Chapter 9
Ontology-based Knowledge Elicitation: An Architecture

Marcello Montedoro, Giorgio Orsi, Licia Sbattella, and Roberto Tedesco

Abstract This chapter overviews the process of collection and automatic analysis of data and documents both inside and outside the Networked Enterprise. We will address the following research problems: discovery of the useful information sources, in terms of the enterprise documentation, of structured and unstructured data provided by existing information systems, of web-available knowledge, of event flow within the business processes; extraction of synthetic knowledge from these information sources, possibly in terms of a common, semantic data model; automatic interpretation and integration of the acquired information; analysis and dissemination of such knowledge to all decisional levels, appropriately adapting it to the user’s function and context.

9.1 Introduction

In a networked enterprise, a large amount of information is exchanged among the players. Thus, methodologies and tools are needed which can identify, elicit, and represent knowledge (often tacit) where it manifests its relevance for the network partners. Since collective knowledge evolves dynamically, sometimes it may manifest itself upstream in the supply (or
design) chain, sometimes downstream (e.g., at the consumers level). The ArtDeco project focuses on the problems of finding, extracting, representing and formalising knowledge, in all various forms in which it may be embedded (in particular, natural language) in order to build a semantic model (via ontologies) of the business domains of networked enterprises. In this scenario we have implemented the ArtDeco Web-Portal, which aims at providing access to an advanced document and data repository, shared by the network partners.

9.2 System Architecture

Figure 9.1 depicts the main architecture of the ArtDeco Web Portal, which provides five main functionalities: taxonomy-driven word tagging and tag-based querying (the OmniFind module); knowledge extraction from natural language sources and concept-based natural language querying (the Knowledge Indexing & Extraction module); data and knowledge extraction from sensor networks and (semi-)structured data sources (the ArtDeco Dynamic Data Integration Systems - AD-DDIS module); and capture & analysis of data collected from enterprise processes (the PROfessional Metrics - PROM module).

Such modules rely on information stored in the Domain Model. The Domain Model is composed of two sub models: the first one describes the
domain by means of a Taxonomy, while the second one relies on a Domain Ontology, a Semantic Network, and a Mapping Model.

The Taxonomy, compiled by a domain expert, provides a hierarchy of tags that associate labels to the words that lexicalise them. This simple model is then used by the OmniFind module to provide taxonomy-driven word tagging and support tag-based queries (see Section 9.3).

The Domain Ontology, a knowledge base defined by means of Description Logics [3], contains the concepts relevant for the application domain. As, in our opinion, the model should represent the domain at several levels of abstraction, we made a clear distinction between conceptual level and lexical level. The Domain Ontology represents the conceptual level model, which contains all the knowledge about the domain we need to represent. This level is language independent, as the ontology contains the definition of concepts [11], without considering any possible linguistic representation. The Domain Ontology is defined by a domain expert, starting from, and extending, the Taxonomy.

The lexical level provides the vocabulary, i.e. terms that the system is able to recognise. In particular, we adopted a Semantic Network [17] as a lexical-level model. The semantic networks connect words using specific linguistic relationships, such as synonymy, antinomy, hyponymy, etc. These relationships enable vocabulary navigation in order to discover word similarities and meanings.

Domain Ontology and Semantic Network are connected by means of a stochastic Mapping Model, which permits to lexicalise the concepts—it enables the translation of abstract definitions to concrete words.

Domain Ontology, Semantic Network, and Mapping Model are used by the Knowledge Indexing & Extraction module to provide knowledge extraction from natural language sources and support concept-based natural language queries (see Section 9.4).

The Extractors gather data from heterogeneous sources, such as textual documents, web pages, applications, sensor networks, database, XML files, and processes.

The OmniFind and the Knowledge Indexing & Extraction modules analyse and index the content of textual documents. Data coming from application and sensor networks are given to the Internal Enterprise Data module (see Section 9.5), while data captured from processes enter the PROM module (see Section 9.6); finally, ontologies extracted from structured repositories, such as relational databases and XML files, are used by the AD-DDIS module to allow on-the-fly access from external applications (see Section 9.5).

Users interact with the system by means of the Context-aware Web Portal. The portal provides different user profiles with customised views on the data, thanks to the Context-based Information Filtering facility. Within ArtDeco, context-awareness is achieved through the Context-Based Information Filters that correspond to context-aware user views over the database or the data warehouse. Each view corresponds to a different working context of
the ArtDeco Web Portal and it is determined on the basis of a context-model and a methodology for Context-Aware View Design (see Chapter 14).

