Designing Personalized Digital Libraries over Web-sites with Semistructured Data*

L. A. Kalinichenko, N. A. Skvortsov, D. O. Briukhov,
D. V. Kravchenko, and I. A. Chaban
Institute for Problems of Informatics
Russian Academy of Sciences
ul. Vavilova, 30/6, Moscow, 117900, Russia
E-mail: [leonidk,scvora,brd,dmitry,chb]@synth.ipi.ac.ru

Abstract

Issues concerning the design of personalized digital libraries over collections of semistructured data available on the Web are considered. The approach suggested makes it possible to design libraries adjusted to the personal needs of information users that place various requirements upon the contents and representation of information. The digital library is designed as a composition of fragments of web-sites. In this paper a method for the compositional design of information systems is applied to semistructured data on the Web. This method was developed at the Institute for Problems of Informatics of the Russian Academy of Sciences (IPI RAS) [2]. The design procedure is applied to designing a library over two web-sites containing data about registered patents.

*Published in PROGRAMMING AND COMPUTER SOFTWARE Vol. 26, No. 3, 2000, pp. 123-133

1 Introduction

The approach developed in this paper is aimed at designing virtual digital libraries satisfying specific information requirements of users. The requirements are supposed to be formulated in the form of specifications of a collection that is created to satisfy the needs of a particular group of users. To refine the semantic contents of the specification, its elements are related to the concepts of the subject area ontology to which the library belongs. To use the information available on web-sites as a digital collection, it is necessary to know the specifications of site schemes related to the concepts of the subject area ontology of these sites as well. Our design methods are based on the principle of information reuse and the compositional design of information systems. Fragments of site descriptions that correspond to fragments of the requirement specifications are found such that parts of the site schemes could refine [7] the specification of the library scheme being developed. These fragments are sought on the basis of mapping the ontology of site subject areas into the shared ontology that is similar

to the subject areas of interest and on the basis of further analysis of scheme specification. Then, the library is built as a composition of the fragments found.

The software implementation of our approach is based on the SYNTHESIS CASE-system prototype developed in IPI RAS for compositional development of information systems [2].

The following subsections of Introduction (1.1–1.4) are devoted to basic concepts of our design method. In subsection 1.1, web-sites used as the source of the digital library are considered as databases with semistructured data. The basic principles of digital library personalization are presented in subsection 1.2. In subsection 1.3, the application of compositional design method for building digital libraries over web-sites is discussed. In subsection 1.4, the data model SYNTHESIS used to describe heterogeneous data and work with them in a unified way is briefly described.

In Section 2, an example is presented based on the sites of patent agencies in the US and in Canada. Schemes of these sites are described, along with the scheme of the specifications of requirements upon the personalized library. This example is used to illustrate methods described later in the paper.

In Sections 3 and 4, our design methods are described: approaches to representing ontologies, the ontological integration of site schemes and requirement specifications, finding relevant scheme fragments, and the composition into the desired collection.

1.1 Digital Collections with Semistructured Data on the Web

The Web is a rich source of information that can be used for creating digital libraries in various subject areas. However, it is difficult to determine the semantics of the available information because the data is unstructured or semistructured and the information is heterogeneous. This is one reason for the importance of the study of structuring information available on the Web.

Currently, investigations are being carried out on revealing the structure of data on web-sites with the aim of treating them as databases. For example, the structure of HTML-sites can be revealed by analyzing tags of hypertext documents and establishing regularities in their structure. The World Wide Web Consortium (W3C) adopted the XML standard [13]. According to this standard, the structure of a document is described in the Data Type Definition (DTD) section. A set of documents may have a common page structure or some documents may have unique structure. Thus, unstructured information presented on Web pages conforms to a certain scheme that corresponds to the site considered as a collection of documents.

In this paper, we do not consider the problem of revealing the schemes of HTML-documents. An example of the study of this problem is provided by the Araneus project [9], which uses the ADM model to describe schemes of HTML-sites, as well as specific approaches to reveal the schemes and manipulate the site data [5, 1]. The methods developed in the frame of the Araneus project can be conveniently used, together with our approach, for creating digital libraries: for this purpose it is sufficient to map site models described in the Araneus language into the canonical SYNTHESIS model (see Section 1.4). The approach suggested in this paper is designed to work with arbitrary representations of web-sites (including the representation provided by the ADM). In particular, mapping of the XML standard data model into the canonical SYNTHESIS model is considered in [8].

1.2 Designing Personalized Digital Libraries

The user-specific approach to servicing users of information suggests that there exists a procedure for registering groups of users and defining requirements of these groups. The specifications of users' needs may differ for different digital libraries. For example, a user may describe his requirements by expanding the thesaurus and the list of classification headings or specifying terms related to his field of interest and determining the list of classification headings where the data of his interest belong. At least, the interface of a particular user may include specific services and be different from standard interface of digital libraries or search engines.

In this paper, we restrict ourselves to considering the requirement specification in the form of the scheme of the digital library being designed over Web resources. Thus, the user is to specify the contents and the form of information he is interested in. In other words, the registration procedure consists of the specification of this scheme. Web-sites that can add useful information to the collection being designed are used as the initial collection.

