Linköping Electronic Articles in
Computer and Information Science
Vol. n(199n): nr nn

Knowledge Mediation in the WWW Based on Labelled DAGs with Attached Constraints

Jutta Eusterbrock
Softlution WebTechnology GmbH
Bgm.-Neff Str. 13, D-68519 Viernheim
Email: j.eusterbrock@softlution.de

Linköping University Electronic Press
Linköping, Sweden

http://www.ep.liu.se/ea/cis/199n/nnn/

Published on February nn,, 199n by
Linköping University Electronic Press
581 83 Linköping, Sweden

Linköping Electronic Articles in
Computer and Information Science
ISSN 1401-9841
Series editor: Erik Sandewall

Typeset by the author using LaTeX
Formatted using étendu *style*

Recommended citation:
<Author>. <Title>. Linköping Electronic Articles in Computer and Information Science, Vol. n(199n): nr nn. http://www.ep.liu.se/ea/cis/199n/nnn/. *February nn,, 199n.*

This URL will also contain a link to the author's home page.

Abstract

This paper introduces a framework where the semantic access to Web resources is realised by a mediator module. Users can express their knowledge needs using the terms of a shared ontology without taking into account context-specific locations, encodings, and vocabulary. Integration among shared ontologies and Web meta data descriptions is achieved by inserting an intelligent meta-level layer using lifting rules. Lifting rules can be used to automatically decompose semantic information needs into concrete information requests. Wrapper and facilitator agents are used to find suitable information on the Web.

Labelled directed acyclic graphs (DAGs) allow a natural modelling of syntactical, structural and semantic properties of standardised XML Web data meta data, and ontologies, and their operations. However, graph-based data models are only a first step towards intelligent tools for automated knowledge extraction from the WWW. Planning and design synthesis of technical systems involve dealing with interacting constraints, imposed by selected tasks, such as cost or compatibility constraints, or general constraints.

As a step to cope with these requirements, a novel approach combining constraint processing and reasoning with Web (meta-)data is presented. Canonical graphterms are used for the encoding of labelled DAGs, which are embedded into constraint logic formulas, thus representing Web (meta-)data and ontologies. Graph rewriting and application of the built-in reasoning facilities such as matching, unification, subsumption and constraint solving of a constraint-logic language can then be used to efficiently automate mediation tasks.

1 Introduction

The World Wide Web (WWW) contains a vast amount of information valuable for planning and design tasks in engineering. Electronic product catalogues and online repositories, for instance, provide component descriptions which can support the human engineer as well as semi-automatic synthesis programs in finding appropriate parts. Engineering design involves usage of domain specific technical knowledge together with creative problem solving skills to select properly functioning components that comply with a set of requirements like performance goals, physical or cost constraints. Intelligent agent perform specific tasks on the behalf of users. The KRAFT project [PHG+99] analyses, how to fuse and transform selected semi-structured Web data and meta-knowledge, especially constraints, into a form which can be applied by intelligent agents in order to support further knowledge processing.

Web resources related to a domain may differ in syntax, structure and semantic content. The general idea of a *mediator system* is to integrate applications and heterogeneous information sources in such a way that they keep their autonomy (cf. [Wie92]). A mediator has to ensure that information requests can be transformed, distributed and matched by information offers which are stored as concrete resources. Common interchange formats, document type definitions (DTDs), and shared meta-level knowledge and data, like standardised conceptual models, called *ontologies*, have been identified as central issues for the construction of distributed information systems which take their input from Web resources. The e*X*tensible *M*arkup *L*anguage (XML) has emerged as a new standard for exchanging and representing data and structure on the Web. For a wide range of applications, XML allows users to define the structure of documents by means of Document Type Definitions. A DTD is essentially a formal grammar which restricts the structure of valid documents. Semantically similar product categorisations may coincide or differ in various structural aspects, such as equivalence, overlap, and inheritance of attributes or arithmetic measurement of values. An ontology is a formal explicit specification of an agreed standardised vocabulary, a precise definition of the basic terms and the relationships that can exist between them [Gru93]. Hence, the use of terms from an agreed ontology in product descriptions is suggested as a means to circumvent misunderstandings and support mediation. However, in many cases, the mediation task may require not only to match structures and semantic contents of documents, but more elaborate reasoning as well.

Among the many data models that can be used to enable reasoning and construction with the semi-structured information contained in Web resources and associated meta-information and knowledge, it is believed that graphs are the most adequate ones. Graphs provide natural algebraic means, for example projections and equivalence, to model knowledge transformation and sharing. Most important, labelled directed acyclic graphs offer a formal set-theoretic data model for subsets of the standard XML for exchanging Web data and representing meta-data, and for the design of query languages as XML-QL (cf. [D+98]).

Facilities to process interacting constraints and constraints which apply to the structure and content of Web resources are essential for the automatisation of complex tasks using the Web for data exchange. Using current data models for representing WWW (meta-)information, there seems to be no way for the declarative specification of constraints and, furtheron, means for the automated processing in the same formalism. However, it is a standard practice to combine logical reasoning and constraint solving in "constraint logics". In order to combine the advantages of an efficient graph-based handling of knowledge representation with the expressiveness and reasoning facilities of constraint logic, this paper analyses a logic-based representation framework which allows for efficient graph operations without losing the reasoning and constraint solving capabilities of constraint logic. It is based on graph-based data models for XML (cf. [GMW99]) and exploits a convenient formal correspondence between mathematical semantic structures (DAGs), syntactic term encoding, and logical specifications. This correspondence is extensively exploited in the SEAMLESS framework (cf. [Eus00]) which allows to specify and solve complex synthesis problems as is required for the mediator tasks described above. A graph term algebra and canonical forms for classes of equivalent objects which enable efficient graph operations within a logic-based framework are employed to implement the functionality.