User queries coming from the Web Portal will be answered by means of the views associated to the current context instead of resorting to the whole database or data warehouse schemata that may contain unnecessary information.

Enterprise Applications can exchange information with the Internal Enterprise Data database, in order to take advantage of the data collected by the system, or interact with the portal to gain information from the other data sources, possibly filtered on the basis of the context.

In the following the aforementioned modules will be briefly presented.

9.3 The OmniFind module

OmniFind permits to associate words with tags defined in the Taxonomy. The tag-based query mechanism is then provided to the user, as a set of pre-defined forms. Each form is designed to meet a specific user requirement; forms are associated to user profiles, thus each profile has its own view on the data.

Figure 9.2 shows the architecture of OmniFind, based on functionalities provided by IBM OmniFind Enterprise Edition. In the following, the components of OmniFind are introduced.

The Document Crawler performs the function of crawling the various data sources (usually, web sites) at intervals configured by the administrator, and populates a raw data store (file system based) with the contents extracted from the data sources. Users can define one or more collections where crawlers can store documents.

The Knowledge Extractor analyses documents collected by the crawler, parses and indexes them. The parser analyses documents’ content and metadata; users can improve the quality and precision of the parsing phase by integrating custom text processing algorithms in the parser. These text processing algorithms are developed using the Unstructured Information Management Architecture (UIMA)\(^1\), which is a framework for creating, discovering, composing, and deploying text analysis functions. The parser performs the following tasks on each document: extracts the text; detects the source language; applies parsing rules specified for the collection, for example it is possible to define categories so that users can search documents by means of the categories they belong to. Moreover, the parser associates texts’ words with related labels defined into the Taxonomy. Finally, the parser stores the results of the analysis in a file temporary system data store.

\(^1\) See: incubator.apache.org/uima
The indexer may be scheduled at regular intervals in order to periodically read from the temporary store, adding information about new and changed documents to an index that is stored in a file system. In practice, the index is split into (i) a *main index*, which has the global ranking order of all the documents and (ii) the *Delta index*, which corresponds to those documents that have not yet been merged into the main index.

The Search Component relies on OmniFind Search Servers to provide search functionalities to the user. The search application receives input from a browser and invokes the search servers through the Search and Index API (SIAPI). Search Servers execute the query against the index and returns a list of results to the search application; each result includes a “quick link”, which displays a preview of the related document.

The Web Application represents the external interface through which users interact with OmniFind. This component is also used to create and administer collections, start and stop other components (such as the crawler, parser, indexer, and search), monitor system activity and log files, configure administrative users, and associate search applications with collections. Moreover, Web Application has been customised to provide the query forms. Such queries depend on the user profile, provided by the ArtDeco Context-aware Web Portal.

---

**Fig. 9.2 The IBM OmniFind Enterprise Edition High Level Architecture**
9.4 The Knowledge Indexing & Extraction module: Concepts from texts

Figure 9.3 depicts the functionalities provided by the Knowledge Indexing & Extraction module, which –after the training phase– is able to extract and index concepts from documents written in natural language (e.g., PDF or Word files, web pages, etc.), execute natural language based queries, extend the ontology, and export concepts to the Internal Enterprise Data module for further elaboration.

For example, an Oenologist designing a new red wine could be interested in gathering information from specialized reviews available on Internet. Using the system Web Portal, she/he inserts the URLs of such web sites; then, the Text Extractor downloads the web pages and extracts the text, while the Knowledge Indexing & Extraction module discovers and indexes the relevant concepts. The advanced query typologies supported by the system provide a powerful tool for both documental and conceptual search, where a large variety of different requests can be expressed.

In the following such functionalities will be described; for an in-depth presentation, as well as experimentation results, see Chapter 10.
9.4.1 Training the Mapping Model

Words are usually polysemic: depending on the *linguistic context*, the same word carries different meanings. Thus it is, in general, not possible to map a given word onto one and only one concept. Instead, the relationship between concepts and words is in general many-to-many. We rely on a particular combination of stochastic models to provide such many-to-many mappings: the Mapping Model.

