Int J Adv Manuf Technol (2008) 37:1250–1260
DOI 10.1007/s00170-007-1052-6

ORIGINAL ARTICLE

Ontology-based semantic models for supply chain management

Yan Ye · Dong Yang · Zhibin Jiang · Lixin Tong

Received: 3 October 2006 / Accepted: 12 April 2007 / Published online: 17 May 2007
© Springer-Verlag London Limited 2007

Abstract The efficiencies of supply chain management (SCM) are often impaired by inconsistent exchange and sharing of knowledge semantics among supply chain partners. To address this problem with semantic integration, this paper presents an approach to developing ontologies of supply chain management (Onto-SCM) as a common semantic model of the SCM domain. The Onto-SCM semantic model is constructed in a modular way in order to enhance its reusability and maintainability. The IDEF5 schematic language is employed to provide the graphical representation of Onto-SCM for intuitive communication between domain experts and users. Furthermore, Ontolingua is adopted to define formal semantics of Onto-SCM for effective knowledge interoperability. In addition, a case study of a printer supply chain is illustrated to demonstrate the proposed approach to semantic integration for SCM. Finally, a prototype is developed to support visualized knowledge modeling of the case system using the IDEF5 schematic language and to implement consistent knowledge transformation among heterogeneous applications in the supply chain.

Keywords Supply chain management · Ontologies · Semantic model · Knowledge

Y. Ye · D. Yang (✉) · Z. Jiang · L. Tong
Department of Industrial Engineering & Management,
School of Mechanical Engineering,
Shanghai Jiao Tong University,
800 Dong Chuan Road, Min Hang District, Shanghai 200240,
People's Republic of China
e-mail: dongyang@sjtu.edu.cn

Y. Ye
College of Mechanical Engineering,
Zhejiang University of Technology,
Hangzhou,
Zhejiang 310032, People's Republic of China

1 Introduction

Supply chain management (SCM) has become a popular approach to enhancing the competitiveness of enterprises in an increasingly competitive and customer-driven market. Its main purpose is to enable a supply chain (SC) composed of geographically distributed enterprises in different business sectors to efficiently transform raw materials into products and to deliver products and relevant services to consumers at the right time and at the right place [1]. For this purpose, supply chain systems need to be provided with high flexibility and agility through effective integration. By flexibility we mean that a supply chain network can meet changing requirements from a variety of customers, and by agility we mean that a supply chain system can rapidly respond to market changes by quickly delivering right products and services. Both require effective information exchange and knowledge sharing among collaborative supply chain partners, which therefore act as a critical success factor for efficient SCM.

Maintaining semantic consistency of shared information is especially crucial for effective information exchange and knowledge sharing between upstream and downstream enterprises in supply chains. Such consistency means that supply chain members need to have a common understanding of the semantic model of the domain. However, today it is common that knowledge interoperability in SCM is often hindered by inconsistent terms and semantics applied by supply chain partners to the descriptions of their knowledge. Inconsistency of terms and semantics may be due to the following factors. First, differences within business contexts and cultures typically exist among different enterprises dynamically participating in supply chains. In general, people in the same organization (enterprise, department) tend to have their own internal vocabularies to represent work domains and to explain implicit and explicit knowledge in their own ways due to the same working backgrounds and business cultures. In contrast, supply chain members affiliated with different organizations may fail to reach a mutual understanding when they communicate using different terms and vocabularies. Secondly, supply chain partners own heterogeneous applications and legacy systems, developed independently with different knowledge modeling schemata. This can be observed by the facts that the same term may be used to denote different concepts, and different terms represent the same entity and concept [2]. For example, the concept *activity* may be represented by the term *operation* or *task*. Finally, the semantics of each term and vocabulary may not be adequately defined or unambiguously explained by the applications. Such inconsistent terms and semantic mismatches impair semantic integration for SCM. As a result, it is rather difficult to achieve effective and efficient SCM without the common semantic model where agreed terms and vocabularies within SCM domain are defined and explained explicitly.

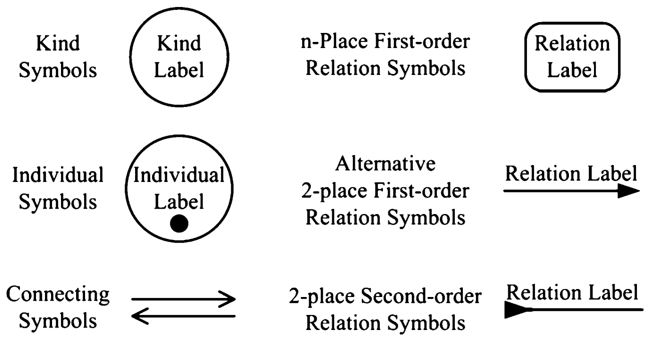
As an unambiguous knowledge representation method, the ontology provides a promising solution to the problem of semantic integration mentioned above. The term ontology comes from philosophy, where it is a systematic account of existence of beings in the world. The term has been adopted by the information science community to describe the knowledge of a domain in a declarative formalism and in a machine-understandable way. An ontology is a formal, explicit specification of a shared conceptualization [3]. Typically, the ontology identifies terms representing domain entities, their intended semantics and formal axioms. It provides a shared and common understanding of a domain that can be communicated between people and heterogeneous applications, thus facilitating semantic exchange, sharing, reuse and interoperability of the knowledge among information systems [4].

Ontology has found wide applications in several domains and systems, such as spatial information systems [5], manufacturing domains [2, 6] and enterprise modeling [7, 8]. However, these applications are not focused on supply chain domain. To the best of our knowledge, only one effort reported by Blomqvist et al. [9] has developed supply chain domain ontologies by means of the formalism of object-oriented constraint networks. It puts much emphasis on a particular area of supply chains, namely supply chain configuration task. In this research, the emphasis is on the semantic model for information integration in efficient SCM. This paper presents an approach of building ontologies of supply chain management (Onto-SCM) as a semantic model to cover all the terminology requirements necessary for SCM systems. First, the IDEF5 schematic language is adopted for graphical representation of Onto-SCM to facilitate visual understanding and communication between knowledge engineers and domain experts. Secondly, the approach uses Ontolingua, an ontology representation language, to provide well-defined and formal semantics for Onto-SCM. Thirdly, the semantic model is organized in a modular way to support the reusability and maintainability of ontologies. Finally, a prototype is developed to graphically model SCM knowledge with the IDEF5 schematic language through its visualization tool IDEFOnto and to enable semantically equivalent transformation and integration of the knowledge represented by different ontologies. As a result, the resulting Onto-SCM model can explicitly represent concepts, relationships, and domain knowledge of SCM and thus provide a semantic foundation for domain knowledge sharing and integration.