The remainder of this paper is organised as follows. Section 2 presents an architecture for knowledge mediation on the Web. Section 3 describes the graph-based knowledge representation language. Section 4 shows how to formalise ontologies as concepts with attached constraints. Section 5 sketches the synthesis of rules that link ontological descriptions and DTDs, and describes the mediator shell. Finally, a summary concludes this paper.

2 An Architecture for Web Knowledge Mediation

The WWW can be considered as a market place where services and products are freely offered and requested. The general idea of a *mediator system* is to integrate applications and heterogeneous information sources in such a way that they keep their autonomy (cf. [Wie92]). A mediator has to ensure that semantic knowledge requests from users or applications can be transformed, distributed and matched by information offers which are stored as concrete resources, having resolved semantic ambiguities. Matching service descriptions have to be transformed back into high-level information, suitable for further processing. The KRAFT project has developed an agent-based architecture for distributed query processing [PHG+99]. All information processing units are realised as interacting software agents, using a subset of the Knowledge Query and Manipulation Language (KQML) performatives [FYM97]. The *facilitator* maintains a data base ("yellow pages") of all registered service providers and is able to map service types to appropriate providers. A provider first has to export the type of service it wishes to advertise via the facilitator. Information resources are coupled to the mediator, using *wrapper* modules. Information is exchanged in the KRAFT Constraint Interchange Format (CIF).

With the vast amount of information and hidden knowledge in Web resources, standards as XML and DTDs for structuring, typing and semantic conceptual abstractions as shared ontologies have become evident. Using current data models for WWW information, there seems to be no way of declaring basic types and to constrain admissible values in the same formalism. The presented work establishes means that enable integration, representation and reasoning using Web standards, semantic concepts, and domain-specific knowledge like constraints. Mediation tasks as query processing and integration of heterogeneous information sources can be handled more effectively. Problem solving tools can re-use integrated and trans-

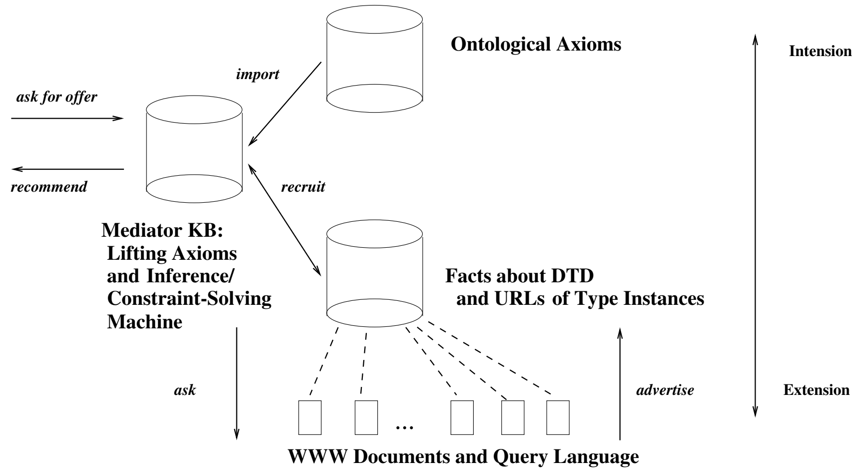

Figure 1: An Architecture for Mediation

formed knowledge for the automated solution of complex tasks. Knowledge integration and transformation is achieved by adopting the SEAMLESS multi-layer architecture (cf. [Eus00]). DTDs and ontology are incorporated into the distributed architecture through an intermediate context-dependent mediator knowledge base that in SEAMLESS terms provides a viewpoint. Figure 1 demonstrates the resulting architecture. The mediator architecture thus employs three essential technologies which will be shortly examined: "XML", "Shared Ontology" and "Viewpoints".

Maintenance of XML Documents The eXtensible Markup Language (XML) is a machine readable format for structured information representation on the Web and data exchange across networks and between heterogeneous information systems. An XML document primarily consists of a strictly nested hierarchy of tagged elements with a single root. Each tagged element has a sequence of zero or more attribute/value pairs, and a sequence of zero or more subelements. Standard vocabularies are summarised in Document Type Definitions (DTDs). A DTD is essentially a formal grammar which declares tags and their structural relations and restricts the structure of valid documents. For example, a simplified document type description for a PC-Modem card is sketched in Figure 2.

```
<!ELEMENT pc_modem_card(type,speed,price)>
<!ATTLIST modem model CDATA>
<!ELEMENT type(#PCDATA)>
<!ELEMENT speed(speed_unit,speed_measure)>
<!ELEMENT speed_value(#PCDATA)>
<!ELEMENT speed_unit(#PCDATA)>
<!ELEMENT price(value,unit,measure)>
<!ELEMENT value (#PCDATA>
<!ELEMENT measure(#PCDATA)>
<!ELEMENT unit(#PCDATA)>
```

Figure 2: XML Document Type Definition

Access to semi-structured XML Web data is realised through a number of query languages as XML-QL (cf. [D+98]) or Lore (cf. [GMW99]). The XML-QL example in Figure 3 selects in the document at `www.modem_company.com` all modems that support "Video-Conferencing" and generates a price list.

```
WHERE <pc_modem_card>
<type>video_conferencing</type></pc_modem_card>
<price><value>$value</value></price>
in www.modem_company.com
CONSTRUCT
$value
```

Figure 3: XML-QL Query

The *facilitator* knowledge base encompasses descriptions of all registered service providers, which are consistent with a given set of DTDs and found at specific URLs. This knowledge is stored as facts about DTDs and associated lists of the locations (URLs, Uniform resource locators) of individual services which are consistent with the type definitions.

