness* metric. Thus part of the answer to our research question is:

> *Aptness is a quality metric for resources on the Web that takes into account the (fuzzy) requirements of searchers as well as uncertainty related to the interpretation of these requirements and the uncertainty related to measuring the support for properties of the resources under consideration.*

This leaves us with the issue of "making it work in practice". Our ambition was, initially, to develop a prototype search engine that was capable of doing aptness-based retrieval on a Web collection. It turned out that this was too ambitious for this dissertation. Instead we did a (smaller) experiment with a small prototype system as described in Section 5.2. From this experiment we learned the following.

First of all, from the searcher point of view it would be best to search the *intensional Web*; that is, the Web as is (the *extensional Web*) and everything that can be generated from it by means of transformations. This intensional Web, however, is infinitely large and thus not computable. Thus, another strategy must be adopted. We have described an approach that uses a push-down selection mechanism where we firstly select resources that are topically relevant and then try to increase their aptness by means of transformations. In Section 4.5 we showed a reference model for searching that includes transformations.

The second lesson resulting from these experiments pertains to the properties of data resources. With our language for information supply, (infinitely many) different properties can be described. If we allow searchers to specify "free style" properties (i.e., searchers are allowed to formulate properties themselves using our language) then we also need infinitely many tools to check for the support of these properties. It does not seem feasible with this many properties to provide the tools that check whether a data resource has a certain property or not. It is likely that semantic web technologies and standards such as RDF and OWL prove to be helpful in this respect; much information about support for properties may be learned or inferred from annotations (Section 1.2.6).

This leads to the third lesson. Searchers must be able to somehow specify the constraints / properties they use to assess the aptness of data resources to the search broker. In this respect, user modeling should be part of the process since user models may provide valuable information about searchers. Unfortunately user modeling is not part of our current work and thus provides an interesting research topic for the future as we will see in Section 7.3. Related to this is the problem of query formulation. As we already observed, it may be possible to allow searchers to specify queries in our language for information supply. But is this really desirable? How, then, should a user interface be designed to assist the searcher in formulating queries as good as possible? This is also a topic for future research. Summarizing:

> *Many issues pertaining the design and implementation of aptness-based retrieval are still open research topics. However, from our initial experiments we have learned that transformations play a central role in implementing a push-down selection based approach to aptness-based retrieval.*

7.2 Exploring the Web

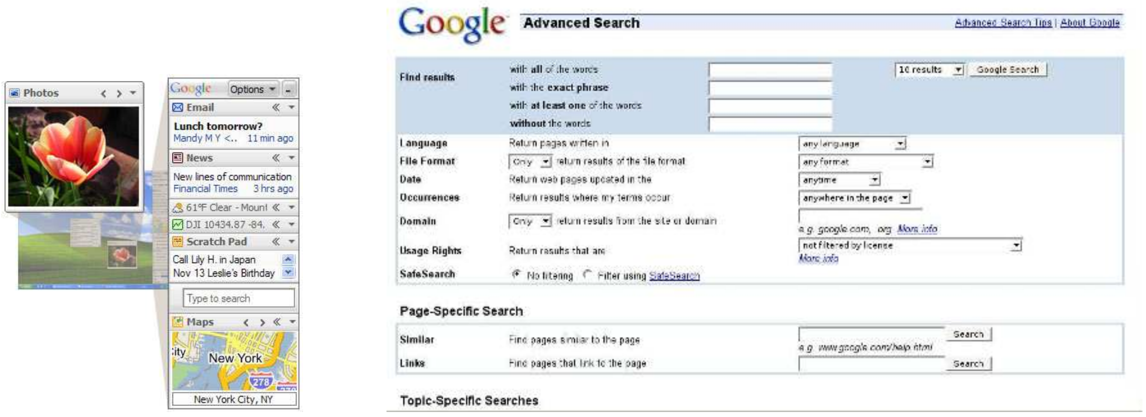
The goal of this section is to see how the material presented in the previous chapters "works in practice" with some case studies. That is, we consider several players in the information market from the perspective of the theory introduced in this dissertation.

7.2.1 Google

One of the most important players on the Web these days is, beyond a doubt, *Google*. In a few years time this search engine grew from an academic initiative to one of the five most popular sites on the Internet with 5,680 full time employees as of December 31, 2005. Its mission is "to organize the world's information and make it universally accessible and useful".

From this mission we conclude that *Google* should be considered a search broker (see Definition 2.2.5 on page 23) on the information market. This broker adds value by making information universally accessible to searchers in a useful way. This universal-principle implies that non-informational aspects are taken into account by *Google*:

- The diversity of information (assets!) is enormous and ranges from web pages to stock quotes, maps, news headlines and so on. From the perspective of our model for information supply (Chapter 3) these can be considered representation types. Interestingly, *Google* has different search interfaces for different kinds of resources such as *Google groups*, book-search, blog search, maps, and so on.



Lunch tomorrow?
Mandy M Y <.. 11 min ago

New lines of communication
Financial Times 3 hrs ago

61°F Clear - Mount

DJI 10434.87 -84.

Call Lily H. in Japan
Nov 13 Leslie's Birthday

New York City, NY

Advanced Search Advanced Search Tips | About Google

Find results	with **all** of the words		10 results · Google Search
	with the **exact phrase**		
	with **at least one** of the words		
	without the words		
Language	Return pages written in	any language	
File Format	Only · return results of the file format	any format	
Date	Return web pages updated in the	anytime	
Occurrences	Return results where my terms occur	anywhere in the page	
Domain	Only · return results from the site or domain	e.g. google.com, .org More info	
Usage Rights	Return results that are	not filtered by license More info	
SafeSearch	[✓ No filtering] Filter using SafeSearch		

Page-Specific Search

Similar	Find pages similar to the page	e.g. www.google.com/help.html	Search
Links	Find pages that link to the page		Search

Topic-Specific Searches

(a) (b)

Figure 7.1: The *Google* interface

- *Google* offers access to over 1 billion Usenet messages as well as other data resources with dozens of data resources types including *Html*, *Pdf*, *Word*, etcetera.

- Search services are offered not only via the browser (i.e., `google.com`) but also with a variety of other devices such as the (windows) desktop, mobile phone and so on. *Google*'s desktop interface is shown in Figure 7.1a. Each of these requires a separate interface and specific search capabilities.

- *Google* is also value adding for suppliers of information on the Web, for example via advertisements on the result pages. Interestingly, the selection for the advertisements that are actually shown are based on the current query of the searcher.

- Support for several properties (Section 4.4.1) of data resources are also taken into account. For example, the advanced search (shown in Figure 7.1b) allows the user to specify such things as *last modification date*, *file type* and *language*.

- There is (limited) support for transformations (Chapter 4). For some data resource types the searcher is offered different data resource types as an alternative. For example, *Pdf* files can also be viewed in *Html* format.

It seems that, apart from financial / economic motivations, *Google* really strives towards aptness based retrieval on the Web. In our view, *Google* and its users can benefit from our theory in several ways. The most prominent example in this respect is the fact that more different data resource types can be indexed by using transformations. Also, as personalization techniques improve, *Google* will be able to achieve more and more progress in making "emotional value" (Section 2.3) computable; *Google* will be better in learning the particularities of individual searchers and adapt its behavior to them in order to better satisfy the searchers needs. Customized advertisements are, in this respect, only the tip of the iceberg.

7.2.2 GMail

Google also offers an E-mail service: *GMail*. This service is slowly evolving from "just E-mail" to a one-stop solution to Web-based communications. Some interesting features of *GMail* from the perspective of this service are:

- *GMail* offers two services: E-mail and chat. Both services are seen as conversations. In terms of our reference model for information supply, this means that *E-mail*, *chat*, and *conversation* are data resource types. Even more, *conversation* is a complex data resource type. The interface is shown in Figure 7.2b.

- E-mail (and chat) conversations are stored / archived in a way that makes it searchable using a powerful search mechanism. One of the motto's of *GMail* is: "search, don't sort". To this end a very simple, yet powerful, approach is deployed where the searcher specifies several properties of this information need. A typical query would be:

  cookies from: erik after: 2006/03/02

  Which could easily be translated to our LISA-D query language (see Section 3.5).

  ```
  Data Resource about "cookies" AND-ALSO having Attribute
     (of type "sender" with Data Value "erik")
  AND-ALSO having Attribute
     (of type "Arrival date" with Data Value > "2006/03/02")
  ```

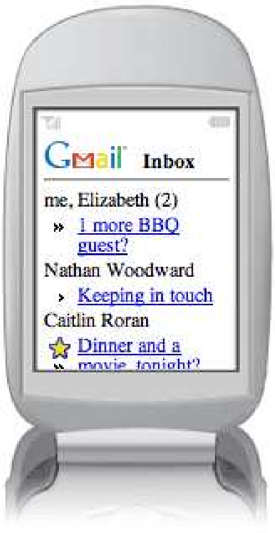
- *GMail*, at least the E-mail part, is available in many different situations and from a wide range of browsers and devices. For example, Figure 7.2a shows the mobile-phone interface to *GMail*. This transformation of the interface, based on the current context of the user makes *GMail* a flexible and powerfull tool where users have access to their E-mail almost "any time, any place, anywhere".

The *GMail* approach to conversations on the Web (which include both chat and E-mail) also has a aptness-aspect to it; it tries to deliver mail and chat services any time any place anywhere with additional (value adding) services such as searching.

7.2.3 Rijksmuseum Amsterdam

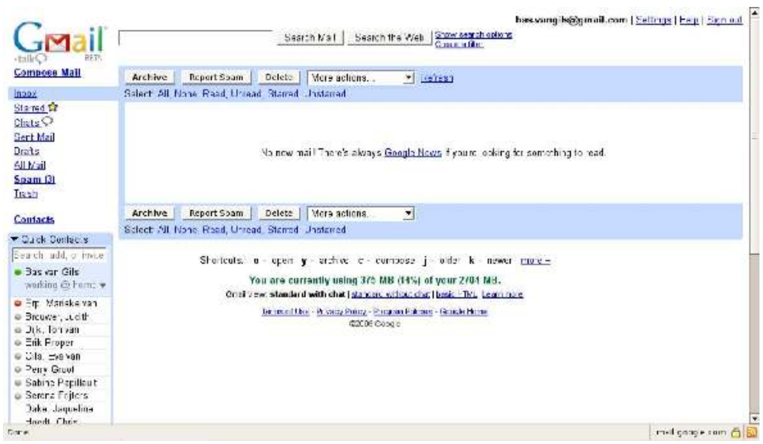
Presentation of data in a way that suits (potential) consumers best can be achieved with an extensive transformation framework. This can, for example, be seen in many museums which offer access to information in several ways. For example:

> a static virtual museum will offer the same "guided tour" and the same narration to visitors with very different goals and background knowledge.
>
> —*Taken from: [Bru01]*



me, Elizabeth (2)
» 1 more BBQ guest?
Nathan Woodward
› Keeping in touch
Caitlin Roran
Dinner and a movie tonight?

(a)



No new mail! There's always Google News if you're looking for something to read.

Shortcuts: o - open y - archive c - compose j - older k - newer more »

You are currently using 375 MB (14%) of your 2701 MB.