The remainder of this paper is organized as follows. The next section briefly describes the IDEF5 and Ontolingua languages. The developed semantic model Onto-SCM is then explained in Sect. 3. Section 4 demonstrates the application of Onto-SCM through a case study of a printer supply chain. Finally, conclusions are given in Sect. 5.

2 IDEF5 and Ontolingua languages

The IDEF5 method developed by Knowledge Based Systems, Inc. (KBSI) has two languages: the IDEF5 schematic language and IDEF5 elaboration language [10]. The IDEF5 schematic language provides visual assistance for the ontology capture process through graphical symbols, some of which are shown in Fig. 1. A circle containing a label represents a *kind*, while a labeled circle including a small, filled-circle represents a specific *individual*. Here, a *kind* and an *individual* can correspond to a *class* and an *instance* in Ontolingua, respectively. In IDEF5, relations are divided into two major categories, namely *first-order relations* and *second-order relations*. The former is defined as the relations among individuals. The latter normally refers to the relationships among kinds, such as subkind-of, or those among kinds and individuals, such as instance-of. On the other


Kind Symbols
Kind Label
n-Place First-order Relation Symbols
Relation Label
Individual Symbols
Individual Label
Alternative 2-place First-order Relation Symbols
Relation Label
Connecting Symbols
2-place Second-order Relation Symbols
Relation Label
Fig. 1 Some basic IDEF5 schematic language symbols

hand, the IDEF5 elaboration language is a structured textual language based on an extended version of first-order predicate calculus called Knowledge Interchange Format (KIF) [11]. Therefore, the language is characterized by strict semantics of the first-order logic. However, it also suffers from three disadvantages. First, the language does not define any modularity construct and therefore can not effectively support the reusability of ontologies. Secondly, it provides comparatively limited language features. For example, it only specifies definition forms for three types of constants: individual, function, and relation. Thirdly, ontologies written in the IDEF5 elaboration language are rather complicated and thus are not easy to understand.

Ontolingua [12, 13] developed by Knowledge Systems Laboratory (KSL) at Stanford University is the most expressive of all the languages that have been used for representing ontologies, allowing the representation of concepts, taxonomies of concepts, n-ary relations, functions, axioms, instances and procedures [14]. Furthermore, Ontolingua is essentially a translation mechanism for ontologies because it supports the translations between multiple representation languages such as Ontolingua, Loom, Epikit, Algernon and pure KIF.

Ontolingua is based on KIF, that is, it has the same logic formalism as the IDEF5 elaboration language. However, different from the IDEF5 elaboration language, Ontolingua defines a language feature *theory* as the main modularity construct and principal building block of the ontology library. A *theory* is a collection of definitions that are somehow related. Typically, a *theory* can include other theories, which means that all the definitions in the included theories are also available in the including theory. In addition, another advantage of Ontolingua is that it extends KIF with a *Frame Ontology* to define object-oriented and frame-language terms [15]. Then, Ontolingua definitions can not only directly contain the KIF primitives and syntax but also use the *Frame Ontology* vocabulary, which therefore enhances the convenience and expressiveness of ontology building.

Ontolingua provides the definition forms of classes, relations, functions and objects (instances) and decomposes each definition into three parts. The first part is a name with an argument list. The second is an informal part documented in natural language. The third is a formal part composed of keywords and sentences using vocabularies in KIF or in the *Frame Ontology*. A simple definition example is as follows.

```
(define-class Student (?x)
  'A student is a person enrolled in a school.'
  :axioms (and (subclass-of Student Person) (do-
      main-of Student has-name) (domain-of Stu-
      dent has-id) (slot-cardinality Student has-id
      1)))
```

The definition describes a class *Student*. The class name *Student* and an argument *?x* form the first part of the definition. Its informal part is the natural language text within a single quotation mark. In the formal part, the sentences labeled by the keyword *:axioms* specify the characteristics of the class *Student*. Among these sentences, the first sentence *(subclass-of Student Person)* means the class *Student* is the subclass of the class *Person*. The next two sentences with the term *domain-of* represent the class *Student* is the domain restriction of two binary relations *has-name* and *has-id*. The last sentence with the term *slot-cardinality* expresses that a student only has an identity number.

3 Ontologies of supply chain management (Onto-SCM)

The purpose of developing Onto-SCM is to provide shared terminologies for representing general concepts and relationships of the SCM domain and therefore to serve as a neutral means for the effective semantic exchange and integration of inconsistent terms used by different supply chain partners. Figure 2 shows the modularization structure and application framework of Onto-SCM.

Onto-SCM is decomposed into five modular theories: SC-Structure theory, SC-Activity theory, SC-Resource theory, SC-Item theory and SC-Management theory, as shown in Fig. 2. These theories form a dependency hierarchy where the SC-Management theory depends on the top four theories and each theory defines a set of related terms. The SC-Structure theory defines the representational machinery for supply chain structures, strategies, goals and

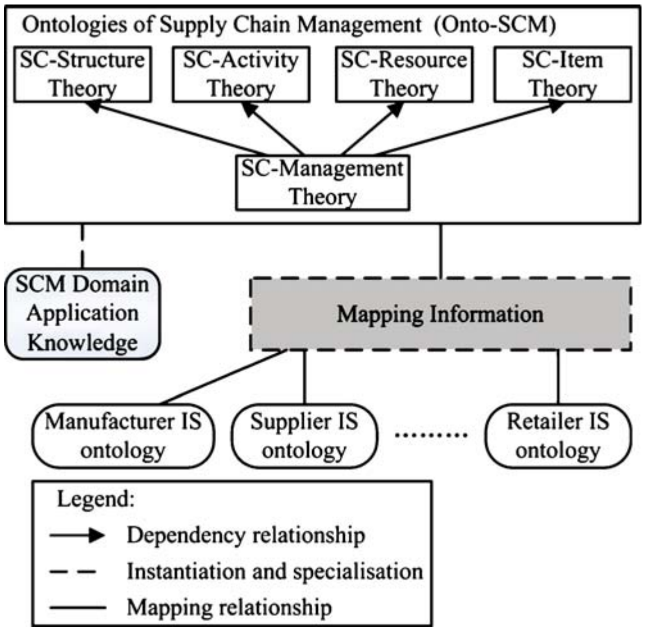


Fig. 2 Modularization structure and application framework of Onto-SCM

processes. The SC-Activity theory provides vocabularies for describing activities, their hierarchy and dependency relationships within supply chain management systems. The SC-Resource and SC-Item theories specify language primitives to represent concepts and relations related to resources and business objects. Finally, the SC-Management theory defines constraint relationships in the supply chain management and operations processes. Basic terms and internal associations in the five theories are shown in Fig. 3 and are explained in Sect. 3.1. For brevity, informal descriptions of some definitions are omitted.