1.3 Compositional Design

For users of a digital library, the very fact of obtaining the desired information is usually important; they rarely need to know the source of this information unless it was explicitly specified in the query. Scheduling the query execution and search for information sources is performed transparently for the user. Digital libraries over the Web may be considered as information systems that include a number of remoter data sources, each of which is represented in the global repository scheme of the system.

In the framework of our approach, a digital library includes registered digital collections of Web documents as its information components. The library is designed as a composition of fragments of the schemes of available information sources; it functions as a unified distributed system that includes these information sources as its components. The compositional design of digital libraries is based on refining the specifications [7] of requirements by the corresponding parts of the scheme specifications of the data available. According to our approach, the development of a library includes the following stages [2].

1. Reducing the collection scheme specifications to descriptions used in the canonical model [6]; relating specification elements to concepts of the ontological contexts of the corresponding subject areas; and mapping ontological descriptions of the subject areas of the collections and of the system being developed into the shared ontology. These operations are performed once for every site to prepare them for being used to design a digital library independently of the requirement specifications of the particular library.

2. Mapping the ontology of the subject area of the requirement specification into the shared ontology and ontological integration of collection schemes with requirement specifications. As a result, the description elements of site schemes are correlated with the requirement specifications describing the corresponding concepts.

3. Finding (among the ontologically relevant description elements of site schemes) those schemes that can be used to refine some fragments of the requirement specifications. Resolving structure conflicts and mismatches between schemes.

4. Implementation of the scheme of the desired digital library as a composition of the selected fragments of digital collection schemes. Developing wrappers for digital collections, which enables the digital library to use those collections.

1.4 Canonical Model for Representing Metadata in Digital Libraries

The SYNTHESIS language is used to construct a uniform representation of heterogeneous information resources and manipulate them in a uniform manner. It is important that SYNTHESIS can represent models of metadata of practically any kind and is extensible to adapt to the future development of data models and technologies. This language provides considerable opportunity for describing heterogeneous information resources, structured (databases), semistructured (hypertext documents), and unstructured (text documents) data, elements of knowledge bases, ontological specifications, activities, and workflows. The complete description of SYNTHESIS can be found in [6].

In SYNTHESIS, frames are used as description units. The object model of the language is built on the basis of frames. This is the notion of the abstract data type (ADT) that underlies the object model. This notion makes it possible to describe data types of any nature. Describing an abstract data type includes specifying attributes, associations, invariants, and type operations. Type operations are described by function types. Associations can be specified by metaclasses of associations, which can describe associations of any complexity. The type hierarchy is specified by the subtype relation. SYNTHESIS includes a set of basic types.

Classes in SYNTHESIS represent sets of homogeneous objects from the subject area. Every object in this set is an instance of the class. An extensional is associated with every class that contains all its instances. Classes, considered as objects, can be typed and their interfaces may be defined. Classes belong to a hierarchy based on the generalization/specialization relation. There are predefined classes in SYNTHESIS. For example, all classes, types, and ontological concepts defined in a given resource are instances of metaclasses named `class`, `type`, and `concept`, respectively. Belonging to such metaclass determines the kind of particular objects.

The following operations on types are defined in SYNTHESIS [7]. The unary operation `reduct` defines the subset of this type of specifications as its supertype. The `meet` operation yields the intersection of two types, including the semantically common part of the specifications of the operand types. The `join` operation yields the union of two types, including the union of the specifications of the operand types. These operations are denoted as follows.

- the reduction of the type T (`reduct`) is denoted as $\sim T$;
- the intersection of the types T_1 and T_2 (`meet`) is denoted as $T_1 \sqcap T_2$;
- the union of the types T_1 and T_2 (`join`) is denoted as $T_1 \sqcup T_2$.

Unstructured data, as any other kind of information, are represented in SYNTHESIS as frames. In the general case, a frame is an autonomous self-defining entity that makes it possible to describe both unstructured and semistructured data. Semistructured data may include structured and unstructured components. To describe the structured component, types are defined. The unstructured

component is represented by slots; the types of slot data are defined dynamically.

A scheme is the unit used for describing a resource in SYNTHESIS. A scheme can describe one or more modules. In particular, for sites or libraries, the scheme describes a module containing types and classes of site specifications or requirement specifications and the corresponding submodule containing ontological definitions of the subject area.