Gmail view: standard with chat | standard without chat | basic HTML Learn more

©2006 Google

(b)

Figure 7.2: The *GMail* interface

In this respect, the Rijksmuseum Amsterdam has also recognized the need to be able to combine and present information in a flexible manner. The Rijksmuseum has a large collection of multi-modal resources ranging from images, textual documents, video and animation which are in many cases interconnected.

People from different backgrounds (see Section 2.3) are likely to have different interests and information needs. For example: laymen might want to know the who created a certain painting, when the painting was created and what it is about. Art students, on the other hand, might be interested also in forerunners of the artist, details about the period in history, etcetera.

It is a huge challenge for the designers of the museums (web-based) information systems to cater for these different kinds of information needs; it seems impossible to deal with *all* these aspects. According to [ROH04, ROH05, HO06] the solution to this problem is the automatic generation of multimedia presentations so that the right information is presented to a particular user in a compelling and useful way.

The basic problem for the information systems of the Rijksmuseum is: dynamically generate (the presentation of) resources based on user requests (query). This generation aspect suggests that transformations are used which, in turn, requires thorough knowledge of the underlying collection of resources. This knowledge can be provided by annotations which is an issue in itself (See e.g., [Hop03]). At the conceptual level our model for information supply (Chapter 3) can be used. At the technical / implementation level Semantic Web technology lies at the heart of the information systems of the Rijksmuseum (Section 1.2.6). The (relations between) resources of the museum are annotated using RDF annotations. Even more, XSLT transformations allow the transformation of machine readable annotations into human readable form which makes them more *apt* for searchers.

For presentation purposes the system makes use of a global interface and a local interface to present information to searchers. The global interface shows a broad and large-scale view of the RDF repositories with annotations of the resources. The local interface shows richer details of the resource under consideration. Thus, based on a query the system selects resources and their annotations. These, in turn, are presented in a way that suits the present user as best as possible using transformation. Even more, the combination of the global and the local interface offers the much desired flexibility from the searcher point of view.

7.2.4 ACM Digital Library

The digital library of the association for computing machinery (ACM-DL) is a vast collection of citations and full text from ACM journal and newsletter articles and conference proceedings. These citations for a scientific paper may consist of:

- abstracts
- citings and references
- index terms from ACM's Computing Classification System (CCS)
- reviews from ACM's Computing Reviews

Some of this information is available for free. Other information (such as the full papers in *PDF* form) is only available to ACM members (membership costs a fee). The resources in this digital library are organized in such a manner that they can both be browsed or searched.

The combination of browsing and searching offers another approach to aptness: in the ACM-DL searchers can choose how they want to locate the resources (publications) they are interested in. Some times a keyword-based search (or meta-data search using the advanced search interface) may be preferable over browsing and vice versa.

A second consideration in this case is the fact that different views on the resources in the ACM-DL are given, once a specific resource is selected. These conform to the representation types in our model:

- A high-level overview of the publication and its context, consisting of bibliographic data, index terms, citings and references and so on
- the paper in *PDF/Html* form
- the bibliographic data in *BiBTeX/EndNote/ACM Ref* form

After the examples presented in this dissertation (especially Section 5.2) it would seem apparent that these are generated by means of transformations. This, however, is only partially true. Indeed, the first view is generated from the data of the other views. Even more, the data resource types for the bibliopgrahic data are probably generated from a single database.

In terms of the information market, ACM-DL can be considered a broker since it adds value in several ways. Firstly, the simple fact that a large number of scientific papers are accessible from a single interface is valuable for scientists. It makes it a lot easier to locate material on a certain subject / to keep ones knowledge on a particular subject up to date and so on. For the authors of the paper, the digital library is valuable also for exactly the same reason: if a publication is listed in this digital library then the odds that it is found by interested readers are higher (in comparison with publishing the paper only in a book in paper form as was the case only a few years ago). The fact that people find the services of the ACM-DL valuable is stressed by the fact that they are often willing to pay a fee.

7.3 Future research

In this last section of this dissertation we present several suggestions for future research. In this dissertation we studied the information market in general and aptness based retrieval on the information market in particular. In studying the information market we used *information supply* as a starting point (Section 1.1). Another option would have been to start with *information demand*, signified by searchers and their information need. This field is generally referred to as user modeling (See also Section 1.2.7).

Due to our choice to start with information supply rather than information demand we "know" very little of searchers. This is particularly inconvenient since, for aptness based retrieval, it is important to have as much information about searchers as possible. After all, this information determines which resources on the Web are apt (or to what extent they are apt). This urges us to look into the field of user modeling in the near future. Specific questions in this respect are:

- Which aspects of searchers are of interest for aptness based retrieval?
- Can these aspects always be translated to property constraints on resources on the information market?
- How can these aspects be learned?

Another issue, which is closely related to the above, is query formulation. One option is to let searchers learn our language for information supply and "force" them to use it. This does not seem feasible for two reasons. First of all, searchers may be unwilling to learn a new (and fairly complex) language. Secondly, allowing users to formulate queries (and thus constraints on properties) will result in infinitely many different properties that a retrieval system has to deal with. For each of these properties, a tool must be available to check the support for this particular property. These tools may simply not be available. Control over the properties that the system knows about is, thus, of the essence. In this respect two aspects of query formulation have to be taken into account. Firstly, a good balance between freedom for searchers and control for the retrieval system has to be found.

Secondly, user interface design has to be such that users can easily formulate their queries. We intend to investigate where a multi-dimensional form of *query by navigation* may be helpfull [Ber98, BBWW98].

Another interesting topic that needs further investigation concerns the actual implementation of a retrieval system, particularly the characterization of information supply and the possibility of decentralization. To start with the former, we want to be able to calculate property support for the properties that our system knows about. As such we may be able to use RDF annotations or OWL for inferences. Furthermore, it seems to be a good idea to use the *service oriented computing* (SOC) paradigm when implementing a search system. In this respect we intend to look into a service architecture with, for example, a search service, a transformation service, a user profile service, etcetera. Even more, these services may be replicated so that they are available from several hosts to increase the robustness and reliability of the search system.

Last but not least, we need to also consider the validity of our models (for the information market, for information supply, for transformations, and for quality) in an experimental setting. One way to do this is by means of a *population check*, which boils down to trying to find real-world cases that do not fit in our model. For the purposes of this dissertation we have presented many examples which illustrate how real-world situations fit in our models. however, full-blown and large scale experiments are still lacking. For example, for our model for markets we could try to populate our model with transactions on, say, the stock market. Similarly, we could try to populate our model for information supply with resources from the university website, etcetera. Using these experiments we hope to improve our models and show the validity of our models and the necessity of aptness based retrieval on the Web.

APPENDIX A

Mathematical notation

Chapter 2

$\mathcal{AS}$	Assets
$\mathcal{PL}$	Players
$\mathcal{TA}$	Transactions
$\_\,[\_]\,\_ \subseteq \mathcal{AS} \times \mathcal{PL} \times \mathcal{AS}$	Transactor
$\_\,[\_]\,\_ \subseteq \mathcal{PL} \times \mathcal{AS} \times \mathcal{PL}$	Transactand
$T \in \mathcal{TA}$	Transaction, transactor view
$\widetilde{T} \in \mathcal{TA}$	Transaction, transactand view
$\mathsf{Participant}(a_1\,[p]\,a_2)$	Participant of a transactor
$\mathsf{Buyer}, \mathsf{Seller} : \mathcal{TA} \rightarrow \mathcal{PL}$	Buyser and seller of an asset
$\mathcal{VD}$	Value domain
$\mathsf{Val} : \mathcal{PL} \times \mathcal{AS} \rightarrow \mathcal{VD}$	Value of an asset to a player
$\mathcal{ST}$	Player states
$\mathsf{Id} : \mathcal{ST} \rightarrow \mathcal{PL}$	Identification of players by states
$s \ltimes T$	State after transaction completes
$\mathcal{I}$	Infons
$\mathcal{IF}$	Infon algebra
$\rightarrow$	Specialisation operator on infons
$\bot, \top$	Least and most meaningful infon

Chapter 3

$\mathcal{DR}$	Data resources
$\mathcal{IR}$	Information resources
$\mathcal{RP}$	Representations
$\mathsf{IRes} : \mathcal{RP} \rightarrow \mathcal{IR}$	Information resource of a representation

$\mathsf{DRes} : \mathcal{RP} \rightarrow \mathcal{DR}$	Data resource of a representation
$i \models d \triangleq \exists_r [\mathsf{IRes}(r) = i \wedge \mathsf{DRes}(r) = d]$	Shorthand for a representation
$\mathcal{RL}$	Relations
$\mathcal{DV}$	Data values
$\mathcal{AT}$	Attributions
$\mathcal{DL} \triangleq \mathcal{DR} \cup \mathcal{DV}$	Data elements
$\mathcal{CN} \triangleq \mathcal{RL} \cup \mathcal{AT}$	Connections
$\mathsf{Src}, \mathsf{Dst} : \mathcal{CN} \rightarrow \mathcal{DL}$	Source and destination of connections
$s \overset{c}{\rightsquigarrow} d \triangleq \mathsf{Src}(c) = s \wedge \mathsf{Dst}(c) = d$	Dereference connections
$s \rightsquigarrow d \triangleq \exists_c [s \overset{c}{\rightsquigarrow} d]$	Shorthand for connections
X_B	Extension of set X in resource base
$\mathcal{RE} \triangleq \mathcal{DL} \cup \mathcal{CN} \cup \mathcal{RP}$	Resource space elements
$\mathcal{TP}$	Types
$\mathsf{HasType} \subseteq \mathcal{RE} \times \mathcal{TP}$	Typing mechanism
$\tau(e) \triangleq \{t \mid e \mathsf{HasType}\, t\}$	Types of an element
$\tau(E) \triangleq \bigcup_{e \in E} \tau(e)$	Types of a set of elements
X_τ	Shorthand for types of a set
$\pi(t) \triangleq \{e \mid e \mathsf{HasType}\, t\}$	Population of a type
$\pi(T) \triangleq \bigcup_{t \in T} \pi(t)$	Population of a set of types
$\_ \xrightarrow{\_} \_ \subseteq \mathcal{TP} \times \mathcal{CN}_\tau \times \mathcal{TP}$	Dereference connection types
$\mathsf{Conn}(t_1) \triangleq \{t \mid \exists_{t_2} [t_1 \xrightarrow{t} t_2]\}$	Connection types of a data resource type
$\mathcal{AC}$	Accessors
$\mathsf{Act}(t) \triangleq \mathsf{Conn}(t) \cap \mathcal{AC}_\tau$	Accessor types of a data resource type
$\mathcal{TP}_c \triangleq \{t \mid \mathsf{Act}(t) \neq \emptyset\}$	Complex types
$\underline{\mathsf{SubOf}}, \mathsf{SubOf} \subseteq \mathcal{TP} \times \mathcal{TP}$	(Proper) subtyping
$\sim \subseteq \mathcal{TP} \times \mathcal{TP}$	Type relatedness

