Transforming *ad hoc* product data into canonical product representation

Jingzhi Guo* and Chengzheng Sun

School of Computing and Information Technology
Griffith University, Nathan, QLD 4111, Australia
E-mail: J.Guo@cit.gu.edu.au jingzhiguo@yahoo.com
E-mail: C.Sun@cit.gu.edu.au
*Corresponding author

Abstract: Interoperation between business partners is extremely important. However, most product data produced in many firms are considered as *ad hoc* and are not qualified for interoperation. To solve this problem, this paper has proposed a concept-centric definition transformation approach, which transforms irregular local product definitions from different data sources to a set of canonical local product representations. These canonical representations are thus able to translate these common product representations to what the public can understand, thus, achieving business interoperation between different semantic communities.

Keywords: product interoperation; e-business; personalisation; *ad hoc* product data; canonical product representation; context; integration; product standard.

Reference to this paper should be made as follows: Guo, J. and Sun, C. (2005) 'Transforming *ad hoc* product data into canonical product representation', *Int. J. Internet and Enterprise Management*, Vol. 3, No. 2, pp.117–138.

Biographical notes: Jingzhi Guo is E-commerce Researcher. He received his BEcon in International Business Management from University of International Business and Economics (UIBE), China; MSc in Computation from University of Manchester Institute of Science and Technology (UMIST), UK; and PhD from Griffith University, Australia. His research interests are in the areas of product data integration, business document integration, process reengineering, supply chain management and global electronic markets. He has published articles in journals and conference papers such as *Electronic Markets*, *ACM Symposium on Document Engineering* and *IEEE Conference on E-Commerce Technology*.

Chengzheng Sun is Professor for Internet Computing at Griffith University, Australia, at the School of Computing and Information Technology. His research focuses on the collaborative systems, internet computing, distributed systems, parallel systems and logical programming. He is the inventor of several web-based collaborative editing systems such as REDUCE. He has published articles in journals and conference papers such as *ACM Transactions on Computer-Human Interaction*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Multimedia*, *IEEE Internet Computing and ACM Conference on CSCW*.

1 Introduction

Product Data Integration (PDI) is an essential issue for business-to-business interoperation (Fensel *et al*., 2001) in the areas of electronic product catalogue (Baron *et al*., 2000; Ginsburg *et al*., 1999; Guo and Sun, 2003b; Handschuh *et al*., 1997; Keller and Genesereth, 1996; Schulten *et al*., 2001; Segev *et al*., 1995; Stanoevska-Slabeva and Schmid, 2000), supply chain management (Ball *et al*., 2000; Christiaanse and Kumar 2000; Kumar, 2001; Omelayenko *et al*., 2002; Welty and Becerra-Fernandez, 2001; Wombacher *et al*., 2003), and semantic web services (Bergamaschi *et al*., 2002; Berner-Lee *et al*., 2001; Omelayenko, 2002). It deals with how to extract product data from rough sources that are heterogeneously represented in different 'semantic communities' (Robinson and Bannon, 1991), and how to map them together for semantics interoperation (Guo and Sun, 2003b). Central to this issue is how to capture the inconsistent semantics of heterogeneous product data, and how to transform them into a consistent semantic context model for semantics mapping (Guo and Sun, 2003c–d).

It is a challenge to present a consistent semantic context model to address product data integration problems. Currently, there are many types of heterogeneous product data representations located in different sources such as the types listed below:

Type 1 There are numerous heterogeneous *de facto* industrial product standards such as xCBL (www.xcbl.org), cXML (www.cxml.org) and ebXML (www.ebXML.org), which prevent business interoperations (Dogac and Cingil, 2001; Ng *et al*., 2000; Omelayenko and Fensel, 2001a; Shim *et al*., 2000).

Type 2 There are several heterogeneous international product classification standards such as UNSPSC (www.unspsc.org) and eCl@ss (www.eclass-online.com), which present different guidelines of classifying products (Schulten *et al*., 2001).

Type 3 There are millions of *ad hoc* enterprise-wide product data representations. These representations generally exist in the Small- and Medium-sized Enterprises (SMEs) that are financially and technically difficult to link with existing product standards because they may require too much reengineering work for PDI (Guo and Sun, 2003c–d).

The above types of heterogeneous product representations are the major obstacles of web-based business interoperation. There are considerable efforts toward integrating the above heterogeneous product data. Current researches generally focus on the standard integration of Types 1 and 2 (Bergamaschi *et al*., 2002; Dogac *et al*., 2002; Ng *et al*., 2000; Omelayenko and Fensel, 2001b; Omelayenko *et al*., 2002; Omelayenko, 2002; Schulten *et al*., 2001). Nevertheless, researches on how to integrate *ad hoc* product data are not sufficient, except those that had been done by Guo and Sun (2003a–d). The following case illustrates the issue of integrating *ad hoc* product data.

Consider that there are two SMEs, A and B, which have their own product catalogues, Catalogue A and B. Catalogue A simply represents the product data on Web page while Catalogue B represents it in relational tables. Assuming that both Catalogue A and B contain a refrigerator specification as shown in Figure 1 and 2, the refrigerator data of Catalogue A and B cannot interoperate with each other because Catalogue A and B cannot understand each other due to the following facts:

- Catalogue A is not machine-readable.
- Even if Catalogue A is machine-readable, Catalogue B cannot understand Catalogue A because their product attributes are differently defined. For example, Catalogue A cannot assure that 'silver' is the colour of the refrigerator or 'fab2az3' is the identifier of refrigerator.
- Catalogue A and B have very different attribute descriptions for both attributes and attribute values.

These issues make Catalogue A and B semantically different for mutual understanding and thus not interoperable.

Figure 1 Refrigerator Example 1

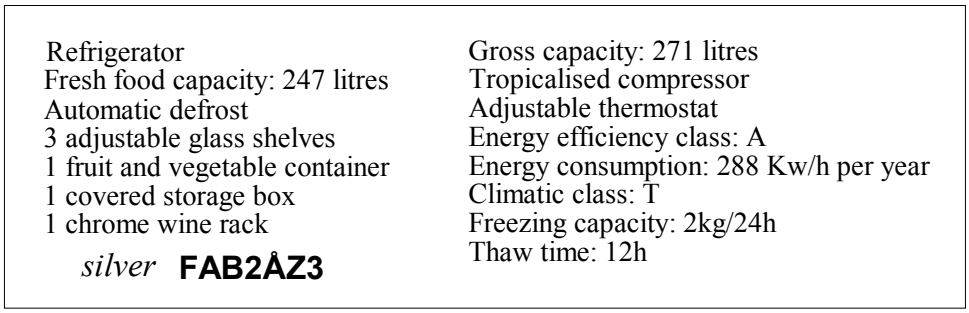


Figure 2 Refrigerator Example 2

ID	Name	Model	Dimension	Weight	Color	Description
5	Refrigerator	HTQ18JAAWW	d255	w132	white	frost free

ID	Width	Depth	Height
d255	29 5/8"	33 1/4"	66 3/4"

ID	Unit	Value
w132	lbs	238

These semantically different product catalogues of SMEs are *ad hoc* product data with several important characteristics: inconstant, small-scale, irregular, heterogeneous and numerous (Guo and Sun, 2003d). *Inconsistent* means the data source follows no standard vocabulary, *small-scale* refers to the small number of products contained in a product catalogue, *irregular* means that there are various types of product data sources such as Web pages and relational tables, *heterogeneous* refers to different languages and semantic encoding (*e.g.*, English, French or Chinese), and *numerous* means millions of product catalogues because there are millions of SMEs. These characteristics indicate that the problem of semantic inconsistency is severe. Reflecting on devising the PDI mechanism for global business-to-business interoperation between firms, three specific problems are detected relating to the semantic context representation and transformation. They are *context extraction*, *context mediation* and *context comparison* (Guo and Sun, 2003c).