As shown in Fig. 2, the vocabularies defined in Onto-SCM can be used, instantiated or specialized to describe the application knowledge of specific SCM domain. Moreover, enterprises in supply chains, such as material suppliers, manufacturers and retailers, can realize semantic sharing and integration of the knowledge by defining mapping information between their information system ontologies and the shared Onto-SCM model, which is described in Sect. 3.2 and is exemplified in Sect. 4.

3.1 Modular theories

3.1.1 SC-Structure theory

The SC-Structure theory contains vocabularies for describing the knowledge about supply chain structures and strategies. The *Supply_Chain* class represents a set of networks composed of different enterprises, which act as different roles, such as suppliers, manufacturers, forwarders, distributors and retailers, and further form the upstream and downstream links and the supplier-buyer relationships to meet the requirements of external customers. The *SC_Structure* class represents such links and relationships and can be specialized into five subclasses: *Dyadic_Structure, Serial_Structure, Divergent_Structure, Convergent_Structure* and *Network_Structure* [16]. The dyadic structure is made up of two enterprises, e.g., buyer and vendor. The serial structure results from cascading several dyadic structures. A typical serial supply chain usually contains retailers, distributors, manufacturers and suppliers. Both the divergent and convergent structures are modified serial structures. In a divergent structure, one supplier distributes products to several downstream enterprises. In a convergent structure, several components and materials provided by upstream enterprises are assembled by a manufacturer. A network structure, a combination of the convergent and divergent structures, represents a complex supply chain.

The *Strategy* class represents the knowledge about the development vision of SCM systems that can affect decisions related to dynamic configuration of supply chain structures, the use of resources and the execution of activities. Generally, a strategy is embodied by several goals to more concretely guide activities and resources. Furthermore, a goal can further break down into several sub-goals through the relation *has-subgoal* and is realized by the effective implementation of a supply chain process, which is represented by an abstract class *SC_Process*. The following two Ontolingua definitions formally describe two relevant relations *has-subgoal* and *has-process*.

(**define-relation** has-subgoal (?super ?sub)
'(*has-subgoal* ?super ?sub) means that the goal ?super has a sub-goal ?sub. This relation is irreflexive and anti-symmetric. In other words, a goal cannot be a sub-goal of itself and two goals cannot be sub-goals of each other.'
:def (and (Goal ?super) (Goal ?sub))
:axioms (and (irreflexive has-subgoal) (antisymmetric has-subgoal)))
(**define-relation** has-process (?x ?y)
'The relation is a one-to-one binary relation that maps a supply chain to a supply chain process.'
:def (and (Supply_Chain ?x) (SC_Process ?y))
:axioms (one-to-one has-process))

3.1.2 SC-Activity theory

The SC-Activity theory defines common terminologies describing the knowledge of activities within SCM systems. Its core class *Activity* represents something done over a particular time interval [8] that uses certain resources to satisfy given goals, as shown in Fig. 3. The class can be specialized or instantiated to represent various activities, such as order acquisition and product delivery. The formal semantics of the *Activity* class is shown as follows.

(**define-class** Activity (?x)
:def (individual-thing ?x)
:axioms (and (domain-of Activity subactivity-of)
(domain-of Activity has-subactivity)
(domain-of Activity connected-activities)
(range-of Activity connects-activities)))

In this definition, the sentences labeled by the keyword *: axioms* express that the class is the domain and range restrictions of four binary relations. The first two relations *subactivity-of* and *has-subactivity* are used to describe the part-whole relationship between two activities. Moreover, *has-subactivity* is an inverse of *subactivity-of*, as shown in the following definition:

(**define-relation** subactivity-of (?sub ?super)
:def (and (Activity ?sub) (Activity ?super))
:axioms (and (irreflexive subactivity-of) (anti-symmetric subactivity-of) (inverse has-subactivity)))

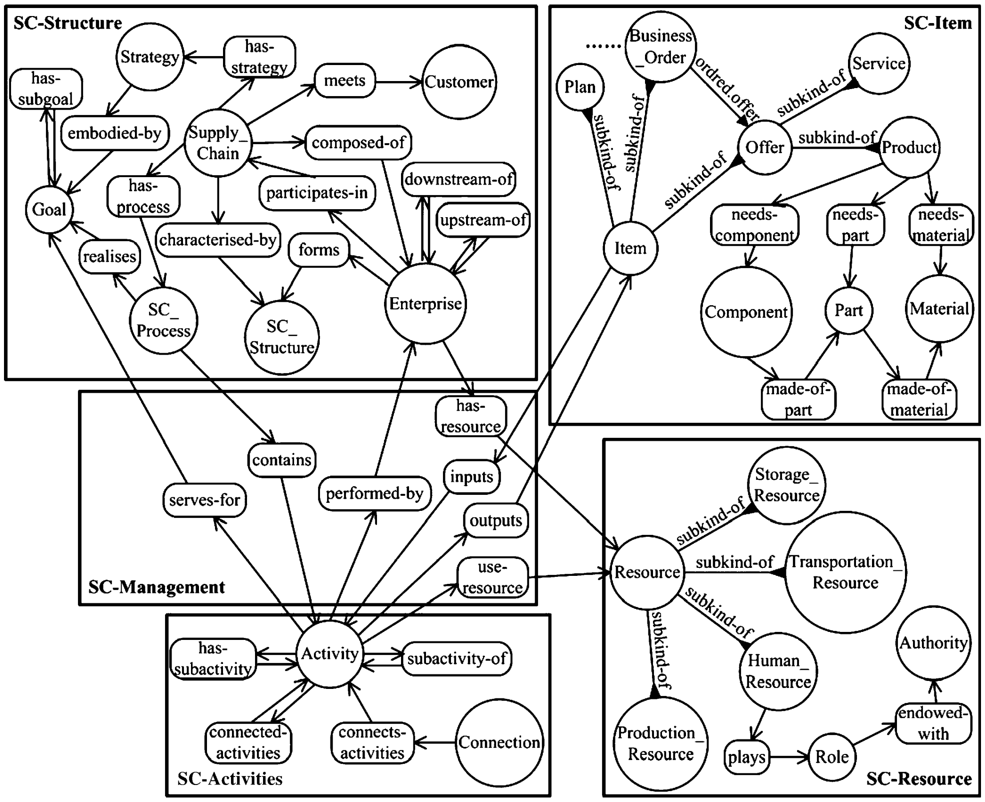


Fig. 3 Graphical representation of basic classes and relations in Onto-SCM with the IDEF5 schematic language