Chapter 4

$\mathcal{TR}$	Transformations
$\mathsf{SEM} : \mathcal{TR} \rightarrow (\mathcal{DR} \rightarrow \mathcal{DR})$	Semantics of transformations
$\overrightarrow{T} \triangleq \mathsf{SEM}(T)$	Shorthand for semantics
$\mathsf{Input}, \mathsf{Output} : \mathcal{TR} \rightarrow \mathcal{DR}_\tau$	Input and output type of transformations
$t_1 \xrightarrow{T} t_2 \triangleq \mathsf{Input}(T) = t_1 \wedge \mathsf{Output}(T) = t_2$	Shorthand for transformations
$\overrightarrow{T_1 \circ T_2}$	Transformation composition
ϱ	Semantics of a removing transformation
δ	Semantics of a deep transformation
ι	Semantics of a type-cast transformation
φ	Property
Φ	Language for properties
$\Gamma : \mathcal{DR} \times \Phi \rightarrow \wp(\mathcal{RE}^+)$	Support for a property
$\mathcal{EC}_i$	Effectclasses, instance level

$\mathcal{EC}_t$	Effectclasses, type level
$\mathsf{Effect} : \mathcal{TR} \times \mathcal{DR} \times \Phi \rightarrow \mathcal{EC}_i$	Effect of transformations, instance level
$\mathsf{Effect} : \mathcal{TR} \times \mathcal{DR}_\tau \times \Phi \rightarrow \mathcal{EC}_t$	Effect of transformations, type level
$\Psi : \mathcal{DR} \rightarrow \wp(\Phi \times \wp(\mathcal{RE}^+))$	Total property support of data resources
$\mathcal{U} : \Phi \rightarrow \mathbb{R}$	Preference assignment

Chapter 6

$\mathcal{AF}$	Artifacts
$\mathcal{RO}$	Roles
$\mathcal{FL}$	Fulfillments
$\mathsf{Artifact} : \mathcal{FL} \rightarrow \mathcal{AF}$	Artifact of a fulfillment
$\mathsf{Role} : \mathcal{FL} \rightarrow \mathcal{RO}$	Role of a fulfillment
$\mathcal{PT}$	Property types
$\mathcal{PD}$	Property domains
$\mathsf{Props} \subseteq \mathcal{RO} \times \mathcal{PT}$	Properties of a role
$\mathsf{PrDom} \subseteq \mathcal{PT} \times \mathcal{PD}$	Domains of a property type
$\mathcal{VL}$	Values
$\mathsf{Value} : \mathcal{PD} \rightarrow \mathcal{VL}$	Values of a property domain
$\mathsf{VlDom} : \mathcal{VL} \rightarrow \mathcal{PD}$	Domain of a value
$\mathcal{FA} \triangleq \mathcal{FL} \times \mathcal{PT}$	Fulfillment aspects
$\mathsf{ValAss} : \mathcal{FA} \rightarrow \mathcal{VL}$	Value assignment
$\mathcal{RQ}$	Requirements
$\mathcal{EX}$	Expressions
$\mathsf{Prop} : \mathcal{RQ} \rightarrow \mathcal{PT}$	Property of a requirement
$\mathcal{CS}$	Constraints
$\mathsf{Constr} : \mathcal{RQ} \rightarrow \mathcal{CS}$	Constraint of a requirement
$\mathsf{Expr} : \mathcal{RQ} \rightarrowtail \mathcal{EX}$	Expression of a requirement
$\mathcal{X}$	Linguistic variable
μ	Membership degree
$\mathcal{SI}$	Situations
$\mathcal{MD}$	Measuring devices
$\mathcal{MV}$	Measured values
$R = [\mathcal{AF} \times \mathcal{SI}] \rightarrowtail \mathcal{MV}$	Measuring device
M	Measurement
$\mathrel{\hat{=}}$	Measurement comparison
Acc	Accuracy of a measurement device

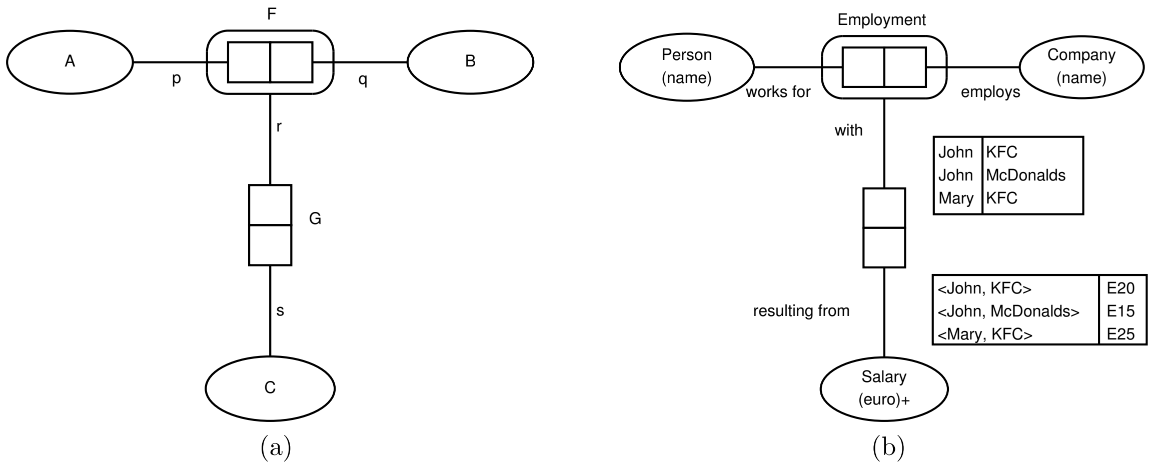
APPENDIX B ORM/PSM overview

This appendix outlines those parts of PSM and LISA-D that we used throughout this dissertation. As such, this appendix is based on the discussions in [HW93, HPW93, PW95]. We will make use of the example schema presented in Figure B.1.

Information structures capture the syntax of PSM. An information structure consists of the following basic components:

- A finite sit $\mathcal{P}$ of *predicators*.
  In Figure B.1a: $\mathcal{P} = \{p, q, r, s\}$.
- A nonempty set $\mathcal{O}$ of *object types*.
  In Figure B.1a: $\mathcal{O} = \{A, B, C, F, G\}$.
- A partition $\mathcal{F}$ of $\mathcal{P}$. Elements of $\mathcal{F}$ are called *fact types*, which are also object types.
  In Figure B.1a: $\mathcal{F} = \{F, G\}$.
- The functions $\mathsf{Fact} : \mathcal{P} \rightarrow \mathcal{F}$ and $\mathsf{Base} : \mathcal{P} \rightarrow \mathcal{O}$ relate predicators to their respective fact types and object types. Note that the Fact relation is derivable, it is defined as:
  $\mathsf{Fact}(p) = f \Leftrightarrow p \in f$
  For example, in Figure B.1a: $\mathsf{Fact}(p) = F$ and $\mathsf{Base}(p) = A$.
- in PSM a distinction is made between specialization (Spec, denoted as a bold arrow in PSM schema) and generalization (Gen, denoted as a dotted arrow in PSM schema). A full discussion of this topic is beyond the scope of this dissertation. The interested reader is referred to [HW93].

An information structure such as Figure B.1a is used as a frame for some part of the world, the universe of discourse (UoD). The state of the UoD corresponds to a population of the information structure. The population π of an information structure $\mathcal{I}$ is the assignment of sets of instances to the object types in $\mathcal{O}$; $\pi : \mathcal{O} \rightarrow \wp(\Omega)$, where Ω denotes the universe of all



John	KFC
John	McDonalds
Mary	KFC

<John, KFC>	E20
<John, McDonalds>	E15
<Mary, KFC>	E25

Figure B.1: Example: information structure and names

instances. Observe that the population of a fact type can thus be seen as a mapping from its predicators to a value of the population of their respective bases. Often, an ordering of the predicators is obvious from the representation of the scheme. In those cases we can denote such a mapping as a tuple.

PSM supports two kinds of subtyping mechanisms: specialization and generalization (graphically depicted as an arrow versus an arrow with a dashed line). In case of specialization, the subtypes inherrit the identification scheme from the supertype. For example, *man* and *woman* can be modeled as subtypes (specialization!) of the object type *person*. If people are identified by their name then so are men and women. Conversely, in case of generalization this identification scheme is not inherrited. For example, assume that *car* is identified by a licence plate number, and an *airplane* by some code. The object type *vehicle* can be seen as a generalization of *car* and *airplane*. Note that it is unclear how a vehicle is identified untill the actual type of vehicle is known!

Path expressions ($\mathcal{PE}$) correspond to a (directed) path through the information structure. Such path is interpreted as describing a relation between beginning and ending point. The semantics of a path expressions are defined as binary, inhomogeneous, tuple-oriented multi-relations over object types. They are built around constants, multisets, object types ($\mathcal{O}$) and predicators ($\mathcal{P}$). Let $\mu : \mathcal{PE} \to \Omega$ denote the semantics of a path expression. Before we can elaborate on μ we need to introduce the following auxiliary functions for the concatenation and reverse of multisets:

$$\begin{aligned} N \circ M &\triangleq \lambda \langle x, y \rangle \,.\, \bigcup_{a \in X} N(x,a) \times M(a,y) \\ N^{\leftarrow} &\triangleq \lambda \langle x, y \rangle \,.N(y,x) \end{aligned}$$

Using these auxiliary functions we can now introduce the semantics of path expressions in two steps: atomic path expressions and composed path expressions:

Atomic path expressions :

name	**expr.**	**semantics**
empty path	$\varnothing$	$\mu\,[\![\,\varnothing\,]\!] = \varnothing$
a constant	c	$\mu\,[\![\,c\,]\!] = \{\![c, c]\!\}$
multiset	X	$\mu\,[\![\,X\,]\!] = \{\![\langle x, x\rangle \uparrow^1 \mid x \in X]\!\}$
an object type	x	$\mu\,[\![\,x\,]\!] = \{\![\langle x, x\rangle \uparrow^1 \mid x \in \pi(x)]\!\}$
a predicator	p	$\mu\,[\![\,p\,]\!] = \{\![\langle v(p), v\rangle \uparrow^1 \mid v \in \pi \cdot \mathsf{Fact}(p)]\!\}$

composed path expressions :

name	**expr.**	**semantics**
concatenate	$P \circ Q$	$\mu\,[\![\,P \circ Q\,]\!] = \mu\,[\![\,P\,]\!] \circ \mu\,[\![\,q\,]\!]$
intersection	$P \cap Q$	$\mu\,[\![\,P \cap Q\,]\!] = \mu\,[\![\,P\,]\!] \cap \mu\,[\![\,q\,]\!]$
union	$P \cup Q$	$\mu\,[\![\,P \cup Q\,]\!] = \mu\,[\![\,P\,]\!] \cup \mu\,[\![\,q\,]\!]$
minus	$P - Q$	$\mu\,[\![\,P - Q\,]\!] = \mu\,[\![\,P\,]\!] - \mu\,[\![\,q\,]\!]$

Furthermore, several operators can be defined for path expressions such as counting, summarizing etcetera. For our purposes the *front* operator is important. For path expression P the operator $\mathcal{f}\ P$ isolates the front elements of a path.

There are many more calculations on multisets and path expressions that we ignore in this article. For our purposes the above will suffice. Recall that the path expressions enable us to reason about the population of the PSM schema. We will now introduce LISA-D with which we can add a 'syntactical sugar layer' on top of path expressions which would lead to natural, readable expressions. This is achieved by adding names to the PSM schema in the following manner:

- Let $\mathcal{N}$ be the set of all names.
- Object types are referenced by a unique name: $\mathsf{ONm} : \mathcal{O} \rightarrowtail \mathcal{N}$.
- Predicators are referenced by a unique name:$\mathsf{PNm} : \mathcal{P} \rightarrowtail \mathcal{N}$.
- Role names correspond to special connections (in the form of path expressions) through (binary) fact types: $\mathsf{RNm} : \mathcal{P} \rightarrowtail \mathcal{N}$.