Solving these problems is essential to integrating *ad hoc* product data. The work of Guo and Sun (2003c) has outlined an initial model, which articulates the context representations in a framework of irregular local product definitions, canonical local

product representations and public common product representations. A transformation process is introduced to transform various irregular local product definitions, such as data stored in different systems like XML files, relational tables and *ad hoc* web pages, into canonical local product representations. After the canonical local product representations are obtained, the consistent local semantic contexts are represented for mediation and comparison. This enables local product data to be interoperable within an enterprise-wide system. The public semantic contexts of *ad hoc* product data are achieved by mapping the canonical local product representations into common public product representations. When public semantic contexts are represented, different *ad hoc* product data located in various SMEs are able to interoperate by comparing the public semantic contexts.

The semantic context representation model (Guo and Sun, 2003c) is logical, but this model is a high-level framework that leaves several lower-level issues undiscussed. The research of Guo and Sun (2003d) has solved some of these issues: how to represent publicly understandable contexts, and how to transform and compare these contexts for product data exchange between *ad hoc* product data. Nevertheless, two issues remain unsolved, that is, how to model a process that will transform irregular local product definitions into canonical local product representations and how to connect the publicly understandable contexts by SMEs to industry-wide standards such as ebXML and UNSPSC.

This paper aims to solve one of the above issues by proposing a novel *concept-centric definition transformation approach*. This approach includes a LOCAL PRODUCT MAP and a *picker* object that are used to transform irregular local product definitions to canonical local product representations based on the *product representation model* (Guo and Sun, 2003d). The approach follows the design guidelines discussed in Guo and Sun (2003b). The issue of how to connect SMEs' popular terminology to industry-wide standards is beyond the scope of this paper, but we will give a brief discussion in Section 5.

Section 2 proposes a novel LOCAL PRODUCT MAP to model local concept generation processes. Section 3 discusses how the transformation process intends to capture *ad hoc* product data from various data sources. In Section 4, we briefly discuss the implementation by proposing a concept-centric catalogue architecture and a set of local XML Product Map documents. Section 5 briefly discusses the issues of why we need canonical product representation, what is the cost of concept mapping and how to interoperate with the existing industrial standard ebXML. Section 6 concludes the paper and outlines future works.

2 Local product map

We propose a novel LOCAL PRODUCT MAP in this section to model local concept generation processes. The model discusses how to transform *irregular local product concepts* into *canonical local product concepts* in Local Electronic Product Catalogue (LEPC) in the form of a four-tuple (product identifier, product annotation, product options, (attribute identifier, attribute annotation, attribute options, (attribute identifier, attribute annotation, attribute options, (…)))). This canonical local product representation (*locRep*) has the same structure as the common product concepts (*comRep*) situated in common electronic product catalogues (CEPC). This is a revision based on the work of Guo and Sun (2003d). We simply denote it as:

locRep := (IID, AN, OP, (IID, AN, OP, (…)))

where IID is the unique identifier of a concept (either a product or an attribute concept), AN is concept annotation and OP is the optional item that details the concept specification. Before we proceed to discuss the LOCAL PRODUCT MAP, we briefly introduce the techniques that transform one locRep to another through comRep. In the work of Guo and Sun (2003d), the interoperation between *locRep* and *comRep* is through the mapping of locIID and comIID. For example, given a refrigerator specification that are separately encoded in two LEPCs as LEPC1: (1.5, fridge, (1.5.1, prc)) and LEPC2: (338, réfrigérateur, (338.1, prix)) and a CEPC as (1.52.14.15.1, domestic refrigerator, (1.52.14.15.1.1, price)), by joining the operations discussed in Guo and Sun (2003d), two LEPCs could be joined in CEPC to obtain the maps as follows:

1 Map(locRep1, comRep) := {(1.5 , 1.52.14.15.1), (1.5.1, 1.52.14.15.1.1)}

2 Map(locRep2, comRep) := {(338, 1.52.14.15.1), (338.1, 1.52.14.15.1.1)}.

When LEPCs have been joined in CEPC, LEPC1 can communicate with LEPC2 simply by querying their locIIDs through CEPC. Since locIID and comIID are mapped in CEPC, 'fridge' can be understood as 'réfrigérateur' and 'prc' can be understood as 'prix'. In this sense, a semantic communication between two *ad hoc* SMEs is a process of checking unique IID mapping between two adjacent catalogues.

2.1 Basic local product map

Nevertheless, how can we obtain the canonical form of (1.5, fridge, (1.5.1, prc)) or (338, réfrigérateur, (338.1, prix)) from various SMEs to be mapped into comRep? The following sections resolve this issue by proposing a novel LOCAL PRODUCT MAP. Firstly, we introduce the term of *normalised concept* to understand the subtle issues in modelling.

2.1.1 Normalised concepts and subtle issues

Let us take Example 1 – refrigerator that is irregularly defined in an SME as shown in Figure 1. By analysing this irregular product definition against the *locRep* representation model, the refrigerator concept can be decomposed into many concepts in different levels such as:

- *Product level*: (refrigerator)
- *Attribute Level* 1: (*capacity*, feature, compressor, thermostat, energy efficiency class, energy consumption, climate class)
- *Attribute Level* 2 (only for capacity): (fresh food, gross, *freezing*)
- *Attribute Level* 3 (only for freezing): (2, kg, 1, *24h*)
- *Attribute Level* 4 (only for 24h): (24, *h*).

Obviously, each concept in each level is irregular. To make these concepts regular and canonical for interoperation, we normalise all decomposed concepts into 'normalised concepts' in the *locRep* form of (identifier, annotation, link) as follows:

- Level 1: $\text{Concept}_{\text{refrigerator}}$:= (fab2az3, refrigerator)
- Level 2: $\text{Concept}_{\text{capacity}}$:= (fab2az3.1, capacity)
- Level 3: $\text{Concept}_{\text{freezing}}$:= (fab2az3.1.3, freezing)
- Level 4: $\text{Concept}_{<24h>}$:= (fab2az3.1.3.3, ***24h***)
- Level 5: $\text{Concept}_{<h>}$:= (fab2az3.1.3.3.2, ***h***)

For the convenience of analysis, we define a *normalised concept* as a denotative concept of *locRep* in the form of (IID, annotation, options (link, structure, …)) that satisfies XPM Rule 2 presented in Guo and Sun (2003d), where each structure points to a set of lower level denotative concepts. Following this definition, we analyse the above irregular refrigerator example and find several subtle issues. First, some attribute concepts are ambiguous and implicit (c.f., they are similar to the semantic issues discussed in interoperable databases (Goh *et al*., 1994; 1999; Kashyap and Sheth, 1996)), that is, no enterprise-wide annotations are given to define these implicit concepts. For example, under $\text{AC}_{\text{freezing}}$, irregular concepts of '2, kg, 1, 24h' are not qualified as normalised concepts because these are not clear and may not be understood by others if we simply represent them in *locRep*. Second, to solve the first issue, we may use understandable annotations (meta concepts) such as 'scalar value, scalar name, unit, unit name' to replace '2, kg, 1, 24h'. Nevertheless, if we make such replacement, from where should these metaconcepts come? Where and how should these specific data be stored to reflect the precise irregular local product definitions? Third, if we compare this analysis against Example 2 in Figure 2, we further find more issues. There are many heterogeneous annotations or product identifiers which are denoted differently between two examples. Fourth, the number of levels of decomposition may be different for different local product definitions. Fifth, the number and semantics of different local concepts on the same level under the same parent concept may be different in three cases: identical in both number and semantics, identical for a few numbers in semantics, and disjoint for all numbers.