Shared Axiomatic Ontology Different applications can associate different meanings with similar or identical terms. Whereas humans can (usually) distinguish between different interpretations of either the information content or transaction intent, agents will only be able to act meaningfully if the content exchanged between them, and the services they provide, carry sufficient explicit information. In order to enable consistent behaviours amongst the (virtual) participants in a virtual market and to allow complex interactions such as mediation, greater levels of semantic content have to be made explicit and represented in a computer readable form. Shared ontologies have been identified as central issue for the development of open distributed environments.

An ontology is a formal explicit specification of an agreed standardised vocabulary, a precise definition of the basic terms and the relationships that can exist between them [Gru93]. An ontology may specify knowledge and information on various abstraction levels. The OntoBroker [DEMS99] uses formal ontologies to extract, reason, and generate meta-data in the WWW. Various organisations develop international standards for business processes, models or product descriptions to enable the integration of heterogeneous applications and human-users for specific business processes across the supply and production chain. For example, RosettaNet, a consortium of manufacturers, distributors, resellers, end users, and solution providers, devises a framework for business-to-business electronic data interchange over the Internet. Within the RosettaNet framework, a technical standard for a universal laptop has been developed (cf. [Ros98]). Ontologies which standardise certain behavioural aspects of business transactions are also being developed.

Ontologies in this extended SEAMLESS KRAFT architecture shall exist as axiomatisations of generic domain models. Axioms identify a basic set of abstract properties, characteristics and associations for the semantic description of services, products or generic processes. For example, generic domain models may exist in the form of formal specifications for devices and components, using constraint logic programs.

Mediation by Viewpoints From the user's perspective, an ontology may provide a meta-level description on the underlying data. In this sense, ontologies act as generic semantic types, that is, they are intensional conceptual descriptions. A concrete information resource, for example a concrete document, may implement an abstract type with certain changes in structure, syntax and symbol names. In concrete inference processes about problem-related facts which need to be extracted from Web documents the system has to rely on knowledge about the related DTDs.

Applying the SEAMLESS architecture, a mediator knowledge-base can be constructed as a viewpoint. Generally, a viewpoint is an intermediate, loosely coupled component that provides the assumptions under which meta-level axioms become true in a particular context. It describes how to "lift" facts and data from the consulted local repositories to a more abstract level so that they can be compared and integrated with others from different sources. A viewpoint may contain additional domain knowledge that supports the integration task.

The viewpoint concept is applied to set-up mediator knowledge-bases that combine a shared ontology and DTDs. Lifting rules shall make causal relationship and applicable constraints among DTDs and a given shared ontology explicit. Using viewpoints supports easy and flexible construction of the mediator knowledge-base in the highly dynamic WWW context. On the basis of a shared ontology, users or software agents can pose semantic queries to the mediator knowledge-base without having to consider the location, the format or context-specific information of the underlying data. The mediator uses knowledge-based reasoning and embeds constraint solving to generate precise data requests. Individual records will then be retrieved using wrappers. Wrappers transform queries from the logical representation format to the existing XML source format, for example XML-QL, using the parsing rules and facilitator knowledge about locations and distribute them in a bi-directional way.

3 Graphs for Structured Web Knowledge Bases

The representation of the various domain-specific Web knowledge resources as XML documents, DTDs, local and shared ontologies in a form suitable for knowledge-based reasoning requires transformations on different levels. In [Eus00], it has been pointed out that graphs are an adequate means to transform data and information from heterogeneous sources on different abstraction levels into re-usable knowledge. SEAMLESS exploits a convenient formal correspondence between mathematical semantic structures (DAGs) or feature graphs, syntactic term encoding and logical specifications. It uses graphterms for the encoding of feature graphs as well as object graphs, and term rewriting for the implementation of operations. Having derived canonical term representations for classes of isomorphic DAGs allows to perform matching operations in an effective way [Eus97].

Graphs capture the structure of a wide range of data and knowledge modelling means as for example entity relationship diagrams. They are perhaps the most common natural means for capturing abstract structure of composed systems and to facilitate construction tasks. A complex site in the World-Wide Web has a natural abstraction as a graph where node parameters may be bound to page locations (URL's) and links describe structural and semantic relationships between parts of the hyper-document. For instance, a software or hardware system may be modularised as a graph or configuration of "components" or "parts" and "connectors" [Gar95]. The structure of a C-Web that is the properties associated to classes of documents and data sets and the various relations linking these classes together, is modelled by a labelled directed graph [Ch00]. Generally, when using graphs for knowledge representation, nodes represent data values, object identifiers or the domain of values and edges the relationships between them.

Most important, labelled directed graphs offer a formal set-theoretic data model for subsets of standards for exchanging or modelling Web data and meta-data, and for the design of query languages. Query languages like XML-QL (cf. [D^+98]) and data management systems like Lore (cf. [GMW99]) use labelled directed graphs to model (semi)-structured data and regular path-expressions to navigate through graphs. In the remainder of this paper, a slight variant of the graph data models [D^+98], [GMW99]) shall be used for the representation of XML documents.