2 Example

Consider an example of designing a personalized digital library containing information on patents registered in the US and Canada. The requirement specification is given in SYNTHESIS by an object model using the scheme `PatentLibrary`. The scheme describes the type `Patent` and the corresponding class `patent` designed to contain objects of this type.

```
{ PatentLibrary;
  in : schema;
  type:
  { Patent;
    in: type;
    title: string;
      metaslot
      obl: invariant,
        {{obligatory}}
      end
    inventors: string;
      metaslot
      obl: invariant,
        {{obligatory}}
      end
    category: string;
    country: string;
    regDate: string;
    abstract: string;
    claims:
      { union;
        type_of_label: integer;
        1: string;
        2: { sequence;
             type_of_element: string}
      };
    descr: string;
  };
  class_specification:
  { patent;
    in: class;
    instance_section: Patent
  }
}
```

The attribute `title` in specifications of the type `Patent` is the title of the patent; the attribute `inventors` is a string with the list of authors; `category` is the category of the patent according to the international classification; `country` is the name of the country that has the priority of using the invention; `regDate` is the date of the patent registration; `abstract` is the annotation of the invention; `claims` is the list of novelties in the invention or a description of the invention subject; and `descr` is a detailed description of the invention.

The library is organized over two digital collections, web-sites containing information on patents registered in the US [11] and Canada [12]. These sites contain textual information (annotation, patent novelties, information about the authors, images of the patent originals, and other data). The schemes of these web-sites are very different from each other. They are obtained as a result of analyzing documents included in the collection and are presented here in the context of the ADM model in the Araneus data definition language [9].

```
SCHEME CanadaScheme
  PAGE-SCHEME CanadaPatentPage
    PatentNumber: TEXT;
    Title: TEXT;
```

```
    ToImages: LINK-TO ImgPage;
    Inventors: TEXT;
    Owners: TEXT;
    FilingDate: TEXT;
    CanadClass: TEXT;
    InterClass: TEXT;
    PriorCountry: TEXT;
    Abstract: TEXT;
    ToClaims: LINK-TO ClaimsPage;
  END
  PAGE-SCHEME ClaimsPage
    PatentNumber: TEXT;
    Title: TEXT;
    C1aims:
      TEXT
      UNION
      LIST-OF (Claim: TEXT);
      OPTIONAL;
  END
  PAGE-SCHEME ImgPage
    ImgList: LIST-OF (Img: IMAGE;);
  END
END-SCHEME

SCHEME USAScheme
  PAGE-SCHEME USAPatentPage
    PubNumber: TEXT;
    Abstract: TEXT OPTIONAL;
    Title: TEXT;
    Inventors: TEXT;
    Assignee: TEXT OPTIONAL;
    Filed: TEXT;
    USClass: TEXT;
    INClass: TEXT;
    Claims: TEXT OPTIONAL;
    Descr: TEXT OPTIONAL;
  END
END-SCHEME
```

In Araneus, scheme is defined consisting of the types of pages that occur on the site. The structure of pages is specified by attributes. The attributes may include unstructured data (`TEXT`), images (`IMAGE`), references (`LINK-TO`), lists (`LIST-OF`), alternatives in the structure (`UNION`), and others. Optional attributes are specified as `OPTIONAL`. The schemes shown above are slightly abridged. The real pages of these sites contain more information on the patents. After the site schemes are described, they must be mapped from the ADM into the canonical SYNTHESIS model.

Specifications of information resources are described as the schemes `CanadaScheme` and `USAScheme`; they define types that correspond to the site page specifications and classes corresponding to sets of such pages. Scheme attributes describing references to other pages are defined as associations. In our case, a single kind of association is sufficient, namely, the kind defined by the association metaclass `one_to_one`.

```
{ one_to_one;
  in: metaclass, association;
  inverse: i_one_one;
  instance_section:
  { association_type:
    {{0, 1}, {1, 1}}
  }
}
```

Below are the specifications of the site schemes in SYNTHESIS.

```
{ CanadaScheme;
  in: schema;
  type:
  { CPatPage;
    in: type;
    PatentNumber: string;
      metaslot
      Obl : invariant,
        {{obligatory}}
      end
    Title: string;
    ToImages: ImgPage;
      metaslot
      in: one_to_one;
      end
```

```
    Inventors: string;
    Owners: string;
    FilingDate: string;
    CanadClass: string;
    InterClass: string;
    PriorCountry: string;
    Abstract: string;
    ToClaims: ClaimsPage;
      metaslot
      in: one_to_one;
      end
  },
  { ClaimsPage;
    in: type;
    PatentNumber: string;
    Title: string;
    Claims:
      { union;
        type_of_label: integer;
        1: string;
        2: {sequence;
            type_of_element: string}
  },
  { ImgPage;
    in: type;
    ImgList:
      { sequence;
        type_of_element: Image}
  };
  class_specification:
  { cPatPage;
    in: class;
    instance_section: CPatPage
  },
  { claimsPage;
    in: class;
    instance_section: ClaimsPage
  },
  { imgPage;
    in: class;
    instance_section: ImgPage
  }
}
{ USAScheme;
  in: schema;
  type:
  { USPatPage;
    in: type;
    PubNumber: string;
    Abstract: string;
    Title: string;
    Inventors: string;
    Assignee: string;
    Filed: string;
    USClass: string;
    INClass: string;
    Claims: string;
    Descr: string
  };
  class_specification:
  { usPatPage;
    in: class;
    instance_section: USPatPage
  }
}
```

The description of the scheme `CanadaScheme` includes the list `ImgList`. This list of images is represented by the attribute that has a sequence of images as the type of its values. `Image` is an abstract data type that must be defined (this definition is not included in the example). In the specifications above, metaslots with definiteness invariants must be defined for all obligatory attributes occurring in the descriptions given in Araneus (as it done for the attribute `PatentNumber` in type `CPatPage`). These invariants show that the corresponding attributes cannot take the empty value (`none`). For the parts of pages that admit alternative structure elements, the union type is used (e.g., the attribute `Claims` in `ClaimsPage`).