These subtle issues can be classified into three categories:

1 Issues for generating enterprise-wide generic terms that define concepts in the form of *locRep*, where data are separated from these generic terms. This is analogous to creating local metaconcepts for *locRep*.

2 Issues that allow heterogeneous terms for different data sources, semantically the same but expressed differently, to be communicated between heterogeneous legacy systems.

3 Issues to include particular product data that are only instances of concepts. These instances should not change the semantics of local metaconcepts.

2.1.2 Modelling basic local product map

We propose a novel basic LOCAL PRODUCT MAP in this subsection to solve Category 1 issue by forcing local product catalogue designers to provide the consistent normalised concepts in the form of identifier, annotation, options to generate normalised *locRep* concepts that are able to communicate with *comRep*.

We start the proposal from the *normalisation analysis*, which is a process of checking whether the irregular concepts from product definitions are consistent with *locRep* model, how many levels are involved, how many attributes are in each level that belong to the parent attributes or products, and how much new information should be added to create a consistent normalised concept. To facilitate the checking process, the LOCAL PRODUCT MAP models the relations between the fields of a concept so that we could prevent ambiguous and implicit instance data that affect the understanding within a firm and could generate local metaconcept correctly. The modelling process is:

First, generating a local metaconcept by defining a concept's *denotation* and *connotation* (Guo and Sun, 2003d): denotative concept defines a local metaconcept's uniqueness, scope and spatial information such as identifier, annotation and link. Connotative concept defines a local metaconcept's lower level concept structure consisting of a set of attribute concepts. For example, inside the brackets of Level 1 concept of Example 1 is the denotation, while all identified Level 2 concepts are connotations of Level 1 concept.

Second, setting a rule that a new local metaconcept can only be created by the first local product catalogue designer who generates the concept. We call this rule as *FISE* – 'first insert, second edit'. This rule is based on the assumption that enterprise-wide catalogue designers have the full knowledge and privilege of creating a correct local metaconcept. Designers who come later could only edit an existing concept by either appending *apposition concepts* (*i.e.*, semantically same but may be expressed differently) with the same concept identifier IID or change the concept structure by including more child-level concepts. This rule guarantees that the existing referenced concepts and instantiated concepts are not affected.

A basic LOCAL PRODUCT MAP devised in Figure 3 provides a structure of a metaconcept model, which guarantees that a local metaconcept is generated correctly according to *locRep* model. This model provides a canonical analysis, input and processing framework for LEPC designers to create the expected local metaconcepts to form *locRep*. Relations in the model are concept-centric. They force all normalised concepts to be transformed into *locRep* concepts in a rigid way.

First, a concept has a type, which is either a product type or an attribute type. A *product type* means a concept is defined on a product catalogue tree as a product node. An *attribute type* means a concept is defined as an attribute node on a product tree.

Second, a concept is assigned a unique local internal identifier when it is being generated. A concept's internal identifier locIID consists of two parts: local product identifier locPID and a local attribute concept identifier locAID. A locPID is the legacy product identifier such as 'fab2az3'. A locAID is defined as a vector concept on a *vector tree* (A_i^1, …, A_i^k) (for the details of vector concept, please refer to Guo and Sun (2003d)). A locIID can be take the following form:

$$LocIID := locPID \times locAID := locPID \times 1.i\ldots i$$

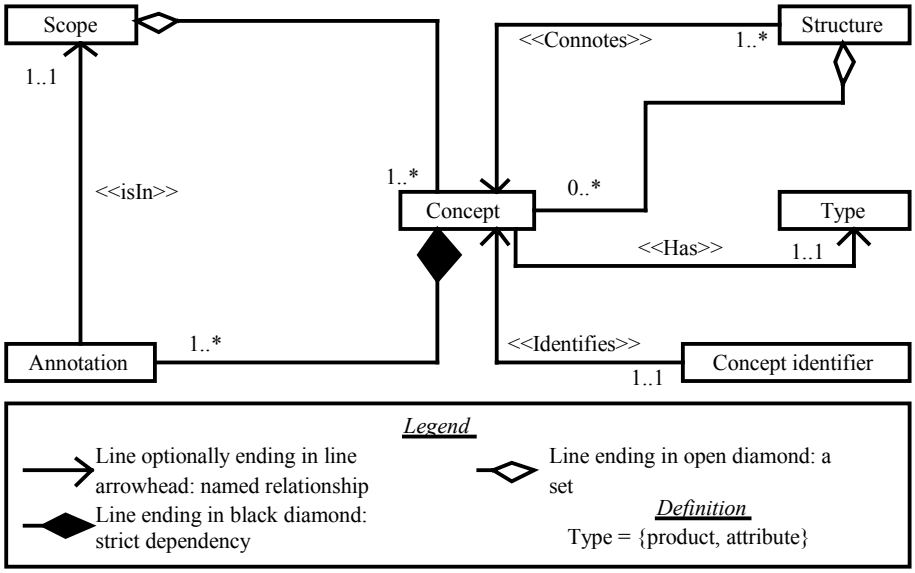
where i is the i in (A_i^1, …, A_i^k) of a vector tree. For simplicity, a locIID is represented as locIID<k, i> where k is the tree level and i is the position of sibling.

Third, a generated locIID is only validated after the concept is denoted by an annotation otherwise it is illegal and void. In this case, a concept is strictly dependent on an annotation following *FISE rule*.

Fourth, a concept may be denoted by many annotations in different scopes, each of which belongs to exactly a concept scope. This defines many concept scopes for the same concept. However, though annotations may have different scopes, they should be semantically equivalent with the annotation following *FISE rule*. For example, 'refrigerator' and 'réfrigérateur' belong to different language scopes, but they are semantically equivalent. In addition, each scope concept should only have one annotation as enterprise-wide reference.

Fifth, a concept is connoted by one-to-many structures, where each conveys a set of child concepts. Connotation is a process of appending child concepts. If a concept has no connotation, it is called a *leaf concept*.

Figure 3 Concept-centric basic product map



2.1.3 Local metaconcept generation process

To describe how systems precisely regulate the generation of local metaconcepts that create or evolve a local product catalogue in conformity with the basic LOCAL PRODUCT MAP, we provide a generic procedure to govern the generation process.

Local metaconcept generation procedure: when an irregular local product definition has been decomposed into a set of normalised concepts and is ready to be part of *locRep*, the following steps are executed:

1 *Browse*

Operations on LEPC (note: the initial LEPC contains only a catalogue root) determine concept locIID by finding the insert position of a new concept.

2 *Retrieve*

Operations on LEPC retrieve the IID-ed concept information to designer's user interface, *i.e.*, concept editor. If the locIID corresponding to the browsed annotation associates a type of *product*, then the retrieved local metaconcept is a locPID from a catalogue tree. If the locIID corresponding to the browsed annotation associates a type of *attribute*, then the retrieved local metaconcept is locPID*locAID from a product tree.