Definition 1 *A XML graph consists of*

1. *A labelled directed graph G, in which each vertex is represented by a unique string called element identifier;*
2. *G's nodes are labelled with sets of attribute-value pairs, where each attribute-name is a string and each atomic-value has an atomic type drawn from integer, real, string or ID, $IDREF$ or $IDREFS$;*
3. *G's edges are either labelled with element tag identifiers, where each label is a string, or, attributes of type IDREF and IDREFS which point from the referring element to the referenced element;*
4. *G has a distinguished node called the root.*

Using the data model in definition 1, concrete XML documents can be mapped rather straightforward into syntactic graphs where certain nodes of the graph are annotated with semantic values of the constituent the node represents. For example, a modem document which conforms to the DTD in Figure 2 could be modelled by the XML-graph below.

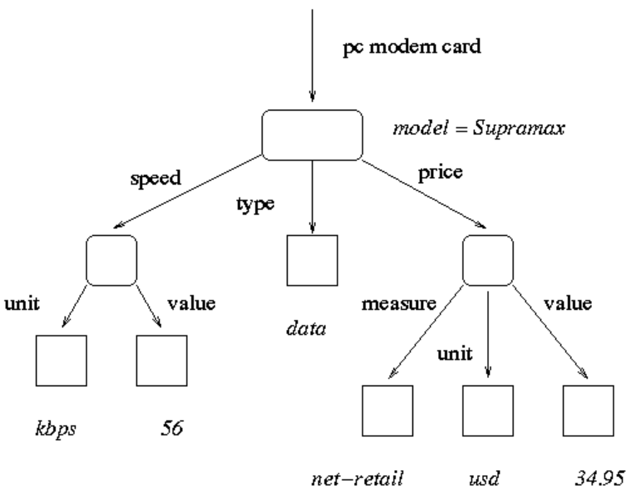

Figure 4: Graph modelling of a modem

An important abstraction mechanism for building knowledge bases is building a hierarchy of classes or concepts and attributes that characterise them, based on the content of their knowledge objects. The classes are ordered, based on the subsumption of their description. Hierarchical orderings can be exploited to build more efficient retrieval and matching procedures. In this case, feature graphs will be employed to structure sets of XML documents in an uniform graph-based framework.

Definition 2 *Suppose that there are given an infinite set of variables, a set of features, and a set of constants. A* feature graph *is a finite, rooted, connected and directed graph whose edges are labelled with feature symbols so that the labels of the edges are pairwise distinct. Every inner node of the feature graph is a variable and every terminal node is either a constant or a variable. Atomic objects have no outgoing edges and contain values of basic types. All other objects may have outgoing edges and are called complex.*

Linear (serial) term representations, named *graphterms*, were devised as data structure for the encoding of labelled DAGs and feature graphs [Eus97]. From features, constants and variables, terms are constructed, if features are seen equivalently as binary predicates that must be interpreted as functional relations. Let f be a feature symbol, X a variable. A *feature graphterm* is an expression $f(X, Graphtermlist)$, where $Graphtermlist$ is either the empty list [], or denotes a list of feature graphterms. To ensure that graphterms represent directed, acyclic graphs, further axioms constrain the valid terms. A graphterm algebra and canonical forms for classes of isomorphic objects were constructed (cf. [Eus97]).

The first step in setting-up a Web knowledge-base consists in extracting the *graph views* from Web resources stored in different persistent data sources to map it onto structures on which logical and algebraic operations apply. Graph views define specific syntactic graphs from existing Web resources. XML DTDs are wrapped into feature graphterms where tags are mapped into feature symbols. Each node variable could be associated with a set of properties by an attribute list. Parsing rules from XML DTDs into feature graphs have to specify the syntax - syntax relation. XML elements and DTDs are represented using the linear recursively constructed graphterms of the format $element(AttributeName(Value), ListofSubelements)$ or $element(Variable, ListofSubelements)$, where $ListofSubelements$ could be the empty list. This results into the encodings in Figure 5.

$$
\begin{array}{l}
pc\_modem\_card(model(Name), \\
\quad [type(T, []), \\
\quad speed(\_, [speed\_value(Sp, []), speed\_unit(Kbps, [])]), \\
\quad price(\_, [value(V, []), unit(U, []), measure(M, [])])]). \\
\\
pc\_modem\_card(model("Supramax"), \\
\quad [type(data, []), \\
\quad speed(\_, [speed\_value(56, []), speed\_unit(kbps, [])]), \\
\quad price(\_, [value(34.95, []), unit(usd, []]), measure(net\_retail, [])])])
\end{array}
$$

Figure 5: Graphterm Representation PC Modem Card and its DTD

Unifying information requests and information offers is the central operation when designing mediators. To define that two structured graph objects “match”, it is recalled that structured representations are merely an alternative kind of predicate calculus formalism. Two objects match if and only if the predicate calculus formula associated with one of them unifies with the predicate calculus formula associated with the other. Canonical forms enable effective graph storing and matching within a logic-based framework. Various operations on graphs that enrich the built-in facilities of a logic programming language were devised and implemented. These functions can be used to query graph-based knowledge bases, to construct or transform graphs and reason with them. Especially:

- Conjunctions of sequences of features (path expressions), which may include wildcards, for example, $conceptsymbol.feature_1.feature_2\ldots$, can be used to retrieve matching concepts.
- Let O be an XML graph and C, C_1, C_2 be feature graphs and $var(C)$ denote the set of variables of any expression. The graph O is an instance of the feature graph C, C *subsumes* O, if there exists a term substitution σ such that $\sigma(C) = O$, supposing canonical term representations.