Local ontological modules, which describe contexts of the subject areas of two collections and the digital library, must also be specified within the schemes described above as submodules of the modules containing descriptions of their type and class.

The specifications obtained become a part of a large repository containing scheme descriptions of many sites. This repository is nec-

essary for finding Web collections that could be used as components of concrete digital libraries.

3 Specification of Ontological Concepts of Site Subject Areas

Specifications of local digital collections and specifications of requirements must be associated with ontological contexts containing concepts of the corresponding subject areas. Ontological concepts are described by means of the canonical model of the SYNTHESIS language. It should be stressed that the context of the shared ontology must be described by specifications in the canonical model. For this purpose, for the most popular models for representation of ontologies mappings to the canonical SYNTHESIS model are developed.

Both the shared ontology and local ontological concepts have their verbal definitions and descriptor lists. Verbal definitions are similar to word definitions in an explanatory dictionary. Descriptor lists included in the specifications of ontology concepts are built on the basis of the meaningful words in the concept's verbal definition. Tools for lexical and morphological analysis are used in the process of building descriptor lists. Normalized words or word stems can be used as descriptors. Concept descriptors are necessary for establishing preliminary relation to other concepts lying outside the given ontological context.

Generalization/specialization (concept/subconcept relations) and positive relations (synonymies) may be defined between ontological concepts. These relations can be fuzzy; i. e., they can be characterized by degree (force) in the range between 0.0 and 1.0. If the value of relation is not defined explicitly it is implied to be equal 1.0. Concepts are also characterized by attributes, associations, and logical constraints. If the meaning of an element of the specification of a collection scheme or library requirements scheme corresponds to a concept included in the ontological context related to this scheme, this element becomes an instance of the class corresponding to this concept.

To compare site and library schemes on the basis of the ontological information available, it is required to reduce local ontological contexts of the collections and of the library to the same ontological context. For this purpose, local ontological contexts of the collections and of the library are mapped into the shared ontology concepts [3]. At the first stage, relations between concepts of different contexts are established by calculating the correlation coefficients between concepts on the basis of the verbal definitions. Then, a deeper and more accurate integration may be performed (if necessary) with regard to the internal structure of the concepts.

3.1 Mapping Local Contexts into the Shared Ontology

Integration of local contexts of ontological descriptions of sites and specifications of requirements is based on the mapping into a shared ontology. Integration at the level of verbal descriptions is based on the analysis of concept descriptors performed with the aim of mapping concepts of one ontological context into the other. For this purpose, the degree of relation between concepts of two ontological contexts is calculated using the vector-space approach [10].

The analysis of a descriptor begins with the estimate of its significance in the definition of particular concepts. Every descriptor is assigned a weight that takes into account the frequency of its occurrence in the definition of a particular concept and the number of con-

cepts in the context whose definitions include this descriptor. The more the frequency and the less the number of concepts defined with the help of this descriptor the more its significance and, thus, its weight. Let X and Y be concepts of different ontological contexts (local context and shared ontology). Let V_X and V_Y be the vectors consisting of descriptors that define the corresponding concepts in X and Y. For V_X and V_Y vectors C_X and C_Y are generated that contain lists of weights W_{Xk} and W_{Yk} for every descriptor k that participates in the definition of X and Y, respectively. The weights are calculated by the following empirical formulas [10]:

$$W_{Xk} = \frac{(1 + \frac{f_{Xk}}{f_{max}}) \cdot \log \frac{N}{n_k}}{\sqrt{\sum_{i \in V_X} ((1 + \frac{f_{Xi}}{f_{max}}) \cdot \log \frac{N}{n_i})^2}} \quad (1)$$

$$W_{Yk} = \frac{f_{Yk} \log \frac{N}{n_k}}{\sqrt{\sum_{i \in V_Y} (f_{Yi} \log \frac{N}{n_i})^2}} \quad (2)$$

where f_{Xk} and f_{Yk} are the frequencies of the occurrence of the descriptor k in V_X and V_Y respectively, f_{max} is the maximal frequency of descriptors in V_X or V_Y, N is the total number of concepts in the shared ontology, and n_k is the number of concepts in the shared ontology whose vector V_Y includes the descriptor k. The first factor in the product increases the significance of descriptors that are frequently mentioned in the definition. The second factor increases the significance of descriptors that occur in a lesser number of vectors V_Y. Frequencies in V_X are reduced to the interval [0.5,1.0], since every descriptor of a local ontology concept is important for finding corresponding concepts in the shared ontology.