3 *Insert*

Operations insert the new local metaconcept into LEPC according to the retrieved locIID. The possible concept insert positions are:

- product sibling or product child if the retrieved concept is locPID
- attribute sibling or attribute child if the retrieved concept is locAID.

This procedure describes the overall governing regulations of how a basic LOCAL PRODUCT MAP could correctly generate enterprise-wide metaconcepts as a set of enterprise-wide *reference metaconcepts* for constructing a canonical LEPC against a set of normalised concepts input from the designers.

2.2 Extended local product map

Two issues about how to accept heterogeneous terms and how to include particular product data are not discussed in basic LOCAL PRODUCT MAP. These two issues are important in integrating heterogeneous *ad hoc* product data and integrating dynamic product data exchange. This subsection aims to solve these two problems.

2.2.1 Allowing heterogeneous concept expressions

In multidatabases, semantic conflicts arising from records are often resolved through comparing contexts against marketplace ontologies (Goh *et al*., 1994; 1999; Kashyap and Sheth, 1996). In this paper, we follow concept exchange approach by comparing a set of concepts (Guo and Sun, 2003b) composed in different departments to determine whether *ad hoc* product definitions that are heterogeneously expressed are semantically the same. Since marketplace product ontologies are generally not available to most *ad hoc* product data, these product data are often generated following no public rules/standards but only the preferences of LEPC designers in their local semantic communities (Guo and Sun, 2003c).

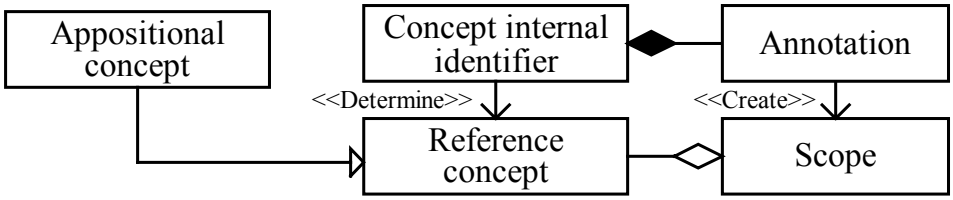
The need and possibility for including heterogeneous concept expressions vary in different firms:

- *Ad hoc* product data have their own characteristics even between departments of a firm.
- Maintaining *ad hoc* data is necessary because many firms have already designed their business processes based on the *ad hoc* product definitions. Changing product definitions means changing the existing business processes, which is not desirable.
- *Ad hoc* product data are often generated in SMEs or their respective departments where each has a small number of *ad hoc* product definitions – from several to tens of products.

These make it necessary and possible for firms (at least for SMEs) to manually map the heterogeneously expressed concepts onto a set of enterprise-wide metaconcepts by browsing an enterprise-wide reference LEPC.

Our specific approach for permitting heterogeneously expressed concepts is to apply scope reference concept and scope apposition concepts. *Scope reference concepts* are a set of language different from but semantically the same as metaconcepts that share the same locIID. *Scope appositional concepts* are a set of heterogeneously expressed concepts that are semantically different from the language of the same metaconcepts that share the same locIID. The modelling relationships are shown in Figure 4.

Figure 4 Appositional concept



The key of scope reference concepts and scope apposition concepts is the shared locIID, which is generated by one of the semantically the same annotations following the *FISE rule*. All concept information except locIID can be overridden according to the new concept expression requirements. This overriding follows the creation sequence of 'first scope reference concept→other scope reference concept→scope apposition concepts'.

For example, if a French department has defined a first reference concept (33568, réfrigérateur) and set the scope in French, the second English designer should only first browse 'réfrigérateur' in LEPC to continue designing 'refrigerator'. He/she should maintain the same locIID 33568 to create a new 'English' scope reference concept, such as (33568, fab2az3, refrigerator) where 33568 and fab2az3 are mapped. For further designing a scope apposition concept in 'English' scope, a designer can override the English scope reference concept such as (fab2az3, SFH335, freezer) on the condition that the catalogue designer can make sure that 'freezer' and 'refrigerator' are semantically the same.

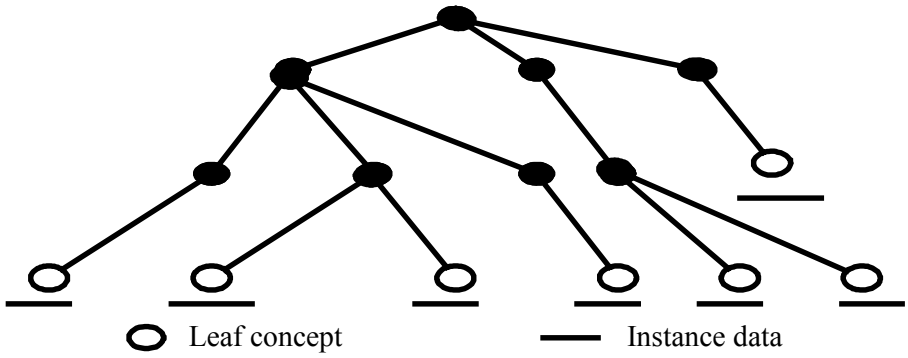
This solution has solved the conflicting semantic problems by permitting heterogeneous local metaconcepts, except that two requirements should be met. First, a designer should understand at least one of the created scope reference catalogues. Nevertheless, a language translation tool could be devised to relieve this requirement. Second, a designer should be able to judge semantic equivalence between his/her concepts and the referenced concepts. Bias may occur and this may be the cost of globalisation. A good concept-mapping guideline and/or some verification mechanisms can decrease such possibility.

2.2.2 Including dynamic instance data

The ultimate purpose of transforming irregular local product definitions is to allow particular product data to be retrieved by remote users in the data supply chain of '*local irregular product definitions*→locRep→comRep→locRep→*local irregular product definitions*'. Therefore, it is important to study how to supply the dynamic particular product data attached on the irregular local product definitions. We call such data as *instance product data*.

In this subsection, we propose a '*data-on-leaf*' model to describe the dynamic instances of *ad hoc* product definitions as shown in Figure 5.

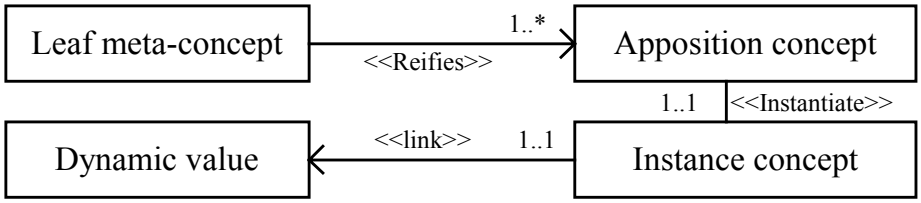
Figure 5 'Data-on-leaf' model



'Data-on-leaf' means that all instance data of a product representation are located on the top of the leaves of a set of metaconcepts. That is, if a product's locIID < m, n > = locPID*(A_1^1, A_i^2, …, A_n^m), then all locAID with *level* < m do not connect to instance data. That is, $\text{locAID}_{\text{level}< m}$ are excluded, where m is the largest level number for each *concept path* (*e.g.*, '6' is the largest level number for locAID = [1, 3, 26, 43, 2, 5]).