The Mapping Model’s training process is divided into several phases. First of all, a specific set of documents in the Document Repository enter the Linguistic Context Extractor that retrieves morphologic and syntactic information (the linguistic “context”), by means of tools such as Freeling [2], the Stanford POS tagger [19], the Stanford parser [13, 5], and the JavaRAP [14] coreference resolutor.

Then, a human expert associates each word with the right concept, creating two sub-sets: the *training set* and the *test set*. The Training Procedure, relying on both the ontology and the semantic network, analyses the training set documents, and generates the statistical model. Finally, the Mapping Model is checked against the test set.

As an example, consider the sentence “The bottle with red label contains Barolo, a red wine”. The human expert associates words and concepts, producing “The bottle/WineBottle with red label/WineLabel contains Barolo/barolo wine, a red/red wine/wine”. Notice that the adjective “red” just before the noun “label” is not tagged, as its meaning (the colour of a label) is not considered useful for the domain (i.e., there is no related concept into the Domain Ontology). Instead, just before the noun “wine”, the adjective “red” refers to a wine colour, and therefore it is tagged with the related concept.

9.4.2 Indexing Documents

The Text Extractor module gathers text from several document formats (PDF, HTML pages, etc.), provided as URLs or directly uploaded by users. The extracted text, as well as the original document file (if any), are stored into the Document Repository.

Then, the text undergoes the indexing process, performed by the Information Extraction Engine, which –exploiting the Domain Model– updates the Conceptual Index: For each word of the documents, the engine selects the re-

---

2 The linguistic context represents the information one can extract by reading the text; the linguistic context of a word is composed of the surrounding words, augmented with morphologic and syntactic information.
lated concept of the Domain Ontology. Thus, the Conceptual Index provides an abstract view of the documents, permitting users to search for concepts.

The Information Extraction Engine searches the Document Repository for newly added documents, retrieves morphologic and syntactic information (the “context”). Then, a mapping algorithm, relying on the Domain Model, maps documents’ words to the related ontology concepts. Finally, a classic TF-IDF scheme is applied, in order to generate the documents’ concept vectors, stored into the Conceptual Index. The engine can also decide that a word does not fit any of the Domain Ontology concepts, as its meaning is outside the domain of interest.

The concepts-words mappings generated during the indexing procedure are then arranged as an index and stored into the Conceptual Index.

Continuing the example, assume the system be trained to recognise concepts in the following set \{WineBottle, WineLabel, barolo wine, red wine, Wine\}. If the document set is composed of $D_1$ = “Barolo is a small village where good wine is produced” and $D_2$ = “Barolo is a red wine”, the system extracts the following sets of concepts $C_1$ = \{Wine\} \(^3\) and $C_2$ = \{barolo wine, red wine, Wine\}. The Concept Frequency Analyser calculates the weight \(w_{c,d}\) associated to each concepts \(c\), for each document \(d\); thus, the following vectors are generated: $V_1$ = \([0, 0, 0, w_{\text{barolo wine},D_1}], w_{\text{Wine},D_1}\)$ and $V_2$ = \([0, 0, w_{\text{barolo wine},D_2}], w_{\text{red wine},D_2}, w_{\text{Wine},D_2}\]$.

### 9.4.3 Querying Documents

The Query Engine, exploiting both the Conceptual Index and the Domain Model, permits to formulate concept-based queries on the document collection. Searching for a given concept, the system finds every mapped word (as well as its synonyms, thanks to the Semantic Network), and calculates a ranked list of documents. The Query Engine accepts several query models: from simple lists of keywords, as usually found in traditional search engines, to complex sentences expressed in natural language.

Keyword based queries are composed of a sequence of words, connected by either AND or OR boolean logic operators. The system maps each word to the related concept, and then search the Conceptual Index for these concepts; in the AND case only documents containing all the concepts are returned; in the OR case, documents containing at least one concept are returned.

As an example of AND query, if the user issues $Q$ = “Barolo wine”, the system indexes the text and produces the following sequence of concepts: $V_Q$ = \([0, 0, w_{\text{barolo wine},Q}, 0, w_{\text{Wine},Q}]\). Then, compares the vector $V_Q$ against $V_1$ and $V_2$, and finds the nearest document.