The weights W_{Xk} and W_{Yk} are normalized to eliminate the dependence on the difference in the length of vectors for different X and Y. If dictionaries or thesauruses containing weight coefficients of words are available, other approaches to determining descriptor weights can be used.

The functions for estimating the correlation between ontological concepts are defined as follows [10, 3]:

$$sim(X, Y) = \frac{\sum_{k=1}^{t} (W_{Xk} \cdot W_{Yk})}{\sqrt{\sum_{k=1}^{t} (W_{Xk})^2 \cdot \sum_{k=1}^{t} (W_{Yk})^2}} \quad (3)$$

$$r(X, Y) = \frac{\sum_{k=1}^{t} \min(W_{Xk}, W_{Yk})}{\sqrt{\sum_{k=1}^{t} (W_{Xk})^2}} \quad (4)$$

$$r(Y, X) = \frac{\sum_{k=1}^{t} \min(W_{Xk}, W_{Yk})}{\sqrt{\sum_{k=1}^{t} (W_{Yk})^2}} \quad (5)$$

The range of values of the function $sim(X, Y)$ is the real interval [0.0,1.0]. A value of 0.0 means that the concepts are not related to each other, and a value of 1.0 means that the concepts are identical (have identical lists of descriptors). The concept X is considered correlating (similar) to the concept Y if $sim(X, Y)$ is greater than a certain threshold value ℓ; in this case, the value of the function is considered a measure of similarity (force of the relation).

The functions $r(X, Y)$ and $r(Y, X)$ are used to find possible relations of the kind concept/subconcept between different contexts. For these functions to give correct results, the weights of descriptors must be normalized. The above method for weight calculation satisfies this requirement. If $r(X, Y)$ and $r(Y, X)$

are less than a certain threshold value ℓ, the concepts X and Y are not related to each other. If both the values of $r(X,Y)$ and $r(Y,X)$ are greater than ℓ, the concepts X and Y are positively associated with each other, and the correlation coefficient (force of the relation) is the minimum of these values. If $r(X,Y)$ is greater than ℓ while $r(Y,X)$ is less, then X is a superconcept of Y, i. e., the generalization association is established between X and Y. In this case, the correlation coefficient (force of the relation) is equal to $r(X,Y)$. On the other hand, if $r(X,Y)$ is less than ℓ and $r(Y,X)$ is greater, then X is a subconcept of Y, i. e., the specialization association is established between X and Y. In this case, the correlation coefficient is equal to $r(Y,X)$. If necessary, the results of the automatic mapping of one context into another can later be refined manually by an expert.

At this stage, we do not consider the internal structure of concepts and the corresponding logical constraints. If the process of integrating ontological contexts is performed only on the basis of verbal descriptions, then the subsequent integration of data relevant to the context concepts can use only the concepts themselves (without regard to their internal structure), positive associations and the generalization hierarchy of the concepts. The integration of the internal structure of the concepts of ontological contexts suggests that concepts be considered as types. However, the specifics of internal structure integration is beyond the scope of this paper.

3.2 Ontological Integration of Schemes

After the local ontology contexts have been mapped into the shared ontology, we pass to the ontological integration of schemes. The main purpose of this stage is to detect (among the descriptions of information collections) types, classes, and their fragments that are relevant to the specifications of requirements of the digital library being developed. The problem is in relating the ontologically relevant specification elements of the library with the elements of the web-site specifications.

Three types of ontological contexts (modules) are involved in the integration process.

- The application ontology module (AOM) contains specifications of the ontological context of the personalized digital library and is connected to the specification module of requirements of the library.
- The resource ontology module (ROM) contains ontological specifications of the subject area of a concrete collection of information and is connected to the module of the object model of the corresponding web-site's scheme specifications.
- The common ontology module (COM) contains the shared ontology of the particular subject area similar to that for which the library is developed. COM helps to relate the concepts of the specifications of requirements with the site schemes concepts.

The results of the integration stage of the local contexts (AOM and ROM) with the shared ontology (COM) with regard to verbal definitions are presented as positive associations and concept/subconcept associations between the local application and resource contexts with the shared ontology of the subject area. To perform the preliminary element integration of the specifications of requirements with specifications of the available resources, it is required to find relations between AOM and ROM contexts. These relations can be established on the basis of internal generalization associations and internal positive relations in COM in

conjunction with the established intercontext relations that join the local ontology concepts with the concepts in COM.

The concepts in ROM that have positive or specialization associations with a concept in AOM (i. e., those that are synonymous with the AOM concept or are its subconcepts) can be related to the AOM concept in question. To reveal such relations, the correspondence paths of AOM and ROM concepts must be analyzed and the concept graph must be complemented with the missing relations.