Figure 6 intuitively explains the model, where each leaf concept positioned by a locAID connects to a piece of instantiated product data through an apposition concept. The connections are one-to-many relationship between leaf metaconcept and apposition concept, one-to-one relationship between apposition concept and instance concept and one-to-one relationship between instance concept and dynamic value (particular data).

Figure 6 Concept instantiation



There are two functions of the data-on-leaf model. First, it allows the instantiation of a leaf concept to carry a set of dynamic values. Second, it prevents concept semantic conflicts while it allows instantiation. The first function is achieved by dynamically connecting to data sources (see Section 3). The second function is achieved by the following rule.

Instantiation rule: any concept in a product tree can be instantiated only if the concept is a leaf concept that is atomic and cannot be broken down into smaller concepts.

This rule indicates that an instantiation will always be in danger of semantic conflicts if a concept, which is being instantiated, is not atomic or an implicit concept that implies lower-level concepts. For example, a 'refrigerator' has a product tree (like colour, dimension [width, length, height]). If designer *A* instantiates 'dimension' as '29 × 33 × 66', and designer *B* instantiates 'dimension' as '29, 33, 66', it is clear that these two instantiations produce semantic conflicts. The instantiation rule forces that if designer *A* insists on instantiating 'dimension' in his/her own way, s/he should generate a new child concept of 'dimension' such as 'three dimension' to match his/her desired format and make a mapping such that Map (threeDimension, dimension [width, length, height]). However, as a general practice, it is important to follow the existing formats if the leaf concepts have already included the needed expressions. One of our concerns in following the general practice is the proliferation of the concepts in a product tree. Eliminating semantic conflicts is the target of *ad hoc* product data integration, but decreasing the redundant concept number in LEPC is also desirable though the redundancy cannot be prevented from technical designs. This is caused by the designers' preferences and cognition and can be minimised by adopting good business practices.

3 Handling heterogeneous data sources

In the previous sections, we have focused on the process of how local metaconcepts are generated, expressed and dynamically instantiated. To finish the whole definition of the transformation process from irregular product definitions to *locRep*, the remaining issue is how to retrieve *ad hoc* product definitions in heterogeneous data stores.

There are a large number of references for retrieving data from heterogeneous data sources such as (Ball *et al*., 2000; Domenig and Dittrich, 2000; Lee *et al*., 2002). However, since these approaches generally apply to their own problem domains, they are difficult to be directly utilised in our context. In this case, this section has devised a *Picker* object to capture *ad hoc* data sources.

The general idea of Picker (see Appendix 1 of Appendices) is that when a query request arrives in LEPC, the LEPC detects all product data sources (*e.g.*, XML files and relational databases) within its control domain and selects the needed processors. The Picker transforms the query request into the queries understandable by those product sources. The specific methods are:

- For XML sources, the local concept identifiers locIID of local metaconcepts maps onto *XPath expression* (Clark and DeRose, 1999) such that Map(LocalConcept(locIID), SourceConcept(XPath)).
- For relational databases, the local concept identifiers locIID of local metaconcepts maps onto *path expression* that is proposed in the research work of LOREL (Quass *et al*., 1995) and TSIMMIS (Garcia-Molina *et al*., 1997) such that Map(localConcept(locIID), SourceConcept(Path)).

By mapping local concept identifier onto path expression that can query source product data, the query semantics are exactly passed from LEPC to source product catalogues and the dynamic values of path expressions are transferred to the leaf metaconcepts as the instance data.

The underlying design strategy is that the systems of source product catalogues do not provide specific functionalities of how to retrieve *ad hoc* data. *Picker* object (see Appendix 1) selects the data processors such as *xsltProcessor* and *databaseProcessor* defined by the designers. It is the responsibility of designers to provide data retrieval codes. The selected data processors specified by designers process these code fragments to retrieve *ad hoc* data and convert them into normalised instance concepts. This strategy has two important benefits. First, it enables the underlying systems to adapt to millions of heterogeneous product data sources. Product standards are not required for local firms/departments. In contrast, local firms/departments can continue to use their legacy data systems for their ongoing business. Second, it makes the system design simpler, more cost effective, and more applicable in a wider range. The design of specific computational rules for processing input data ('inData' in Picker object) is another separate important research. This paper will not make further discussion as it is worthy of another research article.

4 Implementation discussion

There are two issues concerning the implementation of the concept-centric definition transformation approach. First, what architecture should be selected. Second, what kind of language should be used to implement the LOCAL PRODUCT MAP. In this section, we propose a *concept-centric LEPC architecture* and develop a set of new XML PRODUCT MAP (XPM) documents that are consistent with the basic XPM rules stipulated in Guo and Sun (2003d) to illustrate our implementation.

4.1 Concept-centric LEPC architecture

A generic and geographically dispersed enterprise shows that its organisation is generally in the form of headquarters, regional divisions and their lower-level departments. This organisational form in *ad hoc* electronic product data shows that different regions may use different languages for product data formats, different departments may have different dialects to define products, and different departments may have different data stores. Following this observation, we propose a concept-centric LEPC architecture to collect heterogeneous product data and divide different types of concepts into several architectural components: *Regional Reference Catalogues* (RRCs) for regional divisions, *Local Apposition Catalogues* (LACs) for departments and *Local Data Stores* (LDSs) for maintaining various relational databases and XML data stores. Amongst these components, RRCs are partially replicated in different regions. LACs externally connect to RRCs and internally connect to one or more LDSs. All components are connected through intranet. The aim of this architecture is to build an enterprise-wise local product concept standard that could be structurally consistent with common product concepts as discussed in Guo and Sun (2003d), and enables LEPC designers to further participate in the global common product catalogue discussed in Guo and Sun (2003b–c).

4.2 XML product map documents

An enterprise-wide LEPC is a set of electronic documents (Glushko and McGraith, 2002). The key to the implementation of the LOCAL PRODUCT MAP and the picker object lies on how documents are designed. In this section, we define two kinds of XPM documents respectively for RRCs and LACs.

4.2.1 RRC documents

An RRC document is a set of hierarchically arranged denotation concepts where the set of child denotation concepts is the connotation concepts of the parent concept. Each denotation concept includes a locIID and an annotation. Other items are optional. Mapped onto an XML document, its XML DTD for product map is:

```
<!ELEMENT concept (concept*)>
<!ATTLIST concept locIID NMTOKEN #REQUIRED
                 locAnnotation CDATA #REQUIRED
                 locOption_i CDATA #IMPLIED>
```

Physically, RRC XPM documents can be categorised into two types of metaconcept documents: *local catalogue documents* (locCat.xml) and *local product documents* (locProd.xml). A locCat.xml is a language-scoped catalogue that includes many locProd.xml. A set of RRC documents constitutes a regional LEPC system that connects with other regional LEPCs by replicating the shared locIIDs.