The last two binary relations *connects-activities* and *connected-activities* can build the connection relationship between two activities. In fact, such a relationship forms a logically existent activity that is represented by an abstract concept *Connection*. In the formal description of *Connection* given below, the sentence *(connects-activities C A)* means that *C* is a connection and *A* is an activity. Moreover, the minimal number of activities in the connection *C* is specified as 2 by the sentence with the relation primitive *minimum-slot-cardinality*. In addition, the sentence *(connected-activities A B)* means that two activities *A* and *B* connect and they are not the sub-activity of each other.

```
(define-class Connection (?x)
   :def (and (Activity ?x) (exists (?a ?b)
      (and (Activity ?a) (Activity ?b) (connects-
      activities ?x ?a) (connects-activities ?x ?b)
      (connected-activities ?a ?b))))
   :constraints (and (minimum-slot-cardinality ?x ?
        connects-activities 2) (=> (connected-activi-
        ties ?a ?b) (and (not (Connection ?a)) (not
        (Connection ?b))))))
```

The class *Connection* is normally specialized into several subclasses to describe different types of connection, such as *Sequential_Connection, Parallel_Connection, Conditional_Connection* and *Iterative_Connection*. For example, in the following definition of the class *Sequential_Connection*, the sentence *(Connection ?x)* represents the class is the subclass of the *Connection* class and the other complicated sentence in the formal part gives the semantic description of the class *Sequential_Connection* in detail.

```
(define-class Sequential_Connection (?x)
   :def (and (Connection ?x)(exists (?a ?b) (and
      (Activity ?a) (Activity ?b) (connects-activities
      ?x ?a) (connects-activities ?x ?b) (connected-
      activities ?a ?b) (forall ?p (and ((Item ?p) (=>
      (outputs ?a ?p) (inputs ?p ?b))))))))
```

3.1.3 SC-Resource theory

The SC-Resource theory mainly describes resources, their attributes and relationships in SCM domain. The basic class *Resource* is an important part of the capabilities of enterprises and supply chains, representing a support mechanism for the execution of activities. It has wide scope and various types. At the higher level of abstraction, it may consist of the subclasses *Production_Resource, Storage_Resource, Transportation_Resource* and *Human_Resource*. They can also be specialized to more specific subclasses at the lower level of abstraction to refer to a certain type of machines, materials and employees. For example, the classes *Vehicle, Container* and *Conveyor* are the subclasses of the *Transportation_Resource* class. It deserves to be noted that the

Human_Resource class describes a set of employees that usually use, manage or maintain other types of resources and play different roles endowed with certain authorities. Furthermore, the class *Role* is defined as the super-class of specific role subclasses, such as *Board_Chairman, General_Manager, Department_Manager* and *General_Employee*.

In addition, different attributes of the above classes can be expressed by Ontolingua functions. For instance, the function *conveyor.length* denotes the conveying length of the *Conveyor* class, which can be stated in various known length units through the use of the standard-dimensions theory included in the SC-Resource theory, as shown in the following two definitions.

```
(define-function conveyor.length (?con) :->?clen
  :def (and (Conveyor ?con) (Length_Quantity ?clen))
(define-class Length_Quantity (?lq)
  'The class is a scalar quantity of the physical
  dimension 'length'.'
  :def (and (scalar-quantity ?lq) (= (quantity.dimension
     ?lq) length-dimension)))
```

3.1.4 SC-Item theory

In the SC-Item theory, the basic class *Item* describes a set of business objects that flow through activities in supply chains. These objects are inputted into activities and are exported through the transformation behavior of the activities. Moreover, they can be either physical, such as products and orders, or abstract, such as the plan, demand forecast and service.

The *Item* class has two important subclasses: *Offer* and *Business_Order*. The former is used to explicitly represent product-service 'hybrids' and is the super-class of two classes *Product* and *Service*. The latter is considered as the power of driving supply chain operations and has the subclasses *Customer_Order, Purchase_Order* and *Sales_Order*. The formal definitions of the *Product* and *Business_Order* classes are described as follows.

```
(define-class Product (?x)
  :axioms (and (subclass-of Product Offer)
           (domain-of Product needs-component)
           (domain-of Product needs-parts)
           (domain-of Product needs-material))
(define-class Business_Order (?x)
  :def (and (Item ?x) (value-type ?x ordered.offer Offer)
       (value-type ?x ordered.quantity Natural)
       (value-type ?x priority Priority_Value)
       (value-type ?x issue-date-time Datetime)
       (value-type ?x delivery-date Date))
```

In the definition of the class *Product*, three binary relations in the sentences with the relation term *domain-of* are used to express the knowledge of product structures in both discrete and continuous manufacturing. In the case of discrete manufacturing such as automobile industry, a product is usually assembled from components and/or parts. Furthermore, components are made up of parts and parts result from processing raw materials, as shown in Fig. 3. In the case of continuous manufacturing such as chemical industry, products are normally produced by chemical reactions among several types of raw materials according to a certain proportion. In addition, the definition of the class *Business_Order* shows that the class is the independent variable of two relations, namely *ordered.offer* and *ordered.quantity*, and three functions, namely *priority, issue-date-time* and *delivery-date*. Among these relations and functions, the relations are used to indicate the ordered products and/or services and their quantities. The functions can create the priority value, issue date and time, and delivery date of an order, respectively.

3.1.5 SC-Management theory

The SC-Management theory combines the above four theories mainly by specifying constraint relationships between concepts in SCM systems. These concepts may come from different theories, as shown by several basic relations in Fig. 3. Among these relations, six relations are used to express relationships between the class *Activity* from the SC-Activity theory and the classes from other three theories. The last relation *has-resource* describes that certain resources are located at given enterprises in supply chains.