The actual naming is administered by the function $\mathsf{Path} : \mathcal{O} \times \mathcal{O} \times \mathcal{N} \rightarrowtail \mathcal{PE}$ that assigns, in a given context, a path expressions to a name. For optimization purposes, beginning and endpoints of the paths are registered in the dictionary. That is, in case of $\mathsf{Path}(x, y, N) = P$: N describes a path from x to y that should be interpreted as P. Naming works as follows:

- The name $\mathsf{ONm}(x)$ of object type x stands for path expression x: $\mathsf{Path}(x, x, \mathsf{ONm}(x)) = x$

- If p a predicator then PNm describes a path from the base of p to its corresponding fact type: $\mathsf{Path}(\mathsf{Base}(p), \mathsf{Fact}(p), \mathsf{PNm}(p)) = p$
- If predicator p of a binary fact type $f = \{p, q\}$ has a role name then this role name corresponds to the path through the fact type: $\mathsf{Path}(\mathsf{Base}(p), \mathsf{Base}(q), \mathsf{RNm}(p)) = p \circ q$
- Constants do not, in essence, form paths. As such $\mathsf{Path}(*, *, c) = c$

LISA-D is built around *information descriptors* which boil down to the names of the paths as shown above. The function $\mathbb{D} : \mathcal{N} \to \mathcal{PE}$ translates information descriptors to paths. The lexicon Path contains all atomic information descriptiors:

$$\mathbb{D} [\![N]\!] = \bigcup_{\mathsf{Path}(x,y,N)!} \mathsf{Path}(x, y, N)$$

Single object types, predicator names and role names are atomic information descriptors. More fruitfull information descriptors emerge by making combinations by means of concatenation:

$$\mathbb{D} [\![P_1 P_2]\!] = \mathbb{D} [\![P_1]\!] \circ \mathbb{D} [\![P_2]\!]$$

LISA-D supports several path constructors which can be grouped into two classes: constructors that are head-oriented (i.e. that only take the heads of paths into account) and head-tail constructors. In this paper we only need the former class, most notably:

$$\begin{aligned} \mathbb{D} [\![\text{P AND-ALSO Q}]\!] &= \mathcal{f}\, \mathbb{D} [\![P]\!] \cap \mathcal{f}\, \mathbb{D} [\![Q]\!] \\ \mathbb{D} [\![\text{P OR-ELSE Q}]\!] &= \mathcal{f}\, \mathbb{D} [\![P]\!] \cup \mathcal{f}\, \mathbb{D} [\![Q]\!] \\ \mathbb{D} [\![\text{P BUT-NOT Q}]\!] &= \mathcal{f}\, \mathbb{D} [\![P]\!] - \mathcal{f}\, \mathbb{D} [\![Q]\!] \end{aligned}$$

Using the above mechanism we are able to present the details of the example presented in Figure B.1. We start by adding names to the object types and predicators in Figure B.1a which results in Figure B.1b. Part of the 'dictionary' is:

- $\mathsf{Path}(A, A, \mathsf{Person}) = A$
- $\mathsf{Path}(A, B, \mathsf{works\ for}) = p \circ q^{\leftarrow}$
- $\mathsf{Path}(B, A, \mathsf{employs}) = q \circ p^{\leftarrow}$
- $\mathsf{Path}(A, F, \mathsf{having}) = p$
- $\mathsf{Path}(F, A, \mathsf{of}) = p^{\leftarrow}$
- $\mathsf{Path}(*, *, \text{“KFC”}) = \text{“KFC”}$

Observe that Figure B.1a also presents a population for the schema, showing how People work for companies to earn their respective salaries. To see how the translation from LISA-D queries to path expressions and finally to answering the query in terms of the population works, we will work out two example queries:

The first query is to try to answer the question: which persons work for "KFC." This translates to the following path: Person works for Company with name "KFC". However, for purposes of this example we abbreviate this as follows:

$$\begin{aligned}
&\mathbb{D}\left[\!\left[\,\textsf{Person works for ``KFC''}\,\right]\!\right] = \\
&\mathbb{D}\left[\!\left[\,\textsf{Person}\,\right]\!\right] \circ \mathbb{D}\left[\!\left[\,\textsf{works for}\,\right]\!\right] \circ \mathbb{D}\left[\!\left[\,\textsf{``KFC''}\,\right]\!\right] = \\
&A \circ p \circ q^{\leftarrow} \circ \text{``KFC''}
\end{aligned}$$

We can now calculate which part of the population conforms to this path:

$$\begin{aligned}
&\mu\left[\!\left[\,A \circ p \circ q^{\leftarrow} \circ \text{``KFC''}\,\right]\!\right] = \\
&\mu\left[\!\left[\,A\,\right]\!\right] \circ \mu\left[\!\left[\,p\,\right]\!\right] \circ \mu\left[\!\left[\,q^{\leftarrow}\,\right]\!\right] \circ \mu\left[\!\left[\,\text{``KFC''}\,\right]\!\right] = \\
&\mu\left[\!\left[\,p\,\right]\!\right] \circ \mu\left[\!\left[\,q^{\leftarrow}\,\right]\!\right] \circ \mu\left[\!\left[\,\text{``KFC''}\,\right]\!\right]
\end{aligned}$$

Working out the joins leads to:

from	*to*
John	"KFC"
Mary	"KFC"

BIBLIOGRAPHY

[Ack89] R. Ackoff. From data to wisdom. *Journal of Applied Systems Analysis*, 6:3–9, 1989.

[AH04] G. Antoniou and F. van Harmelen. *A Semantic Web Primer*. The MIT Press, Cambridge, Massachusetts 02142, USA, 2004. ISBN 9780262012102

[AK00] M.J. Albers and L. Kim. Implications of the wireless web for technical communicators: User web browsing characteristics using palm handhelds for information retrieval. In *Proceedings of IEEE professional communication society international professional communication conference and Proceedings of the 18th annual ACM international conference on Computer documentation*, September 2000.

[Ald02] K. Alder. *The measure of all things: The Seven-Year Odyssey and Hidden Error That Transformed the World*. Free Press, New York, NY, USA, 2002. ISBN 074321675X

[Als99] M.V. van Alstyne. A proposal for valuing information and instrumental goods. In *Proceeding of the 20th international conference on Information Systems*, pages 328–345, Charlotte, North Carolina, USA, 1999. Association for Information Systems. ISBN ICIS1999X

[AR99] K.D. Althoff and M.M. Richter. Similarity and utility in non-numerical domains. *mathematische Methoden der Wirstchaftswissenschaften*, pages 403–413, 1999.
`http://www.iese.fraunhofer.de/pdf_files/althoff_pub/simutil1999.pdf`

[Arm95] W.Y. Arms. Key concepts in the architecture of the digital library. Technical report, Corporation for National Research Initiatives, July 1995. D-Lib Magazine.

[Bar89] J. Barwise. *The Situation in Logic.* CSLI Lecture Notes. CLSI, Stanford, California, USA, 1989.

[BBWW98] F.J.M. Bosman, P.D. Bruza, Th.P. van der Weide, and L.V.M. Weusten. Documentation, cataloging and query by navigation: A practical and sound approach. In C. Nikolaou and C. Stephanidis, editors, *Research and Advanced Technology for Digital Libraries, 2nd European Conference on Digital Libraries '98, ECDL '98*, volume 1513 of *Lecture Notes in Computer Science*, pages 459–478, Berlin, Germany, EU, September 1998. Springer.

[BCM04] G Bellinger, D. Castro, and A. Mills. Data, information, knowledge, and wisdom, 2004. Last checked: 27-Jan-2006.
`http://www.systems-thinking.org/dikw/dikw.htm`

[BDM00] P.D. Bruza, S. Dennis, and R. McArthur. Interactive internet search: keyword directory and query reformulation mechanisms compared. In *Proceedings of the 23rd Annual ACM Conference of Research and Development in Information Retrieval (SIGIR'2000)*, New York, NY, USA, 2000. ACM Press. ISBN 1581132263

[Ber98] F.C. Berger. *Navigational Query Construction in a Hypertext Environment.* PhD thesis, University of Nijmegen, The Netherlands, EU, 1998.

[Bev99] N. Bevan. Quality in use: meeting user needs for quality. *Journal of System and Software*, 49(1):89–96, 1999.

[BF92] N. Borenstein and N. Freed. Mime: Multipurpose internet mail extensions. Technical Report RFC 1341, IETF Network Working Group, June 1992. Last checked: 18-Oct-2005.
`http://www.ietf.org/rfc/rfc1341.txt`

[BG04] D Brickley and R. Guha. Rdf vocabulary description language 1.0: Rdf schema, w3c recommendation 10 february 2004. Technical report, W3C, February 2004. Last checked: 04-Okt-2005.
`http://www.w3.org/TR/rdf-schema/`

[BGP$^+$05a] P. van Bommel, B. van Gils, H.A. (Erik) Proper, E.D. Schabell, M. van Vliet, and Th.P. van der Weide. Towards an information market paradigm. In O. Belo, J. Eder, O. Pastor, and J. Falcao e Cunha, editors, *Forum proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE'2005)*, pages 27–32, Porto, Portugal, EU, June 2005. FEUP. ISBN 9727520782

[BGP$^+$05b] P. van Bommel, B. van Gils, H.A. (Erik) Proper, M. van Vliet, and Th.P. van der Weide. The information market: Its basic concepts and its challenges. In A.H.H. Ngu, M. Kitsuregawa, E.J. Neuhold, J.Y. Chung, and Q.Z. Sheng, editors, *Web Information Systems Engineering (WISE'05)*, volume 3806 of *Lecture Notes in Computer Science*, pages 577–583, Berlin, Germany, EU, November 2005. Springer-Verlag. ISBN 3540300171 `doi:10.1007/11581062_50`

[BGP$^+$05c] P. van Bommel, B. van Gils, H.A. (Erik) Proper, Th.P. van der Weide, and M. van Vliet. Value and the information market, 2005. Submitted to: Data & Knowledge Engineering.

[BH94] P.D. Bruza and T.W.C. Huibers. Investigating aboutness axioms using information fields. In W.B. Croft and C.J. van Rijsbergen, editors, *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 112–121, Berlin, Germany, EU, July 1994. Springer. ISBN 038719889X

[BH96] A.C. Bloesch and T.A. Halpin. Conquer: A conceptual query language. In B. Thalheim, editor, *Proceedings of the 15th International Conference on Conceptual Modeling (ER'96)*, volume 1157 of *Lecture Notes in Computer Science*, pages 121–133, Berlin, Germany, EU, October 1996. Springer. ISBN 3540617841

[BJ87] F.P. Brooks Jr. No silver bullet: essence and accidents of software engineering. *IEEE Computer*, 20(4):10–19, April 1987.

[BKM94] P. van Bommel, G. Kovács, and A. Micsik. Transformation of database populations and operations from the conceptual to the internal level. *Information Systems*, 19(2):175–191, 1994.