---

\(^3\) Notice that in $D_1$ “Barolo” refers to a village and should not be considered.
Phrase based queries are written in natural language, as questions or descriptions. Such queries rely on three main steps. First, the phrase issued by the user is indexed as if it were a sentence in a document; the result is a sequence of concepts. Second, such concepts, as well as the contextual information, are analyzed by a unification-based parser, which finds the appropriate structure tree (relying on a feature-based grammar, defined by a human expert, which defines the structure of phrases the system should be able to parse.) Third, an SQL query template (also defined by a human expert) is selected and filled with nodes of the parse tree, and the resulting query is executed.

As an example of phrase-based query, assume that the Domain Ontology contains information about the facts that wines have colours, and that Barolo’s colour is red. Moreover, assume that an expert prepared the following query template 4: \(T = \langle \text{what}, \text{wine-attribute-name}, \text{wine-instance} \rangle\). If the user issues \(Q = \text{what is the colour of Barolo?}\), the system extracts \(C_Q = \{\text{WineColor}, \text{barolo wine}\}\), finds the phrase template \(T\), and in fills the placeholders: \(\langle \text{what}, \text{WineColor}, \text{barolo wine} \rangle\), and executes a parametric SQL query \(S_T\) associated by the same human expert, to the template \(T\). Notice that the template \(T\) is actually able to capture several requests, e.g. the colour of wine, the price of wine, etc. In other words, requests with similar structure (in terms of sequence and type of concepts) can be captured by means of the same template.

Notice that, as a further advantage of the concept-based queries with respect to traditional word-based engines, the search is multi-language in nature: the language used to search for documents does not depend on the language used to write them.

### 9.4.4 Extending the Domain Ontology

The Ontology Extender relies on Drools 5, a production-rule engine. Such rules predicate on information collected while indexing the text, to infer that a given word could be a new concept to be added to the ontology. However, rules cannot guarantee that the newly discovered association text-concept is sound; therefore, the system tags these new concepts as “guessed” to distinguish them from concepts defined by the human expert.

As an example, assume that the Domain Ontology contains information about the fact that wines have production regions. Moreover, assume that the system indexes the sentence “Piemonte produces wines” and associates words to concepts as illustrated in Figure 9.4. A simple rule could state that:

---

4 Just to get the idea, the actual definition relies on a feature-based grammar, which permits to specify templates in a way that can capture several different superficial forms of the same request.

5 See: [www.jboss.org/drools](http://www.jboss.org/drools)
IF subject_of\((W_1, \text{"to produce"}) \land \text{direct_object_of}(W_2, \text{"to produce"}) \land \\
mapped_on_class(W_2, \text{Wine}) \land \text{mapped_on_relationship}(\text{"to produce"}, R) \land \\
connected_by_relationship(R, \text{Wine}, C)\)
THEN
\(W_1\) is a new individual of \(C\)
and the system could infer that \(W_1=\text{"Piemonte"}\) should generate a new individual \(\text{piemonte}\) belonging to the class \(C=\text{Region}\).

Fig. 9.4 Extending the Domain Ontology: an example

9.5 The Internal Enterprise Data module

In the Art\(\text{Deco}\) scenario the involved actors are not only large enterprises but also small companies or even single users: a dedicated, human-intensive solution is not a reasonable way to integrate such a variety of heterogeneous information, and automated data wrapping and integration are needed. This is even more evident as, in the Art\(\text{Deco}\) environment, the data sources are not necessarily stable, and moreover they may vary from structured to semi-structured or even totally unstructured and dynamic, such as web pages. In order to be manageable and meaningful to the users, the input information of the Art\(\text{Deco}\) system must be machine processable and entails the need for explicit representation of data semantics.

The information management sub-system of the Art\(\text{Deco}\) project is thus devoted to providing a uniform, ontology-based semantic access to information coming from heterogeneous data sources. In Art\(\text{Deco}\), the choice was taken to keep a centralised repository, in the form of a Relational Database with an associated Data Warehouse, to store and manage all the interesting corporate information [15].

9.5.1 Structuring Web Data

In Art\(\text{Deco}\) the decision-making process is largely based on information obtained through the analysis of natural language documents published on the Web. There is no doubt that the entire process may greatly benefit (both
in terms of effectiveness and efficiency) from a structured representation of this information.