3.2 Semantic mapping for SCM

Different enterprises in supply chains normally adopt different applications that are based on different ontologies (e.g., manufacturer IS ontology and supplier IS ontology simplified shown in Fig. 2) involving inconsistent terms and semantics, which influences effective semantic interoperability for supply chain integration. The developed Onto-SCM model identifies general terms representing concepts and relationships common to all the applications and therefore provides a neutral interchange language for knowledge sharing and integration across the heterogeneous applications in these enterprises. Thus, the problem of semantic integration caused by inconsistent terms and semantics can be addressed by implementing semantic mappings between different application ontologies and the interlingua Onto-SCM, denoted by the box with the dashed line in the application framework in Fig. 2.

Typically, semantic mappings between two interacting ontologies express semantic equivalences of terminologies within these ontologies. Therefore, the mapping between application ontologies of supply chain partners and Onto-

SCM in Fig. 2 determines instance translations between concepts/relations in the applications and corresponding concepts/relations in Onto-SCM. In this subsection, an approach to establishing semantic mapping based on KIF rules is described.

In fact, the application of semantic mappings can transform the knowledge represented by terminologies in one ontology to the corresponding knowledge represented by semantically equivalent terminologies in the other ontology. In other words, when the knowledge in the first ontology is known, the semantically consistent knowledge in the second ontology can be inferred through the mappings. From this perspective, such mappings can be expressed as rules with the form of an implication between the premise and consequent, which means that the conditions specified in the consequent must hold whenever the conditions specified in the premise are satisfied. KIF supports the representation of such rules, where both the premise and consequent are KIF sentences. In addition, considering that the Onto-SCM model is formally defined in Ontolingua that is based on KIF, it is instinctive to adopt KIF to describe semantic mapping rules.

For convenient explanation, it is assumed that an ontology *A* contains three classes: *aclass1, aclass2, aclass3* and two relations, namely *arel1* and *arel2*, and another ontology *B* has two classes, namely *bclass1* and *bclass2*, and one relation *brel1*. Moreover, there exist given semantic mapping relationships between the two ontologies.

Semantic mapping falls into two major categories: one-to-one mapping and complex mapping. The former shows the matches between a pair of terms belonging to two ontologies, which are represented by KIF rules, as exemplified in the following rule:

(=>> (aclass1 ?x) (bclass1 ?x)) .

The KIF rule represents the class *aclass1* in the ontology *A* is mapped to the class *bclass1* in the ontology *B*. Moreover, an inverse mapping is hold for the pair of terms and is represented by a reverse rule, namely

(<<= (aclass1 ?x) (bclass1 ?x)).

A complex mapping specifies that a combination of terminologies in one ontology corresponds to a combination in the other. Such combination can be created in many ways, such as the disjunction or conjunction of terms. For example, the following rules build complex mappings among terms in the two example ontologies.

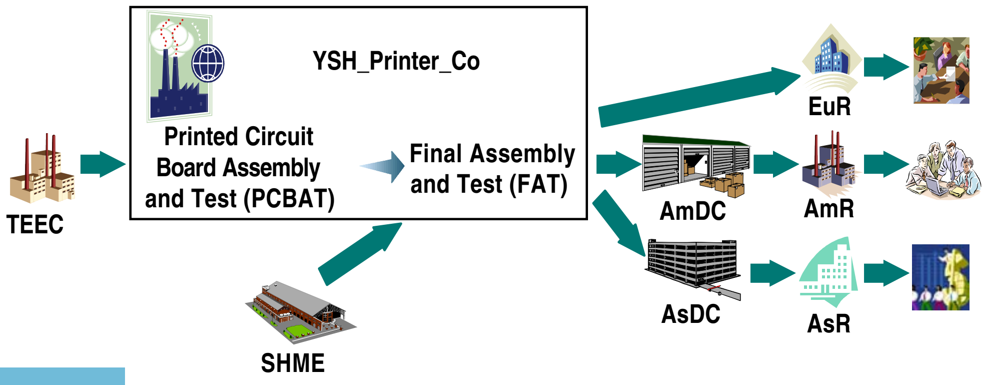
(=>> (bclass2 ?x) (or (aclass2 ?x) (aclass3 ?x)))

(=>> (and (arel1 ?x ?y) (arel2 ?y ?z)) (brel1 ?x ?z))

The first rule shows the class *bclass2* in the ontology *B* is mapped to the disjunction of two classes *aclass2* and *aclass3* in the ontology *A*, that is, the class *bclass2* is transformed to the class *aclass2* or *aclass3*. The second rule indicates that the conjunction of two relations *arel1* and *arel2* in the ontology *A* is translated to the relation *brel1* in the ontology *B*.

4 A case study for application of Onto-SCM

A printer supply chain (PSC) shown in Fig. 4 is used to illustrate the application of Onto-SCM. For the sake of simplicity, only two suppliers (TEEC and SHME), one manufacturer (YSH_Printer_Co), two distributors (AmDC and AsDC) and three retailers (EuR, AmR and AsR) are considered. TEEC is an electronic company in Tokyo, Japan and provides electronic parts such as ROMs and printed circuit boards (denoted by EConA, for simplicity). SHME is located in Shanghai, China and supplies parts such as printer boxes and electro-motors (EConB). The manufacturer YSH_Printer_Co is responsible for the main manufacturing process of printers, which includes the activities of the printed circuit board assembly and test (PCBAT) and the final assembly and test (FAT). The first activity processes EConA provided by TEEC to produce printer head drive boards (PH_Drive_Board). These boards and EConB from SHME are inputted into the FAT activity that assembles and tests printers. These enterprises are members of this supply chain and also can participate in other supply chains. These statements can be formally