4 Formalising Domain Ontologies with Constraints

Having introduced models to formalise semi-structured data and DTDs, this section aims at formal models for the specification of ontologies. An ontology consists of domain concepts and further background knowledge. Feature graphs provide a means for the representation of domain concepts. Semantic information on the domain of individual type variables provides valuable knowledge for subsequent effective information inferencing. However, current approaches to ontological engineering as for example taxonomies or frame logic don't provide means to deal with arbitrary constraints. Feature graphs have an isomorphic declarative logic representation. While it is a standard practice to combine logical reasoning and constraint solving in "constraint logics", in the specific case of reasoning with graph-structured knowledge the straightforward translation approach will suffer from the generally inefficient handling of this kind of inferences in general-purpose logic. In order to combine the advantages of the more efficient term-based handling of feature graphs used in SEAMLESS with the expressiveness of constraint logic, a combined framework which allows for efficient graph operations without losing the constraint solving capabilities is presented.

Definition 3

- *A* constraint satisfaction problem *(CSP) is given by*
  - *a finite set of variables* $\{X_1, \ldots, X_n\}$*;*
  - *a function* $X_i :: D_i$ *which maps every variable* X_i *to a finite set* D_i*, its domain;*
  - *a finite set of* constraints *which restrict the values that the variables can simultaneously take. For example, an arithmetic constraint is an expression of the form* $X_1\, R\, X_2$ *where* R *is a relation symbol* $\{=, \neq, >, \geq, <, \leq\}$ *and* X_1, X_2 *are either variables or finite domain constants.*
- *A* solution *to a constraint satisfaction problem is an assignment of a value from its domain to every variable, in such a way that each constraint is satisfied.*

Exploiting the logical correspondences among graphs and logical specifications, it is proposed to model semantic constraints as constraint satisfaction problems that augment the declarative feature graph specifications. Domain concepts are represented as constrained feature graphs. Further background knowledge is represented by constrained feature clauses. These are clauses which embed graphterms that are interpreted as feature graphs.

Definition 4

- *A* constrained feature graph *is an expression*

$$\begin{array}{l} \forall X, X_1, \ldots, X_n : c(X, Gterm) \ \longleftarrow \\ \quad X :: D \ \wedge \ X_1 :: D_1 \wedge \ \ldots \ \wedge \ X_n :: D_n \end{array}$$

- *A* constrained feature clause *is an expression*

$$\begin{array}{l} \forall X, X_1, \ldots, X_n : c(X, Gterm) \text{ or} \\ \forall X, X_1, \ldots, X_n : c(X, Gterm) \ \longleftarrow \\ \quad X :: D \ \wedge \ X_1 :: D_1 \ \wedge \ \ldots \ \wedge \ X_n :: D_n \\ \quad\ \wedge \ Constraints \ P_1 \ \wedge \ \ldots \ \wedge \ P_m \end{array}$$

  where

  - *"Gterm" refers to a feature graph, "$var(GTerm)$" $= X_1, \ldots, X_n$, "Constraints" refers to a finite conjunction of constraints, and $P_1, \ldots, P_n$ denote atoms. An atom is a predicate applied to constants, variables or terms, especially graphterms.*
  - *the sets of variables are disjoint:* $\{X, X_1, \ldots, X_n\} \cap var(P_i)_{i=1,\ldots,m} = \emptyset$. *Joins are made explicit by replacing shared variables with explicit equality; thus* $p(X,Y) \longleftarrow q(X,Z)$ *would be replaced with* $p(X,Y) \longleftarrow q(X,Z) \wedge Y = Z$.
  - *similarly, value assignments are made explicit; thus* $p(X,a)$ *is replaced by* $p(X,Y) \longleftarrow Y = a$.

Figure 6 illustrates the representation of a shared concept for a modem using constrained feature clause.

$$\begin{array}{l} \forall Name, \ldots : modem(Name, [\\ \qquad interface(S, []), \\ \qquad type(T, []), \\ \qquad speed(\_, [speed\_value(Sp, []), speed\_unit(Kbps, [])]), \\ \qquad linux\_compatibility(model(ModelL), [status(CStatus, [])]), \\ \qquad in\_stock(model(ModelS), [status(SStatus, [])]), \\ \qquad price(\_, [value(V, []), unit(U, []), measure(M, [])])]) \ \longleftarrow \\ \qquad\qquad S :: [external, internal] \ \wedge \\ \qquad\qquad T :: [data, fax, voice, video\_conferencing] \ \wedge \\ \qquad\qquad Sp :: [28.8, 33.6, 56] \ \wedge \ KB = kbps \ \wedge \\ \qquad\qquad SStatus :: [yes, no, limited] \ \wedge \\ \qquad\qquad CStatus :: [yes, no, limited] \ \wedge \\ \qquad\qquad V > 0 \ \wedge \ U = euro \ \wedge \ M = gross\_retail\_price. \end{array}$$

Figure 6: Formal Representation of the Domain Concept Modem

As each mediation starts with a user request formalised as query, first the procedure is sketched which retrieves those ontological concepts which are consistent with the query.

- A query is consistent with a concept description, if and only if the canonical graphterms of query and the ontological specifications match, i.e. share the same term structure, and the union of both constraint sets is satisfiable.

5 Knowledge-based Mediation

The goal of this paper is to design a mediator that enables users or knowledge-based applications an integrated semantic access to XML encoded Web resources content and structure via ontologies, exploiting meta-knowledge, especially constraints and applying automated reasoning and constraint solving.