[BLHL01] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*, 284(5):34–43, May 2001.

[Bom95] P. van Bommel. *Database Optimization: An Evolutionary Approach.* PhD thesis, University of Nijmegen, The Netherlands, EU, 1995. ISBN 9090082441

[BP96] P.D. Bruza and H.A. (Erik) Proper. Discovering the information that is lost in our databases – why bother storing data if you can't find the information? Technical report, Distributed Systems Technology Centre, Brisbane, Queensland, Australia, 1996.

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[BRJ99] G. Booch, J.E. Rumbaugh, and I. Jacobson. *The Unified Modelling Language User Guide.* Addison Wesley, Reading, Massachusetts, USA, 1999. ISBN 0201571684

[Bru90] P.D. Bruza. Hyperindices: A novel aid for searching in hypermedia. In A. Rizk, N. Streitz, and J. Andre, editors, *Hypertext: Concepts, Systems and Applications; Proceedings of the European Conference on Hypertext (ECHT '90)*, number 5 in Cambridge Series on Electronic Publishing, pages 109–122, 1990. ISBN 0521405173

[Bru96] P. Brusilovsky. Methods and techniques of adaptive hypermedia. *User MOdeling and User Adapted Interaction*, 6(2–3):87–129, 1996.

[Bru01] P. Brusilovsky. Adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 11:87–110, 2001.

[Bus45] V. Bush. As we may think. *The Atlantic Monthly*, 176(1):101–108, Jul 1945.

[BW90a] K.B. Bruce and P. Wegner. An algebraic model of subtype and inheritance. In *Advances in database programming languages*, pages 75–96. ACM Press, New York, NY, USA, 1990. 0-201-50257-7
`doi:10.1145/101620.101625`

[BW90b] P.D. Bruza and Th.P. van der Weide. Two level hypermedia - an improved architecture for hypertext. In A.M. Tjoa and R.R. Wagner, editors, *Proceedings of the Data Base and Expert System Applications Conference (DEXA 90)*, pages 76–83, Berlin, Germany, EU, 1990. Springer. ISBN 3211822348

[BW92a] P. van Bommel and Th.P. van der Weide. Reducing the search space for conceptual schema transformation. *Data & Knowledge Engineering*, 8(4):269–292, September 1992.

[BW92b] P.D. Bruza and Th.P. van der Weide. Stratified hypermedia structures for information disclosure. *The Computer Journal*, 35(3):208–220, 1992.

[BYRN99] R.A Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, Reading, Massachusetts, USA, 1999. ISBN 020139829X

[CB02] T. Connolly and C. Begg. *Database Systems, a practical approach to design, implementation and management*. Addison Wesley, Reading, Massachusetts, USA, 2nd edition, 2002. ISBN 0201708574

[Che76] P.P. Chen. The entity-relationship model: Towards a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, March 1976.

[CHH$^+$01] D. Connolly, F. van Harmelen, I. Horrocks, D.L. McGuinness, and L.A. Stein. *DAML + OIL Reference Description*. W3C, December 2001. Last checked: 15-Sept-2005.
`http://www.w3.org/TR/daml+oil-reference`

[CHSS98] H. Chen, A. Houston, R. Sewell, and R. Schatz. Internet browsing and searching: User evaluations of category map and concept space techniques. *Journal of the American Society for Information Science*, 49(7):604–618, 1998.

[Cla01] R. Clarke. *Information Wants to be Free.* Xamax Consultancy Pty Ltd, 2001. Last checked: 23-Aug-2005.
`http://www.anu.edu.au/people/Roger.Clarke/II/IWtbF.html`

[Cod70] E.F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[Con87] J. Conklin. Hypertext: An introduction and survey. *IEEE Computer*, 20(9):17–41, September 1987.

[Cru00] A. Cruse. *Meaning in Language, an Introduction to Semantics and Pragmatics.* Oxford University Press, Oxford, United Kingdom, EU, 2000. ISBN 0198700105

[Dam03] oasis-open. *DARPA Agent Markup Language (DAML)*, May 2003. Last checked: 15-Sept-2005.
`http://www.oasis-open.org/cover/daml.html`

[Dat03] C.J. Date. *An introduction to Database Systems.* Addison Wesley, Reading, Massachusetts, USA, 8th edition, 2003. ISBN 0321189566

[Day00] M. Day. Resource discovery, interoperability and design preservation. some aspects of current metadata research and development. *VINE*, 117:35–48, 2000.

[DB04] P Donzelli and B. Bresciani. Improving requirements engineering by quality modelling – a quality-based requirements engineering framework. *Journal of Research and Practice in Information Technology*, 36(4), November 2004.

[Des97] B.C. Desai. Supporting discovery in virtual libraries. *Journal of the American Society for Information Science*, 48(3):190–204, 1997.

[DES05] *Quality Selection Criteria for Subject Gateways*, 2005. Last checked: 27-Oct-2005.
`http://www.sosig.ac.uk/desire/qindex.html`

[Dev90] K. Devlin. Infons and types in an information-based logic. In R. Cooper, K. Mukai, and J. Perry, editors, *Situation theory and its applications*, volume 1 of *CSLI Lecture Note Series*, pages 79–96, Stanford, California, USA, 1990. CLSI.

[Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[Diw03] U. Diwekar. *Introduction to Applied Optimization*, volume 80 of *Applied Optimization*. Springer, Berlin, Germany, EU, 2003. ISBN 1402074565

[DNR02] F.M. Donini, D. Nardi, and R. Rosati. Description logics of minimal knowledge and negation as failure. *ACM Transactions on Compututational Logic*, 3(2):177–225, 2002. ISSN: 15293785
`doi:10.1145/505372.505373`

[DO85] G.B. Davis and M.H. Olson. *Management Information Systems: Conceptual Foundations, Structure and Development*. McGraw–Hill, New York, New York, USA, 1985.

[Eus87] J. Eustace. Descartes' definition of matter. First published in The Journal of the Limerick Philosophical Society in 1987, 1987. last checked: 02-Feb-2006.
`http://www.ul.ie/~philos/vol1/eustac1.html`

[Fau00] L.C. Faulstich. *The HyperView Approach to the Integration of Semistructured Data*. PhD thesis, Institute of Computer Science, Free University Berlin, Berlin, Germany, EU, 2000.

[FHJ01] L. Feng, J.J.A.C. Hoppenbrouwers, and M.A. Jeusfeld. Towards knowledge-based digital libraries. *SIGMOD Record 30*, 1:41–46, 2001.

[Gil88] T. Gilb. *Principles of software engineering management*. Addison Wesley, Reading, Massachusetts, USA, 1988.

[GMSS04] M. Gertz, Tamer M.Ö., G. Saake, and K.U. Sattler. Report on the dagstuhl seminar: data quality on the web. *SIGMOD Rec.*, 33(1):127–132, 2004. ISSN 0163-5808

[Gog94] M. Gogolla. *An Extended Entity-Relationship Model: Fundamentals and Pragmatics*, volume 767 of *Lecture Notes in Computer Science*. Springer, Berlin, Germany, EU, 1994.

[GP99] M. Gordon and P. Pathak. Finding information on the world wide web: the retrieval effectiveness of search engines. *Information Processing and Management*, 35(2):141–180, March 1999.

[GPB04] B. van Gils, H.A. (Erik) Proper, and P. van Bommel. A conceptual model of information supply. *Data & Knowledge Engineering*, 51:189–222, 2004.

[GPBV05] B. van Gils, H.A. (Erik) Proper, P. van Bommel, and P. de Vrieze. Transformation selection for aptness–based web retrieval. In H.E. Williams and G. Dobbie, editors, *Proceedings of the Sixteenth Australasian Database Conference (ADC2005)*, volume 39, pages 115–124, Sydney, New South Wales, Australia, January 2005. Australian Computer Society. ISBN 192068221X

[GPBW04] B. van Gils, H.A. (Erik) Proper, P. van Bommel, and Th.P. van der Weide. Transformations in information supply. In J. Grundspenkis and M. Kirikova, editors, *Proceedings of the Workshop on Web Information Systems Modelling (WISM'04), held in conjunctiun with the 16th Conference on Advanced Information Systems Engineering*, volume 3, pages 60–78, June 2004. ISBN 9984976718

[GPBW05] B. van Gils, H.A. (Erik) Proper, P. van Bommel, and Th.P. van der Weide. Typing and transformational effects in complex information supply. Technical Report ICIS–R05018, Radboud University Nijmegen, Institute for Computing and Information Sciences, 2005.

[Gro00] F.A. Grootjen. Employing semantical issues in syntactical navigation. In *Proceedings of the 22nd BCS–IRSG Colloquium on IR Research*, pages 22–33, 2000.

[GTWW77] J.A. Goguen, J.W. Thatcher, E.G. Wagner, and J.B. Wright. Initial algebra semantics and continuous algebras. *Journal of the ACM (JACM)*, 24(1):68–95, 1977. ISSN 00045411

[Hal01] T.A. Halpin. *Information Modeling and Relational Databases, From Conceptual Analysis to Logical Design.* Morgan Kaufmann, San Mateo, California, USA, 2001. ISBN 1558606726

[Har96] M. Harrison. *Principles of operations management.* Pitman, London, UK, EU, 1996. ISBN 0273614509

[HC05] P. Hernon and P. Calvert. E-service quality in libraries: Exploring its features and dimensions. *Library & Information Science Research*, 27(3):377–404, 2005.

[HCD05] T. Harvey, S. Carberry, and K. Decker. Tailored responses for decision support. In L. Ardissono, P. de Bra, and A. Mitrovic, editors, *User Modeling 2005, 10th International Conference (UM 2005)*, volume 3538 of *Lecture Notes in Computer Science*, Heidelberg, Germany, EU, 2005. Springer. ISBN: 3-540-27885-0

[HE90] U. Hohenstein and G. Engels. Formal semantics of an entity-relationship-based query language. In *Proceedings of the Ninth International Conference on the Entity-Relationship Approach*, Lausanne, Switzerland, October 1990.

[HE92] U. Hohenstein and G. Engels. Sql/eer-syntax and semantics of an entity-relationship-based query language. *Information Systems*, 17(3):209–242, 1992.

[HLR96] T.W.C. Huibers, M. Lalmas, and C.J. van Rijsbergen. Information retrieval and situation theory. *ACM SIGIR Forum*, 30(1):11–25, 1996. ISSN 01635840

[HNR68] P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics SSC4*, 2:100–107, 1968.

[HO06] L. Hardman and J. van Ossenbruggen. Creating meaningful multimedia presentations. Technical Report INS-E0602, Centrum voor Wiskunde en Informatica (CWI), INformation Systems (INS), 2006.

[Hol99] M.B. Holbrook, editor. *Consumer value, a framework for analysis and research.* Routledge, New York, NY, USA, 1999. ISBN 0415191939

[Hop03] S.J.B.A. Hoppenbrouwers. *Freezing Language; Conceptualisation processes in ICT supported organisations.* PhD thesis, University of Nijmegen, Nijmegen, The Netherlands, EU, 2003. ISBN 9090173188

[HP99] S.J.B.A. Hoppenbrouwers and H.A. (Erik) Proper. Knowledge discovery - de zoektocht naar verhulde en onthulde kennis. *DB/Magazine*, 10(7):21–25, November 1999. In Dutch.