The need for structured data management is justified by several reasons. In the first place, the Knowledge Indexing and Extraction module produces as output a large number of facts about the domain for each document being processed (e.g., “Nebbiolo” is a DOCG lightly-coloured red wine, it is produced in Piemonte and has certain aromas). The size of these facts combined with the number of documents collected from the Web may produce a huge dataset that needs to be efficiently managed. The dataset produced by the extraction process must be eventually handed over a data-mining process [16] in order to extract some useful knowledge from it. Despite data-mining techniques for semi-structured and unstructured information exists in the literature, the ones developed for structured information (e.g., relational databases and data-warehouses [12]) are long established and far more effective, especially for business and market intelligence.

For this reason, we propose a (conceptual) modelling pattern for extending enterprise databases to enable Web-document storage and analysis. With reference to Figure 9.1, the DB component represents the relational (and possibly federated) database giving access to a suitable and integrated view of the enterprise data legacy that could be used to support knowledge extraction from Web documents. The schema of this database is constituted by two sets of tables: the first is a set of views over the enterprise database (Enterprise Tables) giving access to relevant information for the extraction process (e.g., a product’s description), the second is a set of tables used to structure the information coming from the extractors (Web-Search Tables). These tables are domain-independent, and thus can be used as a database modeling-pattern for ArtDeco-style applications.

The reference Relational Schema for the Enterprise Tables is shown in Table 9.1 while the conceptual and relational schemas for the Web-Search Tables are shown in Figure 9.5(a) and Figure 9.5(b) respectively.

The core table is FINDING which represent a finding in a web document that is considered “relevant” by the extraction engine. Tables DOCUMENT and SOURCE represent detailed information about the document and the data source (e.g., the website publishing the document). Table EVALUATION stores the associations between adjectives used to give a judgment or an opinion about a source or a finding in a web document (i.e., “wonderful” or “awful”) and a numeric value describing if a particular adjective has to be considered positive or negative.

Example 1. (Structuring Web Data)

Consider, as an example, the following sentence appeared in a blog entry titled “Barbera 2010: Pride in Simplicity?” found in the blog of a wine-expert in the US:

“[…so she poured us the unoaked wines, a fantastic Langhe Nebbiolo and a Barbaresco[…]]”
Table 9.1 The ArtDeco Enterprise Tables

<table>
<thead>
<tr>
<th>Conceptual Schema</th>
<th>Relational Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>FINDING(id, paragraph, row, start-column)</td>
<td>DOCUMENT(id, phys-uri, log-uri, pub-date, title, author, summary, topic, descriptor)</td>
</tr>
<tr>
<td>SOURCE(uri, name, abs-relevance, description, type)</td>
<td>EVALUATION(name, judgement, language, description)</td>
</tr>
<tr>
<td>S-E(source, evaluation, judgement)</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 9.5 DB Extension for NL Web Data: Conceptual and Relational Schemata
From this sentence we can retrieve at least some interesting data such as the fact that “Langhe Nebbiolo” and “Barbaresco” are “unoaked wines” and the fact that the author of the entry considers them “fantastic”. Moreover, the title of the document mentions a Wine Festival event in Italy (“Barbera 2010”) which is referring to a particular family of wines (i.e., the “Barbera”) and the two nouns “pride” and “simplicity” can be used to associate a measure of the consideration that the entry’s author has of this event. Each of these findings correspond to tuples in FINDING along with their position in the document. Information about the document and the website address will be stored in the corresponding table, along with the references to the tuples containing positive judgments for the adjectives “pride”, “simplicity” and “fantastic” in the EVALUATION table.

As already said, the Web-Search Tables are a modelling-pattern and they can be linked to the enterprise tables if necessary. Suppose, as an example, that one of the two wines mentioned in the entry of Example 1 is a product of the company running the ArtDeco system. In this case, it might be useful to link the findings to the information in the enterprise databases such as the product’s description, the bottle or grapevine’s data when they are available. From the modelling viewpoint, this corresponds to the introduction of a set of 1:n relationships between the table FINDING and the interesting tables as shown in Figure 9.6 where the bridge tables FND-G, FND-W, FND-B represents the conceptual connections between a web-search finding and a grapevine, a wine and a bottle respectively.

### 9.5.2 Dynamic and Heterogeneous Data-Source Integration

While the centralised repository is useful for analytical query processing, mediated on-line access to the original data sources should also be allowed, to provide stakeholders with a query interface for detailed and up-to-date access to the overall network knowledge. In ArtDeco, the dynamic integration of (semi-)structured data sources is carried out by the AD-DDIS component (see Chapter 13 for a detailed description). AD-DDIS first integrates the schemata of the various, heterogeneous data sources that may include XML documents and legacy and current databases from the networked business partners. Sensors data are handled as relational databases since we rely on the Perla Language (see Chapter 17) which provides a relational interface for such devices.