YSH_Printer_Co
Printed Circuit Board Assembly and Test (PCBAT)
Final Assembly and Test (FAT)
TEEC
SHME
EuR
AmDC
AmR
AsDC
AsR
Fig. 4 A printer supply chain scenario

represented by instantiating, specializing or using terminologies in Onto-SCM, as shown in the definitions as follows.

```
(define-instance YSH_Printer_Co (Manufacturer)
   :assertions (and (participates-in YSH_Printer_Co
            PSC)
            (downstream-of YSH_Printer_Co TEEC)
            (downstream-of YSH_Printer_Co SHME)
            (upstream-of YSH_Printer_Co AmDC)
            (upstream-of YSH_Printer_Co AsDC)
            (upstream-of YSH_Printer_Co EuR)))
(define-instance AmDC (Distributor)
   :assertions (and (participates-in AmDC PSC)
            (downstream-of AmDC YSH_Printer_Co)
            (upstream-of AmDC AmR)))
(define-instance AmR (Retailer)
   :assertions (and (participates-in AmR PSC)
            (downstream-of AmR AmDC)))
(define-instance EuR (Retailer)
   :assertions (and (participates-in EuR PSC)
            (downstream-of EuR YSH_Printer_Co)))
(define-class PCBAT (?x)
   :def (and (Activity ?x) (performed-by ?x YSH_
        Printer_Co) (inputs EConA ?x) (outputs ?x
        PH_Drive_Board) (exist ?y (=> (FAT ?y)
        (connected-activities ?x ?y)))))
(define-instance Seq_Cnt1 (Sequential-Connection)
   :assertions (exist (?x ?y) (=> (and (PCBAT ?x)
            (FAT ?y)) (and (connects-activities
            Seq_Cnt1 ?x) (connects-activities Seq_Cnt1
            ?y)))))
```

In Fig. 4, eight enterprise individuals participate in the instance *PSC* of the *Supply_Chain* class and form given upstream and downstream relationships between each other. For example, the instance *YSH_Printer_Co* of the class *Manufacturer* in the centre of the supply chain network is associated with other individuals of the classes *Supplier, Distributor* and *Retailer* through the *downstream-of* and *upstream-of* relations. Moreover, two activities performed by *YSH_Printer_Co* are represented by the subclasses *PCBAT* and *FAT* of the *Activity* class, between which there exists the instance *Seq_Cnt1* of the class *Sequential-Connection* due to the following facts. By using the instance *EConA* of the *Part* class as the input, *PCBAT* outputs the instance *PH_Drive_Board* of the class *Component* that is then transformed, together with the instance *EConB* of the class *Part*, into the instance *Printer* of the *Product* class.

In addition, a visualization ontology tool IDEFOnto is developed to graphically model the above knowledge with the IDEF5 schematic language. Figure 5 shows the screenshot of the tool. The right of the screen displays the graphical representation of the case knowledge and the left shows entities (classes, relations, functions, instances/individuals) both in the supply chain scenario and in the Onto-SCM model in a hierarchical way, which offers the convenience of browsing and navigating knowledge. IDEFOnto provides user-friendly, easy-to-operate interfaces for knowledge engineers and domain experts to intuitively express and display knowledge. These users can add new entities by selecting menu commands or dragging and dropping buttons in the toolbar and can also easily modify or delete the existing entities. Furthermore, IDEFOnto can store ontologies and knowledge not only in the form of diagrams but also in Ontolingua ontology files through transformation.

Enterprise individuals in Fig. 4 need to seamlessly exchange information to build good collaboration relationships and in turn to realize efficient SCM. However, owing to long-term business practices and cultures, different enterprises may use different information models or structures. For example, the supplier TEEC and the manufacturer YSH_Printer_Co use UBL and OAGIS, respectively, to describe their business information. They may fail to have a common understanding of the exchanged information due to the differences and inconsistencies between these models. By determining semantic mapping rules between UBL (OAGIS) and the Onto-SCM model, knowledge sharing and interoperability between heterogeneous applications using UBL and OAGIS, respectively, can be implemented.

OAGIS [17] builds a common business object document (BOD) message architecture for communication between business applications, which provides a self-describing mechanism of BODs by defining XML schemas for the following four levels. The first level introduces meta-data, represented as elements and attributes, to describe the BOD itself and the application creating the BOD, such as *ApplicationArea* and *revision*. The second level contains data types that are based on either predefined types or user-defined types, such as *AddressId*. The third level describes extensible component types that are the large-grained building blocks of business documents, such as *PaymentTerms*. The top level defines various BODs, which contain business objects and actions that are to be applied to the objects, such as *AddPurchaseOrder, PurchaseOrder* and *Add*.

UBL [18] is intended to solve the problems caused by multiple industry-specific data formats, which accomplish the same purpose in different business domains, by defining a generic XML interchange format for business documents that can be extended to meet the requirements of particular industries. Specifically, UBL 1.0 provides a library of XML schemas for reusable data components and common business documents constructed from these components,

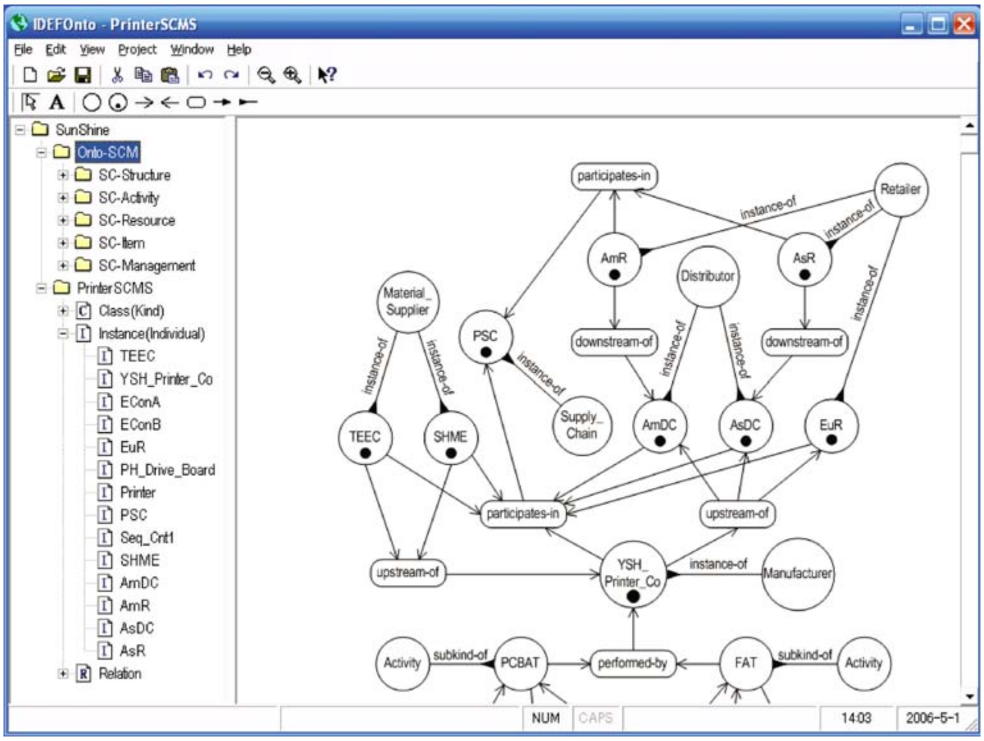


Fig. 5 Knowledge representation of the scenario example in IDEFOnto

such as *Address* and *PaymentMeans*. Currently, basic document types designed by UBL to support a generic order-to-invoice business process include *OrderType, OrderResponseSimpleType, OrderResponseType, OrderChangeType, OrderCancellationType, DespatchAdviceType, ReceiptAdviceType*, and *InvoiceType*.

