A Semantic-Mediation Architecture for Interoperable Supply-chain Applications

Marko Vujasinovic[a*], Nenad Ivezic[a], Boonserm Kulvatunyou[a], Edward Barkmeyer[a], Michele Missikoff[b], Francesco Taglino[b], Zoran Marjanovic[c] and Igor Miletic[c]

[a]*Manufacturing Engineering Laboratory, National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD 20899, USA; { marko.vujasinovic, nenad.ivezic, boonserm.kulvatunyou, edward.barkmeyer } @nist.gov*

[b]*Laboratory for Enterprise Knowledge and Systems, IASI, Consiglio Nazionale delle Ricerche, Viale Manzoni 30, 00185 Rome, Italy; { missikof, taglino } @iasi.cnr.it*

[c]*Faculty of Organizational Sciences, University of Belgrade, Jove Ilica 158, 11000 Belgrade, Serbia; marjanovic.zoran@fon.bg.ac.yu, igor.miletic@brezasoftware.com*

Abstract. This paper presents a semantic-mediation architecture that enables standards-based interoperability between heterogeneous supply-chain applications. The architecture was implemented using a state-of-the-art semantic-mediation toolset for design-time and run-time integration tasks. The design-time tools supported a domain ontology definition, message annotations, message schema transformations, and reconciliation rules specifications. The run-time tools performed exchanges, transformations, and reconciliations of the messages. The architecture supports a supply-chain integration scenario where heterogeneous automotive manufacturing supply-chain applications exchange inventory information.

Keywords*;* semantic-mediation architecture; supply-chain interoperability; ontologies; standards-based interoperability

* Corresponding author. Email: marko.vujasinovic@nist.gov

1. Introduction

Today, manufacturing organizations are challenged to accomplish timely and accurate information exchange among the inventory-management applications and across the supply-chain tiers. As a response to this challenge, the Automotive Industry Action Group (AIAG 2008) initiated the Inventory Visibility and Interoperability (IV&I) project to establish standards for interoperable data exchange among the inventory visibility (IV) applications (IV&I AIAG 2007).

The IV applications use Internet-based technologies to enable near real-time material management by displaying trading-partner inventory data. Currently, the IV applications are not interoperable – they use proprietary electronic visibility models and message sets for inventory management.

Recently, the IV&I project focused on data exchange among manufacturing supply-chain applications to enable the so-called electronic Kanban (eKanban) business process. The eKanban business process regulates the flow of goods from the supplier to match actual usage by the customer, using standard material packages or *kanbans* to facilitate efficient material replenishment. The objective of the project is to provide standard IV&I eKanban messages that may be exchanged between IV tools. Figure 1 shows the target IV&I architecture. If the standard message set is implemented by the IV&I tools, the supplier will be in a position to select one IV&I tool to communicate with all of its customers, and that tool will communicate the information to the appropriate IV tool for each customer. Enabling suppliers to use only one IV&I application for all customer channels could result in the projected savings of $255 million annually across the automotive supply chain (Gould 2007).

The standards-based approach requires each tool provider to build new software to convert between its internal model and the standard exchange forms. This is error-prone and labor-intensive because it requires human agents to interpret the intent of specifications that use informal methods such as syntactic notations, diagrams and text to convey the data exchange requirements and their business meaning. This paper describes a novel, semantic-mediation architecture for overcoming some of the key interpretation problems when implementing to a message exchange standard. In the following, the paper presents essential details of the architecture and an investigation of representational support of the new architecture and an enabling toolset for the adopted automotive manufacturing supply-chain industrial scenario.

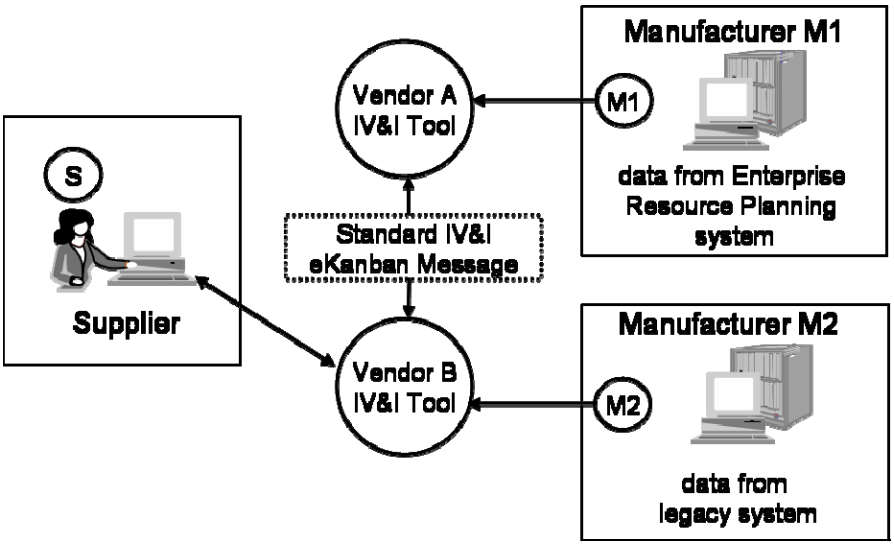

Figure 1 An example of the desired interoperability of IV&I Tools: Supplier S utilizes a single IV&I tool to exchange data with all its customers, either directly (M2) or via other IV&I applications (M1) that exchange standard IV&I eKanban messages