[HPW93] A.H.M. ter Hofstede, H.A. (Erik) Proper, and Th.P. van der Weide. Formal definition of a conceptual language for the description and manipulation of information models. *Information Systems*, 18(7):489–523, October 1993.

[HPW96] A.H.M. ter Hofstede, H.A. (Erik) Proper, and Th.P. van der Weide. Query formulation as an information retrieval problem. *The Computer Journal*, 39(4):255–274, September 1996.

[HW93] A.H.M. ter Hofstede and Th.P. van der Weide. Expressiveness in conceptual data modelling. *Data & Knowledge Engineering*, 10(1):65–100, February 1993.

[IEP06] Aristotle (384-322 bce): General introduction. The Internet Encyclopedia of Philosophy, 2006. last checked: 02-Feb-2006.
`http://www.utm.edu/research/iep/a/aristotl.htm`

[IJ05] P. Ingwersen and K. Järvelin. *The Turn, Integration of Information Seeking and Retrieval in context.* Springer, Dordrecht, The Netherlands, EU, 2005. ISBN: 140203850X

[ISO86] *Information Processing – Text and Office Systems – Standard General MarkUp Language (SGML)*, 1986. ISO 8879:1986, Last checked: 13-Sept-2005.
`http://www.iso.org`

[JM02] K. Januszewski and E. Mooney. *UDDI Version 3 Features List.* Oasis, 2002. Last checked: 15-Sept-2005.
`http://www.uddi.org/pubs/uddi_v3_features.htm`

[KA97] S.H. Kim and B.S. Ahn. Group decision making procedure considering preference strenght under incomplete information. *Computers & Operations Research*, 24(12):1101–1112, 1997.

[KC04] G Klyne and J.J. Carroll. Resource description framework (rdf): Concepts and abstract syntax, w3c recommendation 10 february 2004. Technical report, W3C, February 2004. Last checked: 15-Sept-2005.
`http://www.w3.org/TR/rdf-concepts/`

[KG03] D. Kulak and E. Guiney. *Use Cases: Requirements in Context.* Addison Wesley, Reading, Massachusetts, USA, 2nd edition, 2003. ASIN 0201657678

[Kob01] A. Kobsa. Generic user modeling systems. *User Modeling and User-Adapted Interaction*, 11(1–2):49–63, 2001.

[KR94] M.L. Katz and H.S. Rosen. *Microeconomics.* Irwin, 2nd edition, 1994. ISBN 0256111715

[Läm04] R. Lämmel. Transformations everywhere. *Science of Computer Programming*, 52(1–3):1–8, August 2004.

[Lan05] M.M. Lankhorst, editor. *Enterprise Architecture at Work: Modelling, Communication and Analysis.* Springer, Berlin, Germany, EU, 2005. ISBN 3540243712

[LASG02] V. Lala, A. Arnold, S.G. Sutten, and L. Guan. The impact of relative information quality of e-commerce assurance seals on internet purchasing behavior. *International Journal of Accounting Information Systems*, 3(4):237–253, December 2002.

[Lei98] B.M. Leinder. *The scope of Digital Library*, October 1998.
`http://www.dlib.org/metrics/public/papers/dig-lib-scope.html`

[LH03] S.S.g Liaw and H.M. Huang. An investigation of user attitudes toward search engines as an information retrieval tool. *Computers in Human Behavior*, 19(6):751–765, November 2003.

[LL96] K.C. Laudon and J.P. Laudon. *Management Information Systems, International Edition.* Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1996. ISBN 0132328852

[McC04] S. McConnell. *Code complete, a practical handbook of software construction.* Microsoft Press, Redmond, Washington, USA, 2 edition, 2004.

[McT93] M.F. McTear. User modelling for adaptive computer systems: a survey of recent developments. *Artificial Intelligence Review*, 7:157–184, 1993.

[Mee82] R. Meersman. The ridl conceptual language. Technical report, International Centre for Information Analysis Services, Control Data Belgium, Inc., Brussels, Belgium, EU, 1982.

[MH04] D.L. McGuinness and F. van Harmelen. Owl web ontology language: Overview. Technical report, W3C, February 2004. Last checked: 15-Sept-2005.
`http://www.w3.org/TR/owl-features/`

[MLL03] M. Montaner, B. López, and L. de La. A taxonomy of recommender agents on the internet. *Artificial Intelligence Review*, 19:285–330, 2003.

[MM97] A.O. Mendelzon and T. Milo. Formal models of web queries. In *Proceedings of 16th Symp. on Principles of Database Systems (PODS'97)*, pages 134–143, 1997. ISBN 0-89791-910-6

[MR01] F. Maurer and M. Richter. Similarity. Lecture notes for the courde K.M. in E-Commerce, chapter 6, 2001. University of Calgary, Canada.
`http://sern.ucalgary.ca/courses/SENG/609.13/S2001/slides/06.%20similarity.ppt`

[Nij89] G.M. Nijssen. *Grondslagen van Bestuurlijke Informatie Systemen.* Nijssen Adviesbureau voor Informatica B.V., 1989. In Dutch.

[OAS02] OASIS. *Standard Generalized Markup Language (SGML)*, July 2002. Last checked: 13-Sept-2005.
`http://xml.coverpages.org/sgml.html`

[OAS03] OASIS. *ebXML - Enabling a global electronic market*, 2003. Last checked: 15-Sept-2005.
`http://www.ebxml.org/geninfo.htm`

[Obj01] Object Management Group (OMG). *Common Warehouse Metamodel (CWM) metamodel*, version 1.0 edition, Februari 2001.
`http://www.omg.org/technology/cwm/`

[Oos03] H. van Oostendorp, editor. *Cognition in a digital world.* Lawrence Erlbaum Associates, Mahway, New Jersey, USA, 2003. ISBN 0805835075

[Orr98] K. Orr. Data quality and systems theory. *Communications of the ACM*, 41(2):66–71, 1998. ISSN 00010782

[Pai99] J.J. Paijmans. *Explorations in the Document Vector Model of Information Retrieval.* PhD thesis, Tilburg University, The Netherlands, EU, 1999. ISBN 9036100240

[PBMW98] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[PGPR02] M. Polo, J.A. Gomez, M. Piattini, and F. Ruiz. Generating three-tier applications from relational databases: a formal and practical approach. *Information and Software Technology*, 44(15):923–941, December 2002.

[Pij94] G.J. van der Pijl. Quality of information and the goals and targets of the organization. In *SIGCPR '94: Proceedings of the 1994 computer personnel research conference on Reinventing IS : managing information technology in changing organizations*, pages 165–172, New York, NY, USA, 1994. ACM Press. ISBN 0897916522

[PJ98] T.B. Pedersen and C.S. Jensen. Multidimensional data modeling for complex data. In *Proceedings of the 15th International Conference on Data Engineering*, pages 336–345. IEEE Computer Society, 1998. ISBN 0769500714

[PPY01] M.P. Papazoglou, H.A. (Erik) Proper, and J. Yang. Landscaping the information space of large multi-database networks. *Data & Knowledge Engineering*, 36(3):251–281, 2001.

[PRBV90] W.J. Premerlani, J.E. Rumbaugh, Michael R. Blaha, and Thomas A. Varwig. An object-oriented relational database. *Communications of the ACM*, 33(11):99–109, 1990.

[Pro94] H.A. (Erik) Proper. *A Theory for Conceptual Modelling of Evolving Application Domains.* PhD thesis, University of Nijmegen, Nijmegen, The Netherlands, EU, 1994. ISBN 909006849X

[Pro97] H.A. (Erik) Proper. Data schema design as a schema evolution process. *Data & Knowledge Engineering*, 22(2):159–189, 1997.

[PW95] H.A. (Erik) Proper and Th.P. van der Weide. Information disclosure in evolving information systems: Taking a shot at a moving target. *Data & Knowledge Engineering*, 15:135–168, 1995.

[RCDT01] J.W. Rahayu, E. Chang, T.S. Dillon, and D. Taniar. Performance evaluation of the object-relational transformation methodology. *Data & Knowledge Engineering*, 38(3):265–300, September 2001.

[Rij75] C.J. van Rijsbergen. *Information Retrieval.* Butterworths, London, United Kingdom, EU, 1975. ISBN 0408709294

[Rij04] D. Rijsenbrij. *Architectuur in de digitale wereld (versie nulpuntdrie).* Nijmegen Institute for Information and Computing Sciences, Radboud University Nijmegen, Nijmegen, The Netherlands, EU, October 2004. In Dutch. ISBN 90901882853

[RL96] C.J. van Rijsbergen and M. Lalmas. Information calculus for information retrieval. *Journal of the American Society for Information Science*, 47(5):385–398, May 1996.

[ROH04] L. Rutledge, J. van Ossenbruggen, and L. Hardman. Structuring and presenting annotated media repositories. In S.I. Feldman, M. Uretsky, M. Najork, and C.E. Wills, editors, *WWW (Alternate Track Papers & Posters)*, pages 466–467. ACM, 2004. ISBN: 1-58113-912-8

[ROH05] L. Rutledge, J. van Ossenbruggen, and L. Hardman. Making rdf presentable: integrated global and local semantic web browsing. In A. Ellis and T. Hagino, editors, *Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba*, pages 199–206. ACM, 2005. ISBN: 1-59593-046-9
`http://doi.acm.org/10.1145/1060745.1060777`

[SBS98] M. Sarkar, B. Butler, and C. Steinfield. Cybermediaries in electronic marketspace: Toward theory building. *Journal of Business Research*, 41(3):215–221, March 1998.

[SC96] B.R. Schatz and H. Chen. Interactive term suggestion for users of digital libraries. In *First ACM International Conference on Digital Libraries*, pages 126–133, March 1996.

[SH05] K. van der Sluijs and G Houben. Towards a generic user model component. In *PerSWeb05 Workshop on Personalization on the Semantic Web*, 2005.

[SM83] G.E Salton and M.J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, NY, USA, 1983.

[Som89] I. Sommerville. *Software Engineering.* Addison Wesley, Reading, Massachusetts, USA, 1989.

[SS04] C.H. Scott and J.E. Scott. On models for the operation of a class of electronic marketplaces. *Omega*, 32(5):373–383, October 2004.

[Ste02] The stencil group. *The Evolution of UDDI*, July 2002. Last checked: 15-Sept-2005.
`http://www.uddi.org/pubs/the_evolution_of_uddi_20020719.pdf`

[Sug03] R. Sugden. Reference-dependent subjective exptected utility. *Journal of economic theory*, 11(2):172–191, 2003.

[SV99] C.E. Shannon and H.R. Varian. *Information Rules, a strategic guide to the network economy.* Harvard Business School Press, Boston, Massachusetts, USA, 1999. ISBN 097584863X

[SWC$^+$95] N.A. Stillings, S.E. Weisler, C.H. Chase, M.H. Feinstein, J.L. Garfield, and E.L. Rissland. *Cognitive Science, an introduction.* Massachusetts Institute of Technology, second edition, 1995. ISBN 0262193531

[SWL$^+$02] J.W. Song, K.Y. Whang, Y.K. Lee, M.J. Lee, Han W.S., and B.K. Park. The clustering property of corner transformation for spatial database applications. *Information and Software Technology*, 44(7):419–429, May 2002.