The detailed specifications of the OAGIS and UBL standards can be found in [17, 18]. These standards define different terminologies and their semantics for business document domains. Therefore, for consistent knowledge exchange and sharing, semantic mapping rules among terms in OAGIS, UBL and Onto_SCM need to be built based on the approach described in Sect. 3.2, as shown in the following KIF rules.

Rule 1 (=>> (oagis-PurchaseOrder ?x) (onto_scm-Purchase_Order ?x))

Rule 2 (=>> (onto_scm-Purchase_Order ?x) (ubl-OrderType ?x))

Rule 3 (=>> (onto_scm-issue-date-time ?x ?y) (ubl-IssueDate ?x ?y))

Rule 4 (=>> (and (oagis-Header ?x ?y) (oagis-DocumentDateTime ?y ?z)) (onto_scm-issue-date-time ?x ?z))

The first three KIF rules specify one-to-one mappings between different pairs of terms belonging to OAGIS, Onto_SCM and UBL. Among them, Rule 1 represents the *PurchaseOrder* class in OAGIS is mapped to the class *Purchase_Order* in Onto_SCM. Rule 2 shows the class *Purchase_Order* in Onto_SCM is translated into the *OrderType* class in UBL. Rule 3 expresses the function *issue-date-time* in Onto_SCM is transformed to the *IssueDate* function in UBL. The last complex mapping rule indicates the combination of the *Header* relation and the *DocumentDateTime* function in OAGIS is mapped to the *issue-date-time* function in Onto_SCM.

By applying these mapping rules to relevant knowledge in the application using OAGIS in YSH_Printer_Co, the knowledge can be consistently understood and reused by the application using UBL in TEEC. This can be easily observed by the transformation of knowledge semantics based on the neutral interchange model Onto-SCM shown in Fig. 6. Initially, the application using OAGIS contains an instance *ORDER1* of the class *PurchaseOrder* that has *PO_HEADER1* and *25_May_2006* as its header and creation date, respectively. The knowledge can be consistently translated into the corresponding facts represented in Onto-SCM through the use of Rule 1 and 4. Then, by applying Rule 2 and 3, the facts are further translated to semantically equivalent knowledge represented in UBL. Similarly, the consistent translation of the knowledge in UBL into those in OAGIS can be realized based on the corresponding inverse mapping rules. Thus, heterogeneous applications of supply chain members can effectively share and integrate the knowledge in SCM systems by building and executing the semantic mapping rules between application ontologies and Onto-SCM.

In this research, a prototype is developed to provide the functions of creating knowledge based on different application ontologies (e.g., OAGIS and UBL), and further

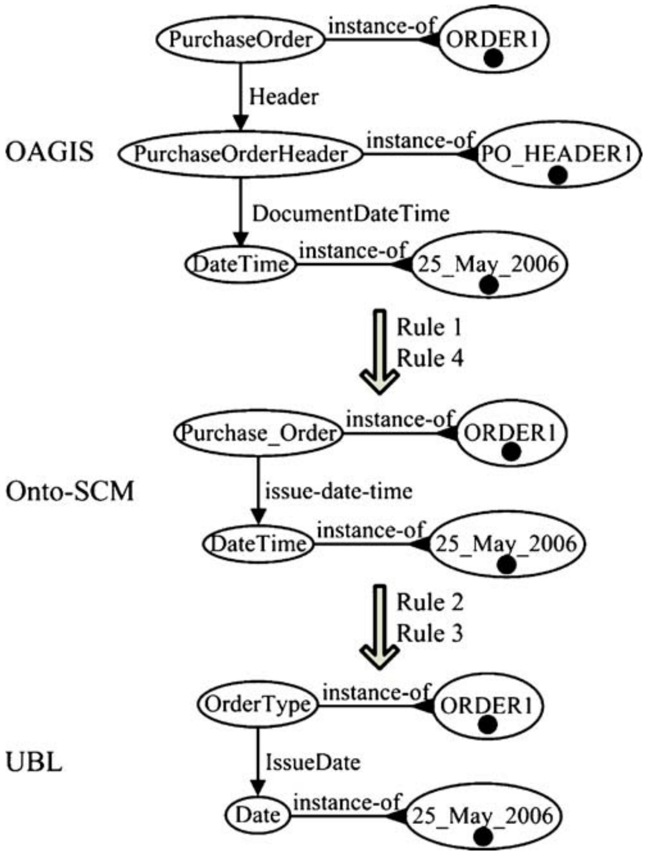


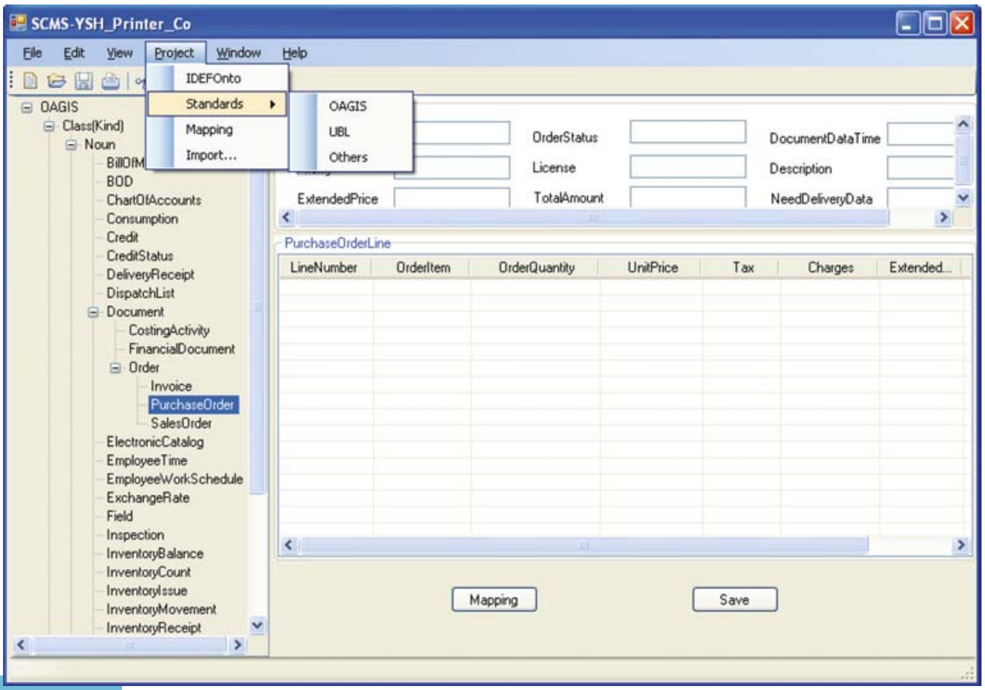
Fig. 6 Semantically equivalent transformation based on mapping rules in KIF