2. Semantic-mediation architecture

The proposed architecture for interoperable supply-chain application includes following advances over the standards-based approaches: (1) *formal* capture of the semantics concepts in the business domain; (2) *precise* semantics annotation of message interfaces and specification of semantics reconciliation rules between application and standard message interfaces; and, (3) *automated* and *consistent* standard-interface implementation through the reconciliation-rules execution. The important contribution of this architecture is the use of semantic-mediation techniques to achieve standards-based interchange. Activities supported by the architecture may be summarized as follows:

1) At the *Business Domain Ontology Definition* level, the domain concepts are formalized on the basis of the business process model and the data-exchange-requirements specification, to capture the intended meaning of the information exchanges. As a result, Reference Ontology (RO) is developed using an ontology authoring tool.

2) *At the Design-Time Annotation and Reconciliation Rules Specification* level, two independent annotation activities take place: (1) each application provider annotates his application interface (typically an XML schema) to relate each data item to the corresponding information concept in the RO; and (2) following adoption of the standard message schemas (also called Business Object Documents or BODs), the eKanban message specification team annotates the BODs in the same way. The annotations are done using a semantic- annotation tool. Next, each team uses the annotations to define the data reconciliation rules between message instances that

3) At the *Run-Time Reconciliation Execution* level, when the application is sending, a reconciliation engine executes a two-step conversion process using the appropriate reconciliation rule bases, translating message instances from the proprietary application data format to RO instances, and then translating the RO instances to the BOD format. When that application is receiving, the reconciliation engine translates from the BOD messages to the proprietary format using the appropriate rule bases. Effectively, the reconciliation rule execution engine implements the standard interface for the application. Ultimately, the BOD instances are sent over the Internet using standard web service (WS) calls.

A principal benefit of the new architecture is to provide a basis for objective, unambiguous interpretation of the interoperability artifacts. First, by representing the business domain concepts in an formal ontology, the meaning of the message elements, and their relationships over a suite of messages, may be established unambiguously. Next, it is possible for the software engineer to derive the actual reconciliation rules with greater clarity and consistency, and with greater computational support. Finally, the reconciliation rules may be executed directly at run-time, providing a consistent, traceable specification of the transformation of the message instances.

3. ATHENA toolset support for the semantic-mediation architecture

The proposed architecture is implemented with a semantic-mediation toolset developed within the EU Framework Programme 6 ATHENA research project (Athena 2007). The semantic-mediation architecture, as supported by the ATHENA toolset, is shown in Figure 2.