Consider the following scenario. A broker wants to deliver communication devices that match consumer requirements at all times. The broker has relationships with various parts manufacturers all over the world. Producers can advertise their product information to the facilitator by providing machine processable meta descriptions of multiple Web resources. Meta descriptions are submitted in the form of the DTDs that are used to structure the parts catalogue, such as price or weight, and the locations of information resources. The consumers formulate their requirements using a convenient Web interface at the broker site. Requirements are goals containing complex combinations of conjunction of atoms, graph-path expressions and additional constraints on the occurring variables. Queries are expressed using terms from the shared ontology, without having to know about the location or content of context-dependent dynamically changing information resources. Figure 7 illustrates how a typical goal combines constraints of different types:

$$\begin{array}{l}
\exists Model, Price : modem(model(Model), _) \\
\wedge\ modem.speed \rightarrow max \\
\quad \wedge\ modem.linux_compatibility.status = ok \\
\quad \wedge\ modem.deliverable.status = ok \\
\quad \wedge\ Price = modem.price.value \quad \wedge \quad Price < 40 \ \wedge \\
\qquad\qquad\qquad modem.price.unit = euro.
\end{array}$$

Figure 7: Example Query with Constraints

One of the problems in satisfying this requirement is that the syntactic terms used by producers and consumers don't match. For example, even if a remote parts manufacturer has a required component in stock, the broker can't deliver immediately, because the shipping time has to be taken into account. Producers and consumers may use different measures, eg. "prices before tax" versus "prices after tax", and units, eg. "dollar" versus "euro", to calculate prices. Supplementary background knowledge, that is not provided by the producer, is needed to ensure that a specific modem is compatible with the Linux operating system.

The fundamental problem which shall be solved by the mediator is a semantic matching problem. That is, there is a goal object, the user request, expressed against the terms of a shared ontology, that has to be matched with semantic meaningful information. Information has to be constructed by exploiting additional knowledge, assembling and transforming information pieces from various Web resources.

An ontology in this framework consists of domain concepts that are encoded by a set of rules built of feature graphs, containing a set of variables with attached constraints and stored in a knowledge base. The facilitator knowledge base stores meta information of Web resources, based on DTDs and locations of resources, by non-instantiated facts using feature graphs. When integrating ontologies with meta information that describe various Web resources, DTDs and parts of the domain concepts may equal and differ in various syntactic, symbolic and structural aspects such as renaming of symbols, overlap or set containment of attributes and features, and measurement of units, although they share some common semantics or an application objective. Knowledge sharing among parts of the recursively constructed domain concepts and DTDs is operationalised by mediation rules that are expressed by constrained feature clauses. Following examples show how knowledge sharing can be expressed.

1. **Specialisation**:

$$\begin{array}{l} modem(\_,\_) \;\longleftarrow\; modem.type = internal \;\wedge \\ \qquad pc\_modem\_card(\_,\_) \end{array}$$

2. **Overlapping domain values**:

$$\begin{array}{l} speed\_measure(S_1, []) \;\longleftarrow\; speed\_measure(S_2, []) \;\wedge \\ \qquad S_1 = S_2 \;\wedge\; S_1 \in \{28.8, 33.6, 56\} \;\cap\; \{33.6, 56\} \end{array}$$

3. Equivalence modulo **measurement of units**:

$$\begin{array}{l} price(\_, [value(GP, []), unit(GU, []), measure(GM, [])]) \\ \longleftarrow\; GM = retail\_price \;\wedge\; GU = euro \\ \qquad \wedge\; LM = net\_retail\_price \wedge\; LU = gbp \\ \qquad \wedge\; GP = LP * ((1 + VAT/100) * Exchange) \\ \qquad \wedge\; vat(VAT, []) \\ \qquad \wedge\; exchange\_rate(\_, [target(GU, []), source(LU, []), \\ \qquad\qquad rate(Exchange, [])]), \\ \qquad \wedge\; price(, [value(LP, []), unit(LU, []), measure(LM, [])]) \end{array}$$

Parts of the recursively structured domain concepts have to be associated with concrete context-specific information stored in various XML resources. Starting from some given initial set of knowledge sharing axioms, mediation rules can be synthesised. The general idea of the synthesis process is to derive a conjunction of premises and constraints iteratively by finding equivalent DTDs for parts of the concept and fusing the constraints. The synthesis process terminates when the domain concept is decomposed into a set of subconcepts so that all subconcepts are associated with DTDs. Figure 8 visualises a synthesis step.

The synthesis process is guided by the following logical rules.

$$\begin{array}{l} DomainConcept \;\longleftarrow\; Constraints_x \;\wedge\; DTD_x \\ Subconcept \;\longleftarrow\; Constraints_y \;\wedge\; DTD_y \\ Subconcept \;\subset\; DomainConcept \\ \vdash \\ DomainConcept \;\longleftarrow\; Constraints_x \;\cup\; Constraints_y \;\wedge \\ \qquad DTD_x \;\wedge\; DTD_y \end{array}$$

For example, the generated clause in Figure 9 couples the context-specific description of a "PC modem card" (cf. Figure 5) and further information resources with the shared domain model introduced by the ontology.