4.2.2 LAC documents

An LAC document is instantiated from RRC documents and contains the information of both RRC and LDS. It reserves the departmental personalised data by inserting apposition concepts. Its XML DTD can be expressed in the following:

```
<!ELEMENT appositions (concept*)>

<!ELEMENT concept (#PCDATA)> <!--The instance concept either from
manually inputted ad hoc data or from #PCDATA for 'code fragment' to
manipulate dynamic value stored in the heterogeneous data stores.-->

<!ATTLIST concept locIID NMTOKEN #REQUIRED
                 appoAnnotation CDATA #REQUIRED
                 appoOption_i CDATA #IMPLIED
                 dataType NMTOKEN #REQUIRED>
```

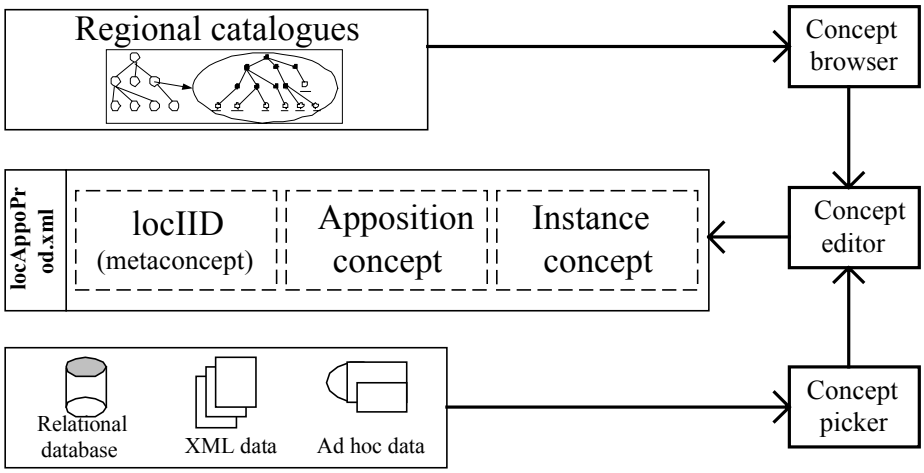
A particular LAC document is an XPM apposition document (*locAppoProd.xml*) that combines a set of local metaconcepts, a set of apposition concepts and a set of corresponding local instance concepts. This document describes a set of particular product definitions that reflect the departmental personalisation and the dynamic values mapped onto the enterprise-wide metaconcepts.

In our current implementation, we have designed a visual concept browser, a visual concept editor and a picker control in Microsoft .NET to bring the three types of concepts together in a locAppoProd.xml document, as shown in Figure 7. The implementation is a part of the global IEPC (interoperable electronic product catalogues) systems (Guo and Sun, 2003b) and includes the following components:

- LEPC browser: a visual control for browsing RRC documents.
- LEPC editor: a visual control that displays the metaconcept browsed from RRC documents and provides editing functionalities to build LAC documents.
- Concept picker: a control for manipulating the 'code fragment' provided by the catalogue designer to edit particular data in different data stores.

Our current implementation experience indicates that the concept-centric *ad hoc* product data integration is conceptually correct and applicable especially for SMEs or departments that have few products for global integration.

Figure 7 Implementation of locAppoProd.xml



5 An extended discussion

This section provides an extended discussion on three general questions that most researchers ask: whether a firm requires a canonical product catalogue to connect to source product catalogues, whether the cost of the concept mapping process is acceptable, and whether LEPC systems can interoperate with popular international product standards or de facto industrial standards.

5.1 Why canonical LEPC?

The answer to why a firm needs a canonical LEPC lies in the fact that each firm is a *semantic community* (Robinson and Bannon, 1991) that has semantic conflicts with others, and there are millions of firms where each may have multiple product data sources. Without a canonical LEPC that can interoperate with Common Electronic Product Catalogues (CEPCs), every data source must maintain a concept-mapping mechanism with CEPCs. The reengineering cost will be huge and the technical requirement will be higher than the cost of maintaining a canonical LEPC. With canonical LEPC that is distributed by CEPC service providers, local firms can easily map source concepts of product data sources onto LEPC concepts that are automatically transformable into CEPCs for remote concept exchange. A comparison between two approaches can be illustrated in the following.

Given that the cost of designing and implementing a canonical LEPC system is *M* dollars, if there are one million firms to participate in CEPCs and each firm has one product data source, then if the canonical LEPC is produced by CEPC service provider and distributed to firms (as users), then each firm only requires ‘1/1million *M* dollars as *X* + a certain amount of service charges as *Y* + editing fees for connecting to data source

as *Z*' to have their heterogeneous product data interoperable. However, if each firm directly produces mapping mechanisms between CEPC and product data source, each firm requires *M* dollars. So the comparison has the following:

$$Result = M - (X + Y + Z).$$

Obvious, only if the cost of $(Y + Z)$ of a firm tends to be [(1 million – 1)/1 million] × *M*, then $M = X + Y + Z$. As we have known, according to the theory of labour division (Smith, 1976) and comparative advantage (Ricardo, 1912), the use of CEPC service provider for canonical LEPC is more profitable than an individual firm producing a mapping mechanism by itself. Therefore, $(Y + Z)$, in general, will not reach to [(1 million – 1)/1 million] × *M* unless a firm has the mapping scale (*i.e.*, need to map a million of data sources or the mapping quantity equals to total quantity of all the customers of the CEPC provider) equal to a CEPC service provider. If such case happens, that firm is suitable for a CEPC service provider to internally provide services and work in collaboration and cooperation with other CEPCs.

5.2 Cost analysis of concept mapping

Given that a firm accepts the argument that the cost of $(X + Y + Z)$ will always be less than *M*, another question is whether there is another method that costs less than *N*, enabling $N - (X + Y + Z) < 0$. Two possible methods are:

1 international standard adoption such as linking to UNSPSC (www.unspsc.org), ec1@ss (www.eclass.de) and HL7 (www.hl7.org)

2 abandon electronic markets and use traditional internationalisation method to participate in global markets.

For some SMEs, the second approach is possible if it has strong global market connections and their products comparatively rely on several known companies. This method is beyond the discussion of this paper.

For the first approach, it is a standard adoption issue (Steinfield *et al.*, 2004). However, adopting existing standards is limited to several aspects (Guo and Sun, 2003a):

- purchasing standard systems that require a considerable amount of money
- reengineering the existing product data source systems that may affect the use of legacy systems
- may retard the process of adapting to the emergent changes of the firm's requirements to insert, delete and modify new data that are beyond the adopted standards
- a certain standard can only cover a limited marketplace.

It is unnecessary to argue whether the standard adoption approach is better or worse than introducing canonical LEPC to interoperate with other firms through CEPCs. The key in judging whether a firm needs to adopt a standard approach is to observe the cost of installing standard systems, the maintenance cost and the benefit from the market coverage of the standard. Therefore, there is no absolute comparison to determine which approach is better. The comparison issue should be left to the judgement of decision makers of individual firms after obtaining the result of '$N - (X + Y + Z)$'.

5.3 Interoperability with popular standards

Given that a firm adopts the canonical LEPC to participate in the global electronic markets, another question is how a firm can interoperate with international standards or *de facto* industrial standards to enlarge market coverage. To answer this question, this subsection discusses the issue by illustrating how a 'fridge' of one firm can be understood by 'réfrigérateur' of another firm through the industrial standard ebXML.

To simplify the discussion, we first present an ebXML purchase order that is encoded in SOAP format, as shown in Appendix 2 (Figure 9). The product 'refrigerator' being purchased is encoded in the XPM format, which links both LEPC and CEPC.