implementing semantically equivalent transformation and integration of the knowledge based on semantic mapping rules for different enterprises in supply chains, in addition to the capability of visualized knowledge modeling offered by its IDEFOnto tool. Figure 7 shows a user interface of the prototype used by the manufacturer YSH_Printer_Co. The left shows the OAGIS model in a hierarchical way. When an entity (e.g., the *PurchaseOrder* class) in the model is selected, the right displays the relations and functions related to the entity, which can be instantiated by users in YSH_Printer_Co. Then, through the *Mapping* button, the knowledge resulting from such instantiation can be transformed to consistent knowledge in Onto-SCM that can be saved and sent to other supply chain partners. On the other hand, the partners can further translate the received knowledge into semantic equivalences represented by their own models (e.g., UBL). Similarly, the above knowledge of the entire printer supply chain represented in Onto-SCM can be transformed to corresponding knowledge represented by the ontologies of relevant applications in YSH_Printer_Co, which is then consistently understood and reused by these applications.

5 Conclusions

Effective knowledge exchange and sharing between cooperative enterprises are fundamental to correct operations of SCM. However, it often happens that the smooth operations of SCM may be hindered by inconsistent terminologies and semantics due to long-term discrepancies in business backgrounds and cultures as well as the heterogeneity of enterprises and their applications. Based on the ontology-based modeling approach, this paper presents a semantic model Onto-SCM to enable consistent exchange and

Fig. 7 A user interface of the prototype



sharing of knowledge semantics in efficient SCM. The IDEF5 schematic language is used to visually represent core concepts and relationships in Onto-SCM, which thus can facilitate communication between knowledge engineers and domain experts. The precise syntax and formal semantics of Onto-SCM are defined in Ontolingua, a highly expressive language and a translation mechanism for ontologies in multiple representation languages, to support semantic interoperability of SCM systems. Moreover, Onto-SCM contains five interdependent modular theories, which helps to enhance the reusability and maintainability of the semantic model. In addition, the developed prototype, on the one hand, includes a visualization tool IDEFOnto to provide user-friendly interfaces for domain experts of SCM systems to conveniently add, modify, delete, browse and communicate knowledge using the IDEF5 schematic language; on the other hand, can be used by different supply chain partners to effective create and integrate the knowledge represented by different semantic models. Future work includes the development and implementation of the semantic mapping rules between Onto-SCM and existing information models that have been adopted by enterprises, in addition to the mentioned OAGIS and UBL. Moreover, the prototype tools will be further enhanced to take full advantage of the proposed semantic models.

Acknowledgements This work was supported by a research grant from the National Natural Science Foundation of China (Project No. 70471023).

References

1. Ulieru M, Cobazru M (2005) Building holonic supply chain management systems: an e-logistics application for the telephone manufacturing industry. IEEE Trans Ind Inform 1:18–30
2. Lin HK, Harding JA, Shahbaz M (2004) Manufacturing system engineering ontology for semantic interoperability across extended project teams. Int J Prod Res 42:5099–5118
3. Studer R, Benjamins VR, Fensel D (1998) Knowledge engineering: principles and methods. Data Knowl Eng 25:161–197
4. Pinto HS, Martins JP (2004) Ontologies: How can they be built? Knowl Inform Syst 6:441–464
5. Benslimane D, Leclercq E, Savonnet M, Terrasse M-N, Yétongnon K (2000) On the definition of generic multi-layered ontologies for urban applications. Comput Environ Urban Syst 24:191–214
6. Grüninger M (2004) Ontology of the process specification language. In: Staab S, Studer R (eds) Handbook on ontologies. Springer, Berlin Heidelberg New York, pp 575–592
7. Fox MS, Gruninger M (1998) Enterprise modeling. AI Mag 19:109–121
8. Uschold M, King M, Moralee S, Zorgios Y (1998) The enterprise ontology. Knowl Eng Rev 13:31–89
9. Blomqvist E, Levashova T, Öhgren A, Sandkuhl K, Smirnov A, Tarassov V (2005) Configuration of dynamic SME supply chains based on ontologies. In: Proceedings of the 2nd International Conference on Industrial Applications of Holonic and Multi-Agent Systems (HoloMAS), Copenhagen, Denmark, August 2005, pp 246–256
10. KBSI (1994) IDEF5 method report. Available at: http://www.idef.com/Downloads.htm
11. Genesereth MR, Fikes RE (1992) Knowledge interchange format version 3.0 reference manual. Available at: http://www.upv.es/sma/teoria/sma/kqml_kif/kif.pdf
12. Gruber TR (1992) Ontolingua: a mechanism to support portable ontologies. Available at: http://citeseer.ist.psu.edu/gruber92ontolingua.html
13. Gruber TR (1993) A translation approach to portable ontology specifications. Knowl Acq 5:199–220
14. Corcho O, Fernández-López M, Gómez-pérez A (2003) Methodologies, tools and languages for building ontologies. Where is their meeting point? Data Knowl Eng 46:41–64
15. Farquhar A, Fikes R, Rice J (1997) The Ontolingua server: a tool for collaborative ontology construction. Int J of Human-Computer Studies 46:707–727
16. Huang GQ, Lau JSK, Mak KL (2003) The impacts of sharing production information on supply chain dynamics: a review of the literature. Int J Prod Res 41:1483–1517
17. OAGi (2006) OAGIS 9.0. Open Applications Group, Incorporated (OAGi). Available at: http://www.openapplications.org/downloads/oagidownloads.htm
18. OASIS (2004) Universal Business Language (UBL) 1.0. OASIS UBL Technical Committee. Available at: http://docs.oasis-open.org/ubl/cd-UBL-1.0/