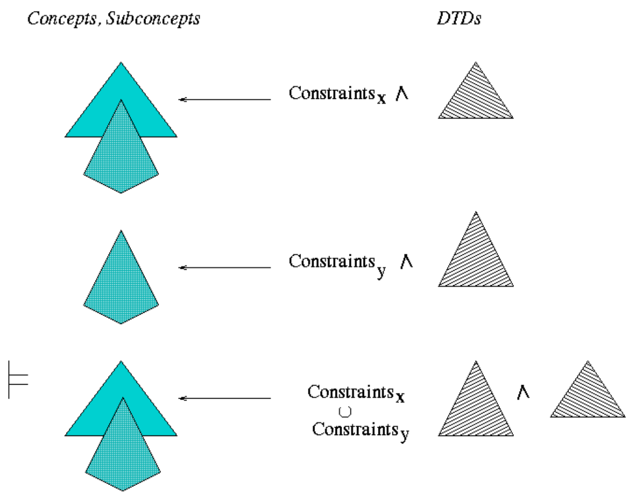

Figure 8: Generating Rules by Substitution of Equivalent Subgraphs

A mediator shell is realised as a viewpoint that uses various methods for knowledge-based and graph-based reasoning of the SEAMLESS system, and the knowledge-base with shared ontologies, the context-dependent mediation rules, the facilitator facts about locations and further design knowledge. Employing the proposed representation formats constrained feature graphs and constrained feature clauses, the knowledge base is considered to be a specific constraint logic program. *Constraint logic programs* contain an inference engine and constraint solvers. Graph selection and application of the built-in reasoning services – such as matching, unification and constraint services – of a constraint-logic language can then be used to efficiently implement basic mediation steps. Applying the proposed framework, the mediation task is achieved rather straightforward and in an automatic way by the following steps in Algorithmus 1:

Algorithm 1 *1.* Transforming *the user request into a logical query by searching concept descriptions from the ontology knowledge base that match the graph path expressions and satisfy the attached constraints;*

2. Decomposing *the logical request into information requests to external resources and additional constraints, using the lifting rules and applying the built-in facilities of the constraint-logic programming language.*

3. Locating *suitable external resources, using the facilitator knowledge base and* wrapping *the information request into a XML document query;*

4. Integrating *returned values into the CSP and testing satisfiability. Whenever a solution to the CSP resulting from a query and a candidate concept description can be computed, the corresponding database entries are regarded as valid answers. Satisfiability causes composition of the single results. Non-satisfiability causes backtracking until all possible information resources are searched.*

$$
\begin{array}{l}
modem(model(GName), [\\
\quad interface(GS, []), type(GT, []),\\
\quad speed(\_, [speed\_measure(Sp, []), unit(Kbps, [])]),\\
\quad linux\_compatibility(model(ModelL), [status(CStatus, [])]),\\
\quad in\_stock(model(ModelS), [status(SStatus, [])]),\\
\quad price(\_, [value(GP, []), unit(GU, []), measure(LM, [])])])\\
\qquad GS = internal \ \wedge\\
\qquad GSp :: [33.6, 56] \ \wedge\\
\qquad GT :: [data, fax, voice] \ \wedge\\
\qquad GName = ModelL \ \wedge \ GName = ModelS \ \wedge\\
\qquad SStatus :: [ok, no, limited] \ \wedge\\
\qquad CStatus :: [ok, no, limited] \ \wedge\\
\qquad GU = euro \ \wedge \ GM = gross\_retail\_price \ \wedge\\
\qquad LU = usd \ \wedge \ LM = net\_retail\_price \ \wedge\\
\qquad GName = LName \ \wedge\\
\qquad GSp = LSp \ \wedge\\
\qquad GP \ is \ LP * ((1 + VAT/100) * Exchange) \ \wedge\\
\qquad vat(\_, [country(Country, []), vat\_rate(VAT, [])]),\\
\qquad exchange\_rate(\_, [target(GU, []), source(LU, []),\\
\qquad\quad rate(Exchange, [])]),\\
\qquad linux\_compatibility(ModelL, [status(CStatus, [])]),\\
\qquad pc\_modem\_card(model(LName),\\
\qquad\quad [type(T, []),\\
\qquad\quad speed(\_, [speed\_value(LSp, []), speed\_unit(Kbps, [])]),\\
\qquad\quad price(\_, [value(LP, []), unit(LU, []), measure(LM, [])])]).
\end{array}
$$

Figure 9: Mediator Lifting Rule

Automated processing of the query in Figure 7 with respect to the mediator knowledge base in Figure 9 and accessing the information about the "pc-modem-card" described in Figure 5, the exchange rate $o.94$ for euro against "usd", "linux-compatibility" and "in-stock", yields $X = "Supramax", GP = 38.44$ taking into account the VAT as well. One of the problems that the mediator architecture addresses is the flexible integration and transformation of multiple dynamic changing resources into a re-usable form via a shared ontology. Exploiting the built-in reasoning and constraint-solving facilities of a constraint-logic programming language, several tasks as constraint rewriting are done in an automatic way.

6 Conclusion

A mediator in agent-based architectures is a flexible component that provides a currently missing link between a lot of interesting data (which resides on the Web) and the applications (which have no direct access to the Web data), using wrappers and facilitators.

Efficient data extraction, storage, retrieval, transformation or modification, and reasoning are major issues when setting up an implementation strategy for a mediator that shall enable constructive reasoning re-using Web data. This paper has proposed a new representation framework for designing a mediator. It is based on labelled DAGs augmented by constraints as a combined model for representing syntactic patterns, structure and shared semantically meta-descriptions of Web resources. Labelled DAGs provide a cognitive and natural approach to (1) represent static structure of XML Web knowledge bases and DTDs, (2) to model knowledge sharing, and (3) to automate query-processing, transformations and constructions. Arithmetic, set-theoretic or boolean constraints are essential for modelling system components with respect to synthesis and design tasks. Neither current XML standards nor graph-based models for Web data support the use and exchange of constraints. In [Eus97], canonical term representatives for isomorphic DAGs and a graph term algebra for implementing various operations on graphs were devised. These results allow storing structured graph representations of Web resources as terms embedded into logical knowledge specifications. Constraints to be applied to the structure or content of Web resources can be described as constraints attached to the graph term variables. Certain operations for retrieval, modification and reasoning that might otherwise have been performed by either combinatorial search or explicit specialised algorithms can be performed in a more efficient way by mechanisms that depend on the structure of the representation and using the built-in algorithms for constraint-solving, unification or subsumption of a constraint-logic programming language. Wrappers which translate XML data and meta-data and XML-QL queries into logical structures can be easily built.