The search for new relations is performed with the help of the algorithm described in [4] (this algorithm uses the transitivity property of relations). Two sequential positive relations give a new positive relation characterized by the force equal to the product of the forces of the given relations. If a sequence of two relations includes a specialization relation, then the force of the resultant relation equals the minimum of the two given relations, and the kind of this relation depends on the kinds of the relations in the path. If the path contains at least one positive relation, then the resultant relation will be positive as well. If the path consists entirely of specialization relations, then the resultant relation will be a specialization one. In order to calculate the force of a path that includes relations of various kinds, it is necessary to find, first of all, the forces of its subpaths consisting only of synonymy relations; then, the forces of fragments that include generalization/specialization relations can be calculated as well. The purpose of the algorithm is to find the maximum value of the relation force between two given concepts. Relations with forces not exceeding the threshold value ℓ are neglected. The same value ℓ that was used at the stage of the mapping of ontological contexts at the verbal level may be used as the threshold.

The results of this algorithm (positive and specialization relations between the concepts of local ontological contexts) are used to find correspondences between the specification elements. For this purpose, the concept of the weak ontological relevance of specification elements is introduced.

Definition 1. The element I_r of an information resource specification is called ontologically weakly relevant to the element I_s of the specifications of requirements of the same kind (type, class, function, attribute, and so on) if I_r is related to a concept C_r, I_s is related to a concept C_s, and there exists a positive relation between C_r and C_s, or C_r is a subconcept of C_s. This can be written as

$$\begin{aligned} &weak\_relevance(I_r,I_s) \iff (\exists C_r,C_s : \\ &\quad inst\_of(I_r,C_r) \wedge inst\_of(I_s,C_s) \wedge \\ &\quad \wedge(positive(C_r,C_s) \vee subconcept(C_r,C_s))) \end{aligned}$$

If a closer ontological integration of the website scheme specifications is required, after structure integration of ontological contexts the notion of the tight ontological relevance of specification elements can be used. This notion is based on the membership of elements in the same common classes of concepts of the shared ontology, where the local contexts are integrated, or in classes of their subconcepts.

Definition 2. The element I_r of an information resource specification is called ontologically tightly relevant to the element I_s of the specifications of requirements of the same kind (type, class, function, attribute, and so on) if I_r is ontologically weakly relevant to I_s, and I_r is an instance of at least one ontological concept C_r that is a specialization (subconcept) of an ontological concept C_s that has I_s as its instance (for this specialization, if the types of the concept instances are specified, the type of instances C_r must be a subtype of the type of the instances C_s), or I_r and I_s must belong

to the same ontological concept C. The condition of the tight ontological relevance can be written as

$$
\begin{aligned}
&tight\_relevance(I_r,I_s) \iff \\
&\quad weak\_relevance(I_r,I_s) \wedge \\
&\quad \wedge((\exists C : inst\_of(I_r,C) \wedge inst\_of(I_s,C)) \vee \\
&\quad \vee(\exists C_r,C_s : subconcept(C_r,C_s) \wedge \\
&\quad \wedge inst\_of(I_r,C_r) \wedge inst\_of(I_s,C_s)))
\end{aligned}
$$

In the subsequent development of the library, the results of the ontological integration of schemes are used (before composing its general scheme) to preliminarily relating elements of collection scheme specifications to the specification elements of requirements that are instances of classes of the ontological concepts defined.

4 Designing Concretizations and Composition

The method used to identify fragments of local specifications that refine fragments of the specifications of requirements is based on the principles presented in [7]. The process of fragment identification and their composition involves several steps.

1. The information obtained at the stage of the integration of ontological contexts is used to select (among the ontologically relevant types and classes of collection schemes) fragments that can be used to refine the corresponding fragments of the library being developed. The fragments of type specifications are generated using the type operation `reduct`. Reducts are considered as patterns for refining specifications. In the process of refining, various conflicts and mismatches in the specification fragments of the collections and the library are resolved.

2. After the reducts of types have been identified, the type operations `meet` and `join` are used to build compositions of reducts of the resource types involved; these compositions refine the requirement specifications.

3. Views are constructed over the classes corresponding to the types of the specifications of requirements. The views are compositions of classes corresponding to the collection specification types involved.

4. Validation of the concretization constructed can be performed using formal proving methods. If the refinement is correct, the concrete types and views become, respectively, subtypes of types and subclasses of classes of the requirements.

We now return to our example. Assume that we have already obtained a set of scheme elements of Canada and USA patent collections that are relevant to the elements of the same kind in the specifications of requirements. From this set, we select classes, types, and their elements that will be used to implement the specifications of requirements.

The type `Patent` and the class `patent` in the specifications of requirements correspond to the type `CPatPage` and the class `cPatPage` of the Canadian site. Almost all attributes of `Patent` found a corresponding element among the attributes of `CPatPage`, except the attributes `descr` and `claims`. The attribute `claims` corresponds to the attribute `Claims` of another type of pages (`ClaimsPage`) of the same scheme; a reference to this attribute is included in `CPatPage`. The attribute `descr` cannot be implemented on the basis of the elements of the given site scheme. Let us write the pairs of elements of the schemes `PatentLibrary` and `CanadaScheme` that correspond to each other.

```
PatentIibrary    CanadaScheme

Patent           CPatPage
Patent.title     CPatPage.Title
Patent.inventors CPatPage.Inventors
Patent.categoryc CPatPage.InterClass
Patent.country   CPatPage.PriorCountry
Patent.regDate   CPatPage.FilingDate
Patent.abstract  CPatPage.Abstract
Patent.descr     -
Patent.claims    ClaimsPage.Claims
patent           cPatPage
```