[SYB98] M. Sahami, S. Yusufali, and M.Q. Baldonado. Sonia: a service for organizing networked information autonomously. In I. Witten, R. Akscyn, and F.M. Shipman, editors, *Proceedings of DL-98, 3rd ACM Conference on Digital Libraries*, pages 200–209, New York, NY, USA, 1998. ACM Press. ISBN 0897919653

[SZ87] K.E. Smith and S.B. Zdonik. Intermedia: A case study of the differences between relational and object-oriented database systems. In *Conference proceedings on Object-oriented programming systems, languages and applications*, pages 452–465, New York, NY, USA, 1987. ACM Press. ISBN 0362-1340

[Tah92] H.A. Taha. *Operations Research, an introduction.* Prentice–Hall, Englewood Cliffs, New Jersey, USA, 4 edition, 1992. ISBN 0131876597

[TG03] H. Tardieu and V. Gyselinck. Working memory constraints in the integration and comprehension of information in a multimedia context. In H. van Oostendorp, editor, *Cognition in a digital world*, chapter 1, pages 3–24. Lawrence Erlbaum Associates, Hillsdale, New Jersey, USA, 2003. ISBN 0805835075

[TLKC99] E. Turban, J. Lee, D. King, and H.M. Chung. *Electronic Commerce, a managerial perspective.* Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1999. ISBN 0139752854

[Ull89] J.D. Ullman. *Principles of Database and Knowledge–base Systems*, volume I. Computer Science Press, Rockville, Maryland, USA, 1989. ISBN 0716781581

[Var95] H.R. Varian. Pricing information goods. In Research Libraries Group, editor, *Scholarship of the New Information Environment Proceedings*, 1995.

[Var96] H.R. Varian. *Intermediate Microeconomics, a modern approach.* Norton, New York, NY, USA, 4th edition, 1996. ISBN 0393968421

[Vit99] M. Vitruvius. *Handboek Bouwkunde.* Athenaeum – Polak & Van Gennep, Amsterdam, The Netherlands, EU, 1999. Translated by: T. Peters. ISBN 9025358705

[VW99] C. Vishik and A.B. Whinston. Knowledge sharing, quality, and intermediation. In *Proceedings of the international joint conference on Work activities coordination and collaboration*, pages 157–166, New York, NY, USA, 1999. ACM Press. ISBN 1581130708

[Wat99] R.T. Watson. *Data Management, Databases and Organizations.* John Wiley & Sons, Inc., New York, NY, USA, second edition edition, 1999. ISBN 0471347116

[Wik06] Wikipedia. A* search algorithm, 2006. Last checked: 16-Feb-2006.
`http://en.wikipedia.org/wiki/A%2A_search_algorithm`

[Wil82] R. Wille. Restructuring lattice theory: An approach based on hierarchies of concepts. In I. Rival, editor, *Ordered Sets*, pages 445–470. D. Reidel Publishing Company, The Netherlands, EU, 1982.

[WKLW98] S. Weibel, J. Kunze, C. Lagoze, and M. Wolf. Dublin core metadata for resource discovery. Technical report, Internet Engineering Task Force (IETF), 1998. Last checked: 13-Sept-2005.
`http://www.ietf.org/rfc/rfc2413.txt`

[Zad75a] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning – i. *Information Science*, 8:199–249, 1975.

[Zad75b] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning – ii. *Information Science*, 8:301–357, 1975.

[Zad75c] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning – iii. *Information Science*, 9:301–357, 1975.

[Zad02] L.A. Zadeh. From computing with numbers to computing with words – from manipulation of measurements to manipulation of perceptions. *International Journal of Applied Mathematics and Computer Science*, 12:307–324, 2002.

[ZCC95] M.R. Zand, V. Collins, and D. Caviness. A survey of current object-oriented databases. *ACM SIGMIS Database*, 26(1):14–29, 1995.

[ZG00] X. Zhu and S. Gauch. Incorporating quality metrics in centralized/distributed information retrieval on the world wide web. In *Proceedings of the 23rd annual international ACM SIGIR conference on Researchand development in information retrieval*, pages 288–295, July 2000. ISBN 1581132263

Author index

INDEX

SIKS dissertatiereeks

1998-01 Johan van den Akker (CWI) *DEGAS - An Active, Temporal Database of Autonomous Objects*

1998-02 Floris Wiesman (UM) *Information Retrieval by Graphically Browsing Meta-Information*

1998-03 Ans Steuten (TUD) *A Contribution to the Linguistic Analysis of Business Conversations within the Language/Action Perspective*

1998-04 Dennis Breuker (UM) *Memory versus Search in Games*

1998-05 E.W. Oskamp (RUL) *Computerondersteuning bij Straftoemeting*

1999-01 Mark Sloof (VU) *Physiology of Quality Change Modelling; Automated modelling of Quality Change of Agricultural Products*

1999-02 Rob Potharst (EUR) *Classification using decision trees and neural nets*

1999-03 Don Beal (UM) *The Nature of Minimax Search*

1999-04 Jacques Penders (UM) *The practical Art of Moving Physical Objects*

1999-05 Aldo de Moor (KUB) *Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems*

1999-06 Niek J.E. Wijngaards (VU) *Redesign of compositional systems*

1999-07 David Spelt (UT) *Verification support for object database design*

1999-08 Jacques H.J. Lenting (UM) *Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism for Discrete Reallocation*

2000-01 Frank Niessink (VU) *Perspectives on Improving Software Maintenance*

2000-02 Koen Holtman (TUE) *Prototyping of CMS Storage Management*

2000-03 Carolien M.T. Metselaar (UVA) *Sociaal-organisatorische gevolgen van kennistechnologie; een procesbenadering en actorperspectief*

2000-04 Geert de Haan (VU) *ETAG, A Formal Model of Competence Knowledge for User Interface Design*

2000-05 Ruud van der Pol (UM) *Knowledge-based Query Formulation in Information Retrieval*

2000-06 Rogier van Eijk (UU) *Programming Languages for Agent Communication*

2000-07 Niels Peek (UU) *Decision-theoretic Planning of Clinical Patient Management*

2000-08 Veerle Coup (EUR) *Sensitivity Analyis of Decision-Theoretic Networks*

2000-09 Florian Waas (CWI) *Principles of Probabilistic Query Optimization*

2000-10 Niels Nes (CWI) *Image Database Management System Design Considerations, Algorithms and Architecture*

2000-11 Jonas Karlsson (CWI) *Scalable Distributed Data Structures for Database Management*

2001-01 Silja Renooij (UU) *Qualitative Approaches to Quantifying Probabilistic Networks*

2001-02 Koen Hindriks (UU) *Agent Programming Languages: Programming with Mental Models*

2001-03 Maarten van Someren (UvA) *Learning as problem solving*

2001-04 Evgueni Smirnov (UM) *Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets*

2001-05 Jacco van Ossenbruggen (VU) *Processing Structured Hypermedia: A Matter of Style*

2001-06 Martijn van Welie (VU) *Task-based User Interface Design*

2001-07 Bastiaan Schonhage (VU) *Diva: Architectural Perspectives on Information Visualization*

2001-08 Pascal van Eck (VU) *A Compositional Semantic Structure for Multi-Agent Systems Dynamics*

2001-09 Pieter Jan 't Hoen (RUL) *Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes*

2001-10 Maarten Sierhuis (UvA) *Modeling and Simulating Work Practice BRAHMS: a multiagent modeling and simulation language for work practice analysis and design*

2001-11 Tom M. van Engers (VUA) *Knowledge Management: The Role of Mental Models in Business Systems Design*

2002-01 Nico Lassing (VU) *Architecture-Level Modifiability Analysis*

2002-02 Roelof van Zwol (UT) *Modelling and searching web-based document collections*

2002-03 Henk Ernst Blok (UT) *Database Optimization Aspects for Information Retrieval*

2002-04 Juan Roberto Castelo Valdueza (UU) *The Discrete Acyclic Digraph Markov Model in Data Mining*

2002-05 Radu Serban (VU) *The Private Cyberspace Modeling Electronic Environments inhabited by Privacy-concerned Agents*

2002-06 Laurens Mommers (UL) *Applied legal epistemology; Building a* *knowledge-based ontology of the legal domain*

2002-07 Peter Boncz (CWI) *Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications*

2002-08 Jaap Gordijn (VU) *Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas*

2002-09 Willem-Jan van den Heuvel(KUB) *Integrating Modern Business Applications with Objectified Legacy Systems*

2002-10 Brian Sheppard (UM) *Towards Perfect Play of Scrabble*

2002-11 Wouter C.A. Wijngaards (VU) *Agent Based Modelling of Dynamics: Biological and Organisational Applications*

2002-12 Albrecht Schmidt (Uva) *Processing XML in Database Systems*

2002-13 Hongjing Wu (TUE) *A Reference Architecture for Adaptive Hypermedia Applications*

2002-14 Wieke de Vries (UU) *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*

2002-15 Rik Eshuis (UT) *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*

2002-16 Pieter van Langen (VU) *The Anatomy of Design: Foundations, Models and Applications*

2002-17 Stefan Manegold (UVA) *Understanding, Modeling, and Improving Main-Memory Database Performance*

2003-01 Heiner Stuckenschmidt (VU) *Ontology-Based Information Sharing in Weakly Structured Environments*

2003-02 Jan Broersen (VU) *Modal Action Logics for Reasoning About Reactive Systems*

2003-03 Martijn Schuemie (TUD) *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*

2003-04 Milan Petkovic (UT) *Content-Based Video Retrieval Supported by Database Technology*

2003-05 Jos Lehmann (UVA) *Causation in Artificial Intelligence and Law - A modelling approach*

2003-06 Boris van Schooten (UT) *Development and specification of virtual environments*

2003-07 Machiel Jansen (UvA) *Formal Explorations of Knowledge Intensive Tasks*

2003-08 Yongping Ran (UM) *Repair Based Scheduling*

2003-09 Rens Kortmann (UM) *The resolution of visually guided behaviour*

2003-10 Andreas Lincke (UvT) *Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture*

2003-11 Simon Keizer (UT) *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*

2003-12 Roeland Ordelman (UT) *Dutch speech recognition in multimedia information retrieval*

2003-13 Jeroen Donkers (UM) *Nosce Hostem - Searching with Opponent Models*

2003-14 Stijn Hoppenbrouwers (KUN) *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*

2003-15 Mathijs de Weerdt (TUD) *Plan Merging in Multi-Agent Systems*

2003-16 Menzo Windhouwer (CWI) *Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses*

2003-17 David Jansen (UT) *Extensions of Statecharts with Probability, Time, and Stochastic Timing*

2003-18 Levente Kocsis (UM) *Learning Search Decisions*

2004-01 Virginia Dignum (UU) *A Model for Organizational Interaction: Based on Agents, Founded in Logic*

2004-02 Lai Xu (UvT) *onitoring Multi-party Contracts for E-business*

2004-03 Perry Groot (VU) *A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving*

2004-04 Chris van Aart (UVA) *Organizational Principles for Multi-Agent Architectures*

2004-05 Viara Popova (EUR) *Knowledge discovery and monotonicity*

2004-06 Bart-Jan Hommes (TUD) *The Evaluation of Business Process Modeling Techniques*

2004-07 Elise Boltjes (UM) *Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes*