There has been some interest recently in moving data from the WWW into knowledge that can be re-used by intelligent applications. Roughly related projects in this area use an ontology approach for specifying the domain of discourse as in HERMES [A95], and declarative languages for specifying mediators and performing reasoning for the automatisation of mediation tasks as in KOMET [CJKS97]. ONTOBROKER [ES00] uses ontologies that are formalised in Frame Logic to derive DTDs, whereas the presented concept of lifting rules provide a flexible means for the linking of independent ontologies and DTDs. [Kel95] presents an agent-based architecture for smart catalogues and brokering that supports cross-search of multiple heterogeneous SQL-based product data bases. Ontologies are used to define common terms. The KRAFT (Knowledge Re-use and Fusion Transformation) architecture [PHG+99] focuses on the utilisation and reuse of constraint knowledge held in distributed sources, by transforming it for use with respect to various ontologies using a non-executable interchange format.

Future work will focus on the loose integration of concrete common ontologies and Web resources as knowledge bases into the SEAMLESS synthesis and learning system [Eus00]. The graph-based methodology itself can be extended to other semi-structured information bases.

References

[A95] S. Adah and R. Emery ˙A uniform framework for integrating knowledge in heterogeneous knowledge systems. In *Proceedings 11th International Conference on Data Engineering*, pages 513–521. Philadelphia, 1995.

[Ch00] V. Christophides. Community Webs (C-Webs): Technological Assessment and System Architecture. Research report, C-WEB IST-1999-13479.

[CJKS97] J. Calmet, S. Jekutsch, P. Kullmann, and J. Schü. Komet - a system for the integration of heterogenous information sources. In *Proc. ISMIS'97*, pages 318–327. Springer Verlag, 1997.

[DEMS99] S. Decker, M. Erdmann, D. Fensel, and R. Studer. Ontobroker: Ontology based access to distributed and semi-structured information. In

R. Meersman et al, editor, *Database Semantics: Semantic Issues in Multimedia Systems, Proceedings TC2/WG 2.6 8th WorkingConference on Database Semantics (DS-8)*. Rotorua, New Zealand, Kluwer Academic Publishers, Boston, 1999.

[D+98] A. Detsch, M. Fernandez, D. Florescu, A. Levy, and D. Suciu. XML-QL: A query language for XML, 1998. "http://www.w3.org/TR/NOTE-xml-ql".

[ES00] M. Erdmann and R. Studer. How to structure and access XML documents with ontologies. *Data and Knowledge Engineering, Special Issue on Intelligent Information Integration*, page to appear, 2000.

[Eus97] J. Eusterbrock. Canonical term representations of isomorphic transitive DAGs for efficient knowledge-based reasoning. In *Proceedings of the International KRUSE Symposium, Knowledge Retrieval, Use and Storage for Efficiency*, pages 235–249, 1997.

[Eus00] J. Eusterbrock. Composing reusable synthesis methods through graph-based viewpoints. In S. Hölldobler, editor, *Intellectics and Computational Logic, Papers in Honor of W. Bibel*, pages 143–158. Kluwer academic publishers b.v., 2000.

[FYM97] T. Finin, Y. Labrou, and J. Mayfield. KQML as an agent communication language. In Jeffrey Bradshaw, editor, *Software Agents*. AAAI/MIT Press, Cambridge, 1997.

[Gar95] D. Garlan. Research directions in software architecture. *ACM Computing Surveys*, 27(2):257–260, 1995.

[GMS95] C.H. Goh, S.E. Madnick, and M.D. Siegel. Ontologies, contexts, and mediation: Representing and reasoning about semantic conflicts in heterogeneous and autonomous systems. Technical report, Sloan School of Management Working Paper 3848, 1995.

[GMW99] R. Goldman, J. McHugh, and J. Widom. From semistructured data to XML: Migrating the Lore data model and query language. In *Proceedings of the 2nd International Workshop on the Web and Databases (WebDB'99)*, pages 25–30. Philadelphia, 1999.

[Gru93] T.R. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993.

[AP95] F. Gudermann, A. Puder, and S. Markwitz. Service trading using conceptual structures. In *3rd International Conference on Conceptual Structures*. Springer Verlag, 1995.

[Ros98] Inc. RosettaNet. RosettaNet laptop technical specification. Final Draft for Member Vote, 1998.

[Kel95] A.M. Keller. Smart catalogs and virtual catalogs, 1995.

[RGM98] A.G. Moukas R.H. Guttman and P. Maes. Agent-mediated electronic commerce: A survey. *Knowledge Engineering Review*, 13(2):147–159, 1998.

[PHG+99] A. Preece, K. Hui, A. Gray, P. Marti, T. Bench-Capon, D. Jones, and Z. Cui. The KRAFT architecture for knowledge fusion and transformation. In *Research and Development in Intelligent Systems XVI (Proc. Expert Systems 99)*, pages 23–38, 1999.

[Wie92] G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, pages 38–49, 1992.