For the American patent site, the type `Patent` and the class `patent` in the specifications of requirements correspond to the type `USPatPage` and the class `usPatPage` of the site specification. Only one element, namely, `country`, cannot be implemented. The pairs of the corresponding elements in the schemes `PatentLibrary` and `USAScheme` are as follows.

```
PatentLibrary    USAScheme

Patent           USPatPage
Patent.title     USPatPage.Title
Patent.inventors USPaCPage.Inventors
Patent.category  USPatPage.INClass
Patent.country   -
Patent.regDate   USPatPage.Filed
Patent.abstract  USPatPage.Abstract
Patent.claims    USPatPage.Claims
Patent.descr     USPatPage.Descr
patent           usPatPage
```

Resolving structure conflicts between the specifications of resources and requirements is performed in the process of comparing ontologically relevant paths. Minimal paths must be chosen; i. e., those that do not contain relevant subpaths. The search for relevant paths is done by applying special rules. In our example, one such rule was used.

Rule 1. Paths are relevant if the types that are end nodes of the path are ontologically relevant, the path on the specifications of requirements side consists of one single-valued attribute, and the path on the resource side consists of single-valued attributes and generalization relations, and ends in a single-valued attribute.

The ontological relevance and resolving structure conflicts provide a basis for the identification of common reducts of the types of requirements and collection types. They determine the fragment of the collection type that can be used to concretize the type of the requirements. For the common reducts, concretizing reducts are constructed in which values of the elements of requirements are brought into exact correspondence with the values of the elements of resource specifications. Thus, it becomes possible to get rid of structural conflicts and other mismatches between the specifications of the integrated types. In our example, the greater part of attributes of the `Patent` type are included in the common reduct; they receive their values directly from the corresponding attributes of types `CPatPage` and `USPatPage` depending on the collection. The exception is provided by the attributes `descr` and `country` that have no implementations for different schemes, and the attribute `claims` because of a structure conflict. For the latter attribute, a function can be constructed in the concretizing reduct that resolves the structure conflict between the `PatentLibrary` and `CanadaScheme` schemes in accordance with the relevant path found by applying the rule mentioned above:

```
f_claims: { in: function;
  params: { +c/CPatPage,
            -return/Patent.claims};
  {{ return = c.ToClaims.Claims}}
}
```

The attribute `descr` is not implemented in the Canadian site, thus for objects taken from this resource, it must be initialized by the value `none`. For objects taken from the American site, the attribute `country` can be initialized by the value `'United States'`. We also note that the type of the attribute in the specification of `USAScheme` refines the type of the union of the attribute `Claims` in the specifications of requirements; thus, no mismatch in the types of attributes occurs in this case.

Development of concretizing types implementing the corresponding types of the requirements is based on the composition of types of the collection specifications. The composition is constructed by applying the `meet` and `join` operations. With regard to the above analysis of relevant paths and reducts, a formula can be obtained that makes it possible to construct a type refining the type in the requirement specification simultaneously avoiding the conflicts detected. In our example, the `CTPatent` composition for the `Patent` type consists of the join of the `CPatPage` and `ClaimsPage` types intersected with the `USPatPage` type:

$$CanPage = CPatPage \sqcup ClaimsPage$$
$$CTPatent \approx CanPage \sqcap USPatPage$$

After performing these operations, the `CTPatent` type lacks the `descr` and `country` attributes that are required for the concretization. However, we already know how to initialize them by default values for those collections where they are not implemented. Thus, the concretizing type is supplemented by these attributes. The resultant `CTPatent` type becomes a subtype of the `Patent` type from the requirement specification.

The development process completes by creating views over the classes of the resource collections that implement classes of the specifications of requirements and become subclasses of the corresponding classes. For the case of the composition of the classes of two sites in the patent digital library, the `vPatent` view is defined over the `cPatPage`, `claimsPage`, and `usPatPage` resource classes.

In our example, the `join` operation on types corresponds to the join of the corresponding classes of objects; the `meet` operation corresponds to the union of classes. Thus, to obtain the class corresponding to the `join` operation of the `CPatPage` and `ClaimsPage` types (its result is denoted as `CanPage` in the view) the join of the `cPatPage` and `claimsPage` classes is formed. The view corresponding to the `meet` operation on the `CanPage` and `USPatPage` types is the union of the class obtained above with the `usPatPage` class. The `vPatent` view contains the function `prop_view` describing the rules for the formation of library class instances and mapping the states of the objects of the resource types into the library type.

```
{ vPatent;
  in: class;
    metaslot
    prop_view: { in: function;
      params: -returns/vPatent as_class;
      enforcement: on_access;
    {{
      ( { cp/CanPage
          [ title/Title,
            inventors/Inventors,
            regDate/FilingDate,
            category/InterClass,
            abstract/Abstract,
            claims/Claims,
            country/PriorCountry ] |
          cPatPage (cp/CanPage
            [CPatPage]) &
          claimsPage (cp/CanPage
            [ClaimsPage])} &
        descr(cp)==none ) |
```

```
      ( ( usPatPage(u/USPatPage
          [ title/Title,
            inventors/Inventors,
            category/INClass ]) &^
          cPatPage(u/CPatPage
          [ title/Title,
            inventors/Inventors,
            category/InterClass ]) ) &
        ( usPatPage(u/USPatPage
          [ title/Title,
            inventors/Inventors,
            category/INClass,
            regDate/Filed,
            abstract/Abstract,
            claims/Claims,
            descr/Descr ]) &
          country(u)=='United States') )
    }}
    end
  superclass: patent;
  instance_section: CTPatent
}
```