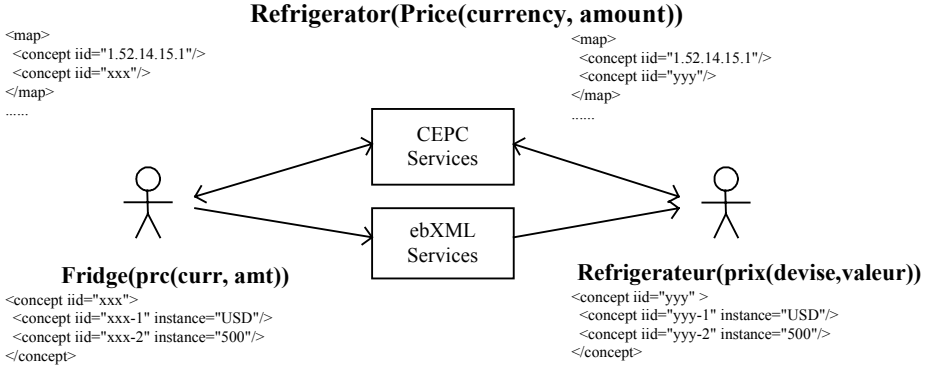
Since ebXML is an open interoperation infrastructure, it provides customers with service registry that links to repositories. The aim of ebXML is not to specify product and service descriptions specifically situated in numerous firms, but to allow them to be discovered through ebXML by combining other standards such as SOAP (Box *et al*., 2000) and WSDL (Christensen *et al*., 2001). Therefore, we should note that the issue of how to make *ad hoc* product representations interoperable with each other is separate and independent.

Nevertheless, *ad hoc* concepts encoded in XPM can be embedded in ebXML for business interoperation as shown in Appendix 2 (Figure 9) of the Appendix. In fact, one of the targets of XPM is to utilise existing industrial standards such as ebXML and UNSPSC (www.unspsc.org). For example, the base concept IIDs of CEPC have been adopted by the UNSPSC classification, such as '1.52.14.15.1' is mapped onto '52141501' for 'domestic refrigerator'.

The embedded heterogeneous product concepts can be exactly exchanged as illustrated in Figure 8. When the firm A sends 'fridge' to firm B, it actually sends the locIID to firm B. When firm B receives the product information, it translates the unknown locIID into its own locIID against the comIID mapped in CEPC. Since locIID is determined by local annotation 'réfrigérateur', firm B can then understand 'fridge'.

The XPM documents devised in this paper have provided canonical local product representations that could be mapped onto common CEPC concepts. When plugging the LEPC/CEPC mapping mechanism into a local concept editor as shown in Figure 8, the LEPC/CEPC mapping process will be automatically achieved. The mapped product concepts can then be embedded in different kinds of business documents for free business interoperation.

Figure 8 Heterogeneous concept exchange between two firms



6 Conclusion

In this paper, we have proposed a concept-centric definition transformation approach to transform *ad hoc* local product definitions to canonical local product representations. Central to this approach is the proposed novel model called LOCAL PRODUCT MAP for generating local metaconcepts, apposition concepts and instance concepts for canonical local product representations. This model has analysed concept generation process and solved the problems of semantic conflict between local concepts.

To obtain the normalised concepts correctly, we have suggested a dynamic *ad hoc* product data-retrieval strategy that allows all *ad hoc* product data to be manually inputted or retrieved against a set of code fragments supplied by designers. This strategy has protected the legacy product data stored in both XML files and relational databases. In this case, different semantic communities are able to maintain their own data preferences that are expressed in different languages, customs and business practices.

The local product map is implemented on a set of XPM documents, which in turn are built on the proposed concept-centric LEPC architecture. The local product concepts implemented in XPM documents can be semantically interoperable between various heterogeneous systems through CEPCs or through ebXML by embedding XPM documents in ebXML documents.

A major contribution of this paper is the novel definition transformation approach for the semantic integration of heterogeneous *ad hoc* product data distributed in different data stores. This allows us to form a set of canonical local product representations that are able to communicate with a set of publicly understandable common product representations (Guo and Sun, 2003c–d). A current constraint of the implementation is that the approach is more suitable for SMEs or small geographical departments that have fewer products and can afford only a small amount of reengineering work to join the global electronic marketplace.

A future direction of this paper is to build retrieval rules for the instance concepts so that only permitted product data instances are presented when remote queries come. This includes a study on how to build XPM rules that could be embedded in LEPC.

Acknowledgements

We thank the anonymous reviewers for their insightful comments, which were invaluable for improving the presentation of this paper. We also thank the editors for their great work to include this paper in the issue.

References

Ball, M., Ma, M., Raschid, L. and Zhao, Z. (2000) 'Supply chain infrastructures: system integration and information sharing', *ACM SIGMOD*, Vol. 31, No. 1, pp.61–66.

Baron, J., Shaw, M. and Bailey, A. (2000) 'Web-based e-catalog systems in B2B procurement', *Communication of the ACM*, Vol. 43, No. 5, pp.93–100.

Bergamaschi, S., Guerra, F. and Vincini, M. (2002) 'A data integration framework for e-commerce product classification', in I. Horroscks and J. Hendler (Eds.) *ISWC 2002*, LNCS 2342, pp.379–393.

Berner-Lee, T., Hendler, J. and Lassila, O. (2001) 'The semantic web', *Scientific American*, May Issue.

Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H., Thatte, S. and Winer, D. (2000) *Simple Object Access Protocol (SOAP) 1.1*, http://www.w3.org/TR/SOAP/

Christensen, E., Curbera, F., Meredith, G. and Weerawarana, S. (2001) *Web Services Description Language (WSDL)*, http://www.w3.org/TR/wsdl

Christiaanse, E. and Kumar, K. (2000) 'ICT enabled coordination of dynamic supply web', *International Journal of Physical Distribution and Logistics Management*, Vol. 30, Nos. 3–4, pp.268–285.

Clark, J. and DeRose, S. (1999) 'XML Path Language (XPath) Version 1.0', *W3C Recommendation*, 16 November, http://www.w3.org/TR/xpath

Dogac, A. and Cingil, I. (2001) 'A survey and comparison of business-to-business e-commerce frameworks', *ACM SIGecom Exchanges*, Vol. 2, No. 2, pp.16–27.

Dogac, A., Tambag, Y., Pembecioglu, P., Pektas, S., Laleci, G., Kurt, G., Toprak, S. and Kabak, Y. (2002) 'An ebXML infrastructure implementation through UDDI registries and RosettaNet PIPs', *Proceedings of 2002 ACM SIGMOD International Conference on Management of Data*, Madison, Wisconsin, USA, pp.512–523.

Domenig, R. and Dittrich, K. (2000) 'A query based approach for integrating heterogeneous data sources', *Proceedings of ACM CIKM 2000*, McLean, VA, USA.

Fensel, D., Ding, Y., Omelayenko, B., Schulten, E., Botquin, G., Brown, M. and Flett, A. (2001) 'Product data integration in B2B e-commerce', *IEEE Intelligent Systems*, Vol. 16, No. 4, pp.54–59.

Garcia-Molina, H., Papakonstantinou, Y., Quass, D., Rajaraman, A., Sagiv, Y., Ullman, J., Vassalos, V. and Widom, J. (1997) 'The TSIMMIS approach to mediation: data models and language', *Journal of Intelligent Information Systems*, Vol. 8, No. 2, pp.117–132.

Ginsburg, M., Gebauer, J. and Segev, A. (1999), 'Multi-vendor electronic catalogs to support procurement: current practice and future directions', *Proceedings of 12th International Bled Electronic Commerce Conference*, Bled, Slovenia.

Glushko, R. and McGraith, T. (2002) 'Document engineering for e-business', *Proceedings of ACM DocEng'02*, McLean, Virginia, USA, pp.42–48.

Goh, C., Bressan, S., Madnick, S. and Siegel, M. (1999) 'Context interchange: new features and formalisms for the intelligent integration of information', *ACM Transactions on Information Systems*, Vol. 17, No. 3, pp.270–293.