A model of the domain semantics is captured by the Domain Ontology, a shared conceptualisation of the application domain, which is used as Global Schema and later mapped to a set of ontologies, each representing the schema of one data source. Each data source ontology is an ontological description of the data source schema and it is extracted by means of domain-aware wrappers i.e., the extracted description depends from both the target domain.
ontology and the data source schema. Data source ontology are then (semi-)automatically mapped to the domain ontology and such information will be later used to enable querying.

### 9.5.3 Knowledge Extraction

Once the information has been stored and structured into the database, it is possible to exploit standard warehousing and data-mining techniques in order to extract structured knowledge that was not explicitly present into the web pages. This is achieved by building a data-warehouse exploiting the methodology presented in [6]. Context-aware analytical queries are supported by the ArtDeco portal (see Chapter 14). Moreover, the data warehouse will be used to support data mining processes in order to analyse temporal trends [7] (e.g., which products are increasingly mentioned in the web-sphere), derive associations rules [1] (e.g., whose trend-setters participate in certain events), construct clusters of documents with similar features [8] (e.g., all the documents related to successful events) and outliers identifica-
to the aim of supporting the product design and innovation processes.

Determining such analysis processes is usually responsibility of the product designer and the R&D personnel. Once the various processes have been identified, the data-mining extension of the ArtDeco Web Portal will expose the proper graphical user-interfaces for them.

9.6 The PROM module: Collecting Data from Processes

The PROfessional Metrics (PROM) system\(^6\) consists of a set of tool for automated data collection from enterprise workflows and their subsequent analysis. PROM accesses and collects both code (i.e., interaction traces from running software) and process (i.e., email communication, supply-chain events, etc.) measures. This is important, since a comprehensive approach is the best way to collect data in a more complete way, in order to understand not only what workers produce, but also how they produce it.

The tool focuses on a comprehensive data acquisition and analysis methodology (see Chapter 11) in order to provide elements to improve products\(^{[18, 4]}\). The approach is based on a measurement framework that aims to achieve mainly two goals\(^{[10]}\): (a) to automatically measure the effort spent per artifact, and (b) to provide a framework that integrates existing tools that extract well known metrics with the measurements about the connected effort.

The system provides a non-invasive, automatic and accurate data collection and analysis, which allows to better understand the company-internal workflows. As an example, consider the process of acquisition of grapes in a wine-farm. While some of the producers will be part of an integrated supply-chain and thus supported by a supply-chain management system that can be easily monitored by software probes, some other producers, especially small farmers, will handle their selling process to the wine-farm through paper purchase-orders and receipts. On the wine-farm-side, this corresponds to manual handling of acquisitions with consequent manual insertion of data, email/document archiving etc. In order to understand and improve the process it is thus necessary to monitor as much as possible of these events by collecting data from every application (email clients, spreadsheets, word processors, etc.) used inside the enterprise.

Through an investigation performed over the collected data, it is possible to conduct an analysis of the company processes (with a special focus on its strengths, weaknesses, opportunities and threats). On the base of this analysis is thus possible to improve workflows, to provide decision support for IT purchases, or to increase the quality of developed products by choosing

\(^6\) G. Succi. Professional Metrics (PROM) Home Page: [www.prom.case.unibz.it](http://www.prom.case.unibz.it)
appropriate development methodologies. Using such approach, companies can benefit from a more detailed picture of the state of company-internal workflows and processes, and they can determine the quality of developed products.

PROM is a client-server system, which is organised on: data collection from plug-ins that are integrated in development tools and other applications, data analysis in a central repository, and knowledge creation in suitable components (see Figure 9.7). The data collection tools store their measurements locally in form of xml files and, when a connection is available to the server, they send them to the central database using web services.

![Fig. 9.7 The PROM Architecture](image)

### 9.7 Conclusions

In this chapter we presented the ArtDeco Web Portal, which aims at providing an advanced document and data repository, shared among the network partners.

The portal provides functionalities that permit to discover, extract, and find information from structured, semi-structured, and non-structured data sources.

A Domain Model contains the concepts relevant for the application domain, as well as the related vocabulary.
References