Classes of different collections can contain, as their instances, some objects corresponding to the same entities of the real world. Such correspondences must be found in the process of building the library classes. This can be implemented by supplementing the predicate that forms the view. For objects that are instances of classes of different collections, identification rules are defined, depending on their internal state. In our example, the following rule is used. If, for two patent objects found in `cPatPage` and `usPatPeige`, the titles, inventors, and categories in the international classification coincide, only the object from the class `cPatPage` is included in the class `vPatent`.

5 Conclusion

In this paper, we presented a method for designing personalized digital libraries over semistructured information resources on the Web using the specifications of user's requirements. The method is demonstrated by creating a library over real digital collections available on web-sites. The stages of the library development are described, starting with the formulation of the requirements to composing real information resources in the digital library. Similar methods and designing techniques (more precisely, a development of these methods) will be used for development personalized digital libraries in project no. 98-07-91061 supported by the Russian Foundation for Basic Research.

Acknowledgements

This work was supported by the Russian Foundation for Basic Research, project no. 98-07-91061, and by the International Association for the Promotion of Cooperation with Scientists from the Newly Independent States of the Former Soviet Union (INTAS), grant no. 97-l l09.

References

[1] P. Atzeni, G. Mecca , P. Merialdo. *Semistructured and Structured Data in the Web*: *Going Back and Forth.* In Sigmod Record, Special Issue on the Workshop on the Management of Semistructured Data, 1997

[2] D. O. Briukhov, L. A. Kalinichenko. *Component-Based Information Systems Development Tool Supporting the SYNTHESIS Design Method.* Proc. of the East European Symposium on Advances in Databases and Information Systems, Poland, Lect. Notes Comput. Sci., 1998, no. 1475

[3] D. O. Briukhov, S. S. Shumilov. *Ontology Specification and Integration Facilities in a Semantic Interoperation Framework.* Proc of the International Workshop on Advances in Databases and Information Systems (ADBIS'95), 1995

[4] P. Fankhauser, E. J. Neuhold. *Knowledge Based Integration of Heterogeneous Databases.* Integrated Publication and Information Systems Institute (GMD-IPSI), Darmstadt, 1993

[5] S. Grumbach, G. Mecca. *In Search of the Lost Schema.* In Proc. of International Conference on Database Theory (ICDT'99), 1999

[6] *SINTEZ*: *yazyk opredeleniya, proyektirovaniya i programmirovaniya interoperabelnykh sred neodnorodnykh informatsionnykh resursov (SYNTHESIS*: *the Language for Desription, Design and Programming of the Interoperable Environments of Heterogeneous Information Resources).* Institute for Problems of Informatics, Russian Academy of Sciences, Moscow, 1993

[7] L. A. Kalinichenko. *Compositional Specification Calculus for Information Systems Development.* In Proc. of the East-West Symposium on Advances in Databases and Information Systems (ADBIS'99), Maribor, Slovenia, Lect. Notes Comput. Sci., 1999

[8] O. Machulsky, M. Osipov, L. A. Kalinichenko. *Otobrazheriye modeli dannykh XML v obyektnuyu model yazyka SINTEZ (Mapping the XML Data Model into the Object Model of the SYNTHESIS Language).* Trudy Pervoi Vserossiyskoy nauchnoy konferentsii "Elektronnye biblioteki: perspektivnye metody i tekhnologii, elektronnye kollektsii" (Proc. 1st All-Russia Sci. Conf. Digital Libraries: Promising Methods and Technologies, Digital Collections), St. Petersburg, 1999

[9] G. Mecca, P. Merialdo, P. Atzeni, V. Crescenzi. *The Araneus Guide to Web-Site Development,* Araneus Project Working Report, AWR-1-99 (version 1.0), 1999

[10] G. Salton, C. Buckley. *Term-Weighting Approaches in Automatic Text Retrieval.* Readings in Information Retrieval, K. S. Jones and P. Willett, Kaufmann, 1997

[11] *United States Patent.* [http://164.195.100.11/netahtml/search-bool.html]

[12] *Canadian Patent Database.* [http://Patents1.ic.gc.ca/intro-e.html]

[13] *Extensible Markup Language (XML) 1.0. W3C Recommendation.* [http://www.w3.org/TR/1998/REC-xml/]