Goh, C., Madnick, S. and Siegel, M. (1994) 'Context interchange: overcoming the challenges of large-scale interoperable database systems in a dynamic environment', *Proceedings of ACM CIKM' 94*, Gaitherburg, MD, USA, pp.337–346.

Guo, J. and Sun, C. (2003a) 'Collaborative product representation for emergent electronic marketplace', *Proceedings of 16th Bled Electronic Commerce Conference*, Bled, Slovenia, pp.847–859.

Guo, J. and Sun, C. (2003b) 'Concept exchange: constructing interoperable electronic product catalogues in an emergent environment', *Proceedings of IEEE International Conference of E-Commerce*, IEEE Computer Society, pp.165–172.

Guo, J. and Sun, C. (2003c) 'Context representation of product data', *ACM SIGecom Exchanges*, Vol. 4. No. 1, pp.20–28.

Guo, J. and Sun, C. (2003d) 'Context representation, transformation and comparison for ad hoc product data exchange', *Proceedings of 2003 ACM Symposium on Document Engineering*, Grenoble, France, 20–22 November.

Handschuh, S., Schmid, B. and Stanoevska-Slaveva, K. (1997) 'The concept of a mediating electronic product catalog', *Electronic Markets*, Vol. 7, No. 3, pp.32–35.

Kashyap, V. and Sheth, A. (1996) 'Semantic and schematic similarities between database objects: a context-based approach', *The VLDB Journal*, Vol. 5, pp.276–304.

Keller, A. and Genesereth, M. (1996) 'Multivendor catalogs: smart catalogs and virtual catalogs', *EDI Forum: Journal of Electronic Commerce*, Vol. 9, No. 3, pp.87–93.

Kumar, K. (2001) 'Technology for supporting supply chain management', *Communications of the ACM*, Vol. 44, No. 6, pp.58–61.

Lee, M., Yang, L., Hsu, W. and Yang, X. (2002) 'XClust: clustering XML schemas for effective integration', *Proceedings of ACM CIKM'02*, McLean, Virginia, USA.

Ng, W.K., Yan, G. and Lim, E. (2000) 'Heterogeneous product description in electronic commerce', *ACM SIGecom Exchanges*, Vol. 1, No. 1, pp.7–13.

Omelayenko, B. (2002) 'Integrating vocabularies: discovering and representing vocabulary maps', in I. Horrocks and J. Hendler (Eds.) *ISWC 2002*, LNCS 2342, pp.206–220.

Omelayenko, B. and Fensel, D. (2001a) 'An analysis of B2B catalogue integration problems', *Proceedings of International Conference on Enterprise Information Systems*, Setubal, Portugal.

Omelayenko, B. and Fensel, D. (2001b) 'A two-layered integration approach for product information in B2B e-commerce', *Proceedings of the 2nd International Conference on Electronic Commerce and Web Technologies*, Munich, Germany.

Omelayenko, B., Fensel, D. and Bussler, C. (2002) 'Mapping technology for enterprise integration', *Proceedings of the 15th International FLAIRS Conference*, Pensacola, FL., pp.419–424.

Quass, D., Rajaraman, A., Sagiv, Y., Ullman, J. and Widom, J. (1995) 'Querying semistructured heterogeneous information', *Proceedings of DOOD Conference*, Available in http://www-db.stanford.edu

Ricardo, D. (1912) *The Principles of Political Economy and Taxation*, introduction by Michael P. Fogarty, London: Dent and Dutton.

Robinson, M. and Bannon, L. (1991) 'Questioning representations', *Proceedings of ECSCW'91*, Amsterdam, Holland.

Schulten, E., Akkermans, H., Guarino, N., Botquin, G., Lopes, N., Dörr, M. and Sadeh, N. (2001) 'The e-commerce product classification challenge', *IEEE Intelligent Systems*, July/August, Vol. 16, pp.86–c3.

Segev, A., Wan, D. and Beam, C. (1995) 'Electronic catalogs: a technology overview and survey results', *Proceedings of ACM CIKM'95*, Baltimore, MD, USA, pp.11–18.

Shim, S., Pendyala, V., Sundaram, M. and Gao, J. (2000) 'Business-to-business e-commerce frameworks', *IEEE Computer*, Vol. 33, No. 10, pp.40–47.

Smith, A. (1976) *An Inquiry into the Nature and Causes of the Wealth of Nations*, in R.H. Campbell, A.S. Skinner and W.B. Todd (Eds.), first published in 1776, Oxford: Oxford University Press.

Stanoevska-Slabeva, K. and Schmid, B. (2000) 'Internet electronic product catalogs: an approach beyond simple keywords and multimedia', *Computer Networks*, Vol. 32, pp.701–715.

Steinfield, C., Wigand, W., Markus, M. and Minton, G. (2004) 'Promoting e-business through vertical IS standards: lessons from the US home mortgage industry', *Workshop on Standards and Public Policy*, Federal Reserve Bank of Chicago, Chicago, Illinois, 13–14 May.

Welty, B. and Becerra-Fernandez, I. (2001) 'Managing trust and commitment in collaborative supply chain relationships', *Communications of the ACM*, Vol. 44, No. 6, pp.67–73.

Wombacher, A., Fankhauser, P. and Mahleko, B. (2003) 'Matchmaking for business processes', *Proceedings of IEEE International Conference on E-Commerce*, IEEE Computer Society, pp.7–11.

Appendices

Appendix 1 Picker function

Assume that the data source type is defined in dataType = {XML, relational, plain} to refer to XML files, relational database and other data that have to be manually processed. The input is defined as *inData* = {code fragment, plain strings}. The expect output *outData* = {normalised instance concepts} for the use of LOCAL PRODUCT MAP.

Function: Picker(inData){

if *dataType* := *XML* then {

Open(*xsltProcessor*);

if *outData* := *Process*(*inData*) then return *outData* else *Terminate*(); }

if *dataType* := *relational* then {

Open(*databaseProcessor*);

if *outData* := *Process*(*inData*) then return *outData* else *Terminate*(); }

if *dataType* := *plain* then {

if *inData* ≠ *VOID* then return *outData* : = *inData* else *Terminate*(); }}

Appendix 2 An example of XPM embedded in ebXML

Figure 9 An example of XPM embedded in ebXML

```
<SOAP-ENV:Envelope><SOAP-ENV:Body>
  <eb:Manifest SOAP-ENV:mustUnderstand="1"eb:version="1.0">
    <eb:Reference xlink:href="cid:ebxmlpayload11@boo.com" xlink:role="XLinkRole" xlink:type="simple">
      <eb:Description xml:lang="en-us">Purchase Order 1</eb:Description>
    </eb:Reference>
  </eb:Manifest>
</SOAP-ENV:Body></SOAP-ENV:Envelope>

--BoundarY
Content-ID: <ebxmlpayload11@boo.com>
Content-Type: text/xml
<?xml version="1.0" encoding="UTF-8"?>
  <purchase_order>
    <po_number>1</po_number>
    <product xmlns="http://xpm.boo.org">
      <concept  iid="1.33.26.8.7" >
        <concept iid="1.33.26.8.7.1">
          <concept iid="1.33.26.8.7.1.1" instance="USD"/>
          <concept iid="1.33.26.8.7.1.2" instance="500"/>
        </concept>
      </concept
    </product>
  </purchase_order>
--BoundaryY--
```