2004-08 Joop Verbeek (UM) *Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politile gegevensuitwisseling en digitale expertise*

2004-09 Martin Caminada (VU) *For the Sake of the Argument; explorations into argument-based reasoning*

2004-10 Suzanne Kabel (UVA) *Knowledge-rich indexing of learning-objects*

2004-11 Michel Klein (VU) *Change Management for Distributed Ontologies*

2004-12 The Duy Bui (UT) *Creating emotions and facial expressions for embodied agents*

2004-13 Wojciech Jamroga (UT) *Using Multiple Models of Reality: On Agents who Know how to Play*

2004-14 Paul Harrenstein (UU) *Logic in Conflict. Logical Explorations in Strategic Equilibrium*

2004-15 Arno Knobbe (UU) *Multi-Relational Data Mining*

2004-16 Federico Divina (VU) *Hybrid Genetic Relational Search for Inductive Learning*

2004-17 Mark Winands (UM) *Informed Search in Complex Games*

2004-18 Vania Bessa Machado (UvA) *Supporting the Construction of Qualitative Knowledge Models*

2004-19 Thijs Westerveld (UT) *Using generative probabilistic models for multimedia retrieval*

2004-20 Madelon Evers (Nyenrode) *Learning from Design: facilitating multidisciplinary design teams*

2005-01 Floor Verdenius (UVA) *Methodological Aspects of Designing Induction-Based Applications*

2005-02 Erik van der Werf (UM) *AI techniques for the game of Go*

2005-03 Franc Grootjen (RUN) *A Pragmatic Approach to the Conceptualisation of Language*

2005-04 Nirvana Meratnia (UT) *Towards Database Support for Moving Object data*

2005-05 Gabriel Infante-Lopez (UVA) *Two-Level Probabilistic Grammars for Natural Language Parsing*

2005-06 Pieter Spronck (UM) *Adaptive Game AI*

2005-07 Flavius Frasincar (TUE) *Hypermedia Presentation Generation for Semantic Web Information Systems*

2005-08 Richard Vdovjak (TUE) *A Model-driven Approach for Building Distributed Ontology-based Web Applications*

2005-09 Jeen Broekstra (VU) *Storage, Querying and Inferencing for Semantic Web Languages*

2005-10 Anders Bouwer (UVA) *Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments*

2005-11 Elth Ogston (VU) *Agent Based Matchmaking and Clustering - A Decentralized Approach to Search*

2005-12 Csaba Boer (EUR) *Distributed Simulation in Industry*

2005-13 Fred Hamburg (UL) *Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen*

2005-14 Borys Omelayenko (VU) *Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics*

2005-15 Tibor Bosse (VU) *Analysis of the Dynamics of Cognitive Processes*

2005-16 Joris Graaumans (UU) *Usability of XML Query Languages*

2005-17 Boris Shishkov (TUD) *Software Specification Based on Re-usable Business Components*

2005-18 Danielle Sent (UU) *Test-selection strategies for probabilistic networks*

2005-19 Michel van Dartel (UM) *Situated Representation*

2005-20 Cristina Coteanu (UL) *Cyber Consumer Law, State of the Art and Perspectives*

2005-21 Wijnand Derks (UT) *Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics*

2006-01 Samuil Angelov (TUE) *Foundations of B2B Electronic Contracting*

2006-02 Cristina Chisalita (VU) *Contextual issues in the design and use of information technology in organizations*

2006-03 Noor Christoph (UVA) *The role of metacognitive skills in learning to solve problems*

2006-04 Marta Sabou (VU) *Building Web Service Ontologies*

2006-05 Cees Pierik (UU) *Validation Techniques for Object-Oriented Proof Outlines*

2006-06 Ziv Baida (VU) *Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling*

2006-07 Marko Smiljanic (UT) *XML schema matching - balancing efficiency and effectiveness by means of clustering*

2006-08 Eelco Herder (UT) *Forward, Back and Home Again - Analyzing User Behavior on the Web*

2006-09 Mohamed Wahdan (UM) *Automatic Formulation of the Auditor's Opinion*

2006-10 Ronny Siebes (VU) *Semantic Routing in Peer-to-Peer Systems*

2006-11 Joeri van Ruth (UT) *Flattening Queries over Nested Data Types*

2006-12 Bert Bongers (VU) *Interactivation - Towards an e-cology of people, our technological environment, and the arts*

2006-13 Henk-Jan Lebbink (UU) Dialogue and Decision Games for Information Exchanging Agents

2006-14 Johan Hoorn (VU) *Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change*

2006-15 Rainer Malik (UU) *CONAN: Text Mining in the Biomedical Domain*

2006-16 Carsten Riggelsen (UU) *Approximation Methods for Efficient Learning of Bayesian Networks*

2006-17 Stacey Nagata (UU) *User Assistance for Multitasking with Interruptions on a Mobile Device*

2006-18 Valentin Zhizhkun (UVA) *Graph transformation for Natural Language Processing*

2006-19 Birna van Riemsdijk (UU) *Cognitive Agent Programming: A Semantic Approach*

2006-20 Marina Velikova (UvT) *Monotone models for prediction in data mining*

2006-21 Bas van Gils (RUN) *Aptness on the Web*

Samenvatting

De hoeveelheid *data* die aangeboden wordt op het Web groeit. De tijd dat het Web nog in de kinderschoenen stond lijkt reeds lang voorbij, hoewel het in kalenderjaren gemeten nog niet zo erg lang geleden is. Deze rappe ontwikkeling en groei van het Web heeft er onder andere toe geleid dat we in toenemende mate afhankelijk zijn van het Web voor onze informatievoorziening. Dit legt nogal wat druk op zoeksystemen die ons helpen bij het vinden van *informatie* in de enorme hoeveelheid aangeboden data. Zoeksystemen zijn het onderwerp van studie in dit proefschrift.

De taak van zoeksystemen is om te bepalen (uit te rekenen) welke resources voldoen aan de zoekopdracht van een specifieke zoeker. Hierbij zijn de volgende aspecten van belang: allereerst gaan we er vanuit dat de zoekopdracht (ook wel de *query* genoemd) een goede indicatie geeft van het soort probleem dat de zoeker heeft. Impliciet nemen we dus aan dat de zoeker in staat is om goede queries te formuleren in de taal die de zoekmachine begrijpt. Daarnaast gebruiken we de generieke term *resources* in plaats van documenten. Dit is een bewust keuze omdat we niet uit willen sluiten dat het antwoord op een zoekvraag niet uit een statisch document, maar wel uit een online database, een E-service of een online conversatie naar voren komt.

Terug komend op de taak van het zoeksysteem: op hoog niveau van abstractie kan gesteld worden dat zoeksystemen resources waarderen aan de hand van de (informatie)behoefte van een individuele zoeker. *Waardering* is een begrip dat uitgebreid is bestudeerd in de economische literatuur. Het ligt dan ook voor de hand om resultaten uit dit onderzoeksveld te herbruiken als het gaat om het zoeken naar informatie op heb Web. Het belangrijkste resultaat van deze exercitie is de observatie dat waarde (van een asset) voor een actor persoonlijk is en derhalve uitsluitend uit te drukken valt in termen van de waardering van andere assets. Met andere woorden: waardering resulteert in een partiële ording van assets voor een actor. Bovendien stellen we dat er drie grondredenen kunnen zijn voor de mate waarin een asset gewaardeerd wordt door een actor: (1) informatie: waar gaat een resource over, en hoe “goed” is dat voor een actor met een specifieke informatiebehoefte (2) structuur: wat voor structurele eigenschappen heeft een resource (denk aan: type, resolutie, prijs, etcetera) en hoe passen deze eigenschappen bij de eisen van de zoeker en (3) emotie: hoe goed past een resource bij de (huidige gemoeds)toestand van een zoeker.

Informationele waarde is het onderwerp van studie in o.a. de traditionele information retrieval literatuur. Het betreft in weze het uitrekenen/ schatten of een resource over het juiste onderwerp gaat. In dit proefschrift hebben we een generieke theorie voor informationelle waarde gepresenteer welke in de literatuur een *infon theorie* wordt genoemd. Het centrale concept in deze theorie is de infon, een informatie-atoom. Door atomen te combineren kunnen complerere informatiestructuren gecreerd worden. Deze structuren kunnen gebruikt worden om aan te geven waar een resource over gaat, dan wel om aan te geven welke informatiebehoefte een zoeker heeft.

Het rekenen met emoties is een stuk lastiger, hoewel literatuur wel enkele aanknopingspunten geeft over hoe emoties van mensen in elkaar zitten. De emotionele aspecten van waardering moeten in de breedste zin van het woord "emotie" gezien worden. Het betreft hier onder andere zaken als gemoedstoestand, de mate van bereidheid om ingewikkelde documenten te bestuderen etcetera. In de context van dit proefschrift zijn de volgende vragen relevant: hoe modelleren we emoties? Hoe representeren we emoties? Hoe voorspellen we welk effect het consumeren van een resource heeft op de emotionele toestand van een zoeker? We zijn niet heel diep op deze aspecten van waardering in gegaan. We verwachten dat de *user modeling* gemeenschap hier in de nabije toekomst aanknopingspunten zal leveren.

Rest nog de structurele aspecten van waardering. Hierbij gaat het om (het meten van) structurele eigenschappen van resources. In dit proefschrift hebben we een (meta) taal gepresenteerd waarmee het mogelijk is om eigenschappen van resources te specificeren. Voorbeelden van eigenschappen zijn bijvoorbeeld document grootte, de taal waarin een resource gesteld is, de prijs ervan, etcetera. Tevens hebben we een mechanisme ontwikkeld waarmee het meten van deze eigenschappen formeel wordt onderbouwd.

Het meten van eigenschappen van objecten is minder triviaal dan het op het eerste gezicht lijkt. Een van de redenen is dat metingen af kunnen hangen van omstandigheden waarin de metingen gedaan worden. Bovendien kan gesteld worden dat metingen subjectief zijn tenzij het "tellen" betreft. Onze theorie voor metingen is gebaseerd op resultaten uit de *fuzzy logic* en vormt de basis voor het waarderen van resources voor zoekers. Bij deze waarderingen nemen we zo veel mogelijk eigenschappen / wensen van zoekers en resources in beschouwing. Derhalve kan gesteld worden dat de waarderingsmechaniek rijk is; rijker dan uitsluitend op basis van keywords vaststellen of een document al dan niet geschikt is voor een zoeker. Deze waarderingsmechaniek vormt de kern van *aptness based retrieval* waarin de geschiktheid van resources centraal staat.

Curriculum Vitae

Bas was born in Tilburg on December 6, 1976. His secondary education took place in Tilburg at the *Cobbenhagencollege* where he did both HAVO and VWO. After graduation he started studying econometrics at the university of Tilburg. After one year he switched to information management and technology (bestuurlijke informatiekunde) at the same university. He graduated on the topic of "application of semantic matching for enterprise application integration".

During his studies Bas got interested in the fields of *information retrieval* and *conceptual modeling*. After receiving his Masters degree at the university of Tilburg he started as a junior researcher at the university of Nijmegen. He worked mostly on the PRONIR (Profile-based Retrieval of Networked Information Resources) project which had a research theme titled *universal access of networked information resources*. While working on his PhD thesis Bas also taught several sources. He is married and has one child.