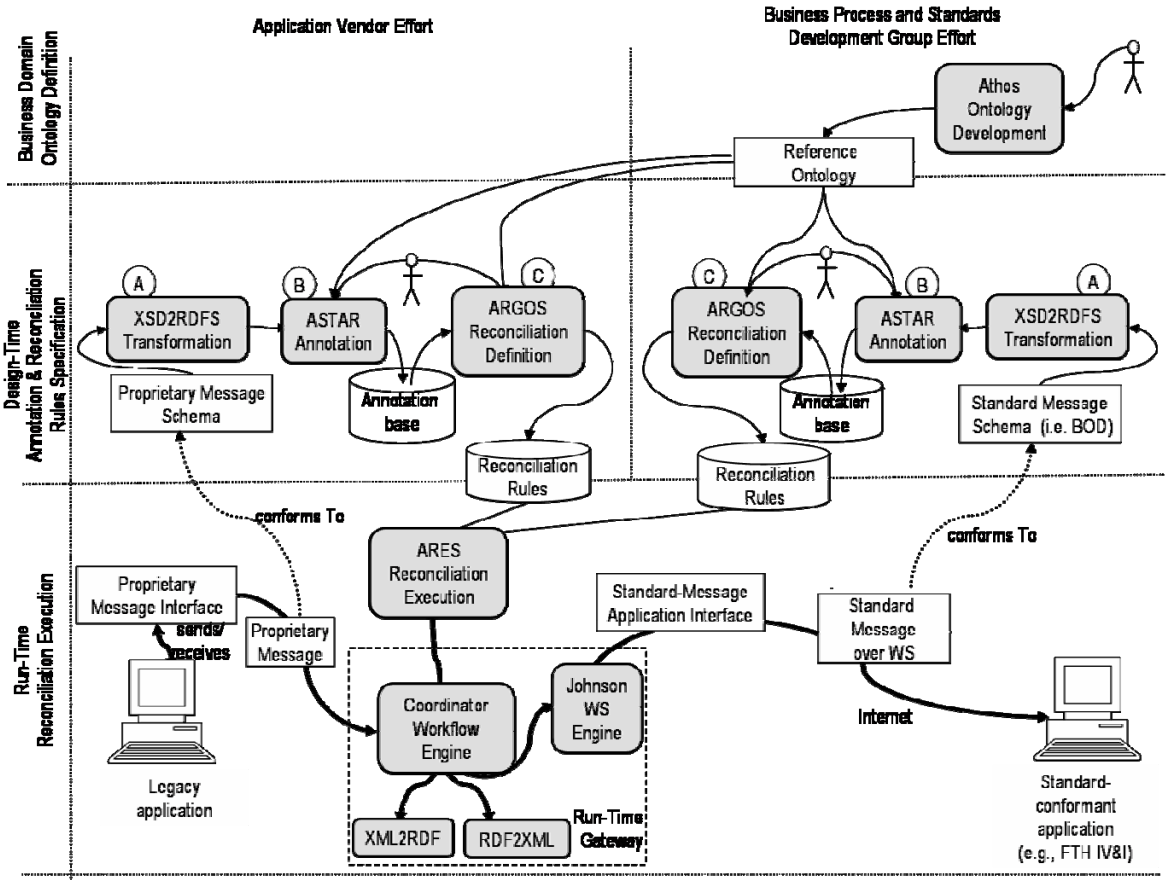

Figure 2 ATHENA-enabled Semantic-Mediation Architecture

During the **analysis phase**, the ATHOS ontology-management tool (Missikoff and Taglino 2006) was used to capture the RO. The RO was based on a specification of the eKanban business process and data requirements, interviews with industry experts, and eKanban BOD schemas adopted by the automotive industry.

At **design-time**, The XSD2RDFS transformer tool (Miletic *et.al*. 2007) was used to convert XML message schemas for automotive IV applications, and the standard eKanban BOD message schemas, to corresponding Resource Definition Framework (RDF) Schema message representations. Next, the ASTAR RDF Schema model annotation tool (Missikoff *et.al.* 2006) was used to define the semantic relationships between the data elements in the RDF-based message schemas and the business concepts in the RO. This is the key process in semantic mediation.

Then, the ARGOS reconciliation rules authoring tool (Coscia *et.al.* 2006) was used to create reconciliation rule bases that implemented conversion of the data elements defined by the RDF-based message schemas to and from RO instances. When combined, these rules effectively specified the data transformations between the proprietary messages and the industry standard BODs.

At **run-time**, the XML2RDF and RDF2XML transformer tools (Miletic *et.al*. 2007) transform XML message instances from and to RDF representations compatible with the transformed schemas. The ARES (Pondrelli *et.al.* 2006) run-time reconciliation engine executes the reconciliation rule bases defined at design-time on RDF-based message

instances representation. Together with the transformation steps, this tool converts messages between the proprietary forms and the standard forms. The Coordinator Tool was developed to orchestrate the run-time transformer tools, the ARES tool, and the Johnson (Vayssière *et.al.* 2006) WS execution engine, which enabled robust support for webservices over the Internet, so as to transform a proprietary message into a standard-conformant message.

4. An Experimental Scenario

The objective of the experimental scenario was to assess the proposed architecture and the supporting ATHENA toolset. Within this scenario, heterogeneous automotive manufacturing supply-chain applications exchange inventory visibility information in the form of standard eKanban messages. In the following, from the perspective of the application provider, the paper discusses the steps an IV application developer sees, and, in particular, details the design-time steps that are applied to an IV application message.

The scenario involved two independently developed IV applications (capable of sending and receiving only their proprietary versions of the eKanban AuthorizeKanban message) and an IV&I-conformant application capable of receiving the standard SyncShipmentSchedule message (Authorize Kanban BOD):

- The Apolon (Novicic *et.al.* 2007) open source IV application with an RDF Schema-based proprietary interface.
- A GM IV experimental enterprise application with an XML Schema-based proprietary interface.
- A Ford Test Harness (FTH) application with an XML Schema-based IV&I-conformant interface.

The Apolon application (running in Serbia) successfully exchanged a message with the FTH (running in Maryland, USA). Also, the GM application (running in Michigan, USA) successfully exchanged information with the FTH and Apolon. Both the GM and Apolon applications utilized the Coordinator gateway.

4.1. Reference Ontology

The eKanban Reference Ontology (Barkmeyer and Kulvatunyou 2007) formally captured the business concepts and relationships for the eKanban business process, as developed by the IV&I eKanban team of industry analysts.

A portion of the RO is shown in Figure 3. *SyncShipmentSchedule* is a *Message* sent by a *sender* (in this case, customer) to a *receiver* (in this case, supplier) to update a *ShipmentSchedule*. *ShipmentSchedule* is a *BusinessObjectDocument* that describes all of the shipments under some agreement.. Each *ScheduleLine* corresponds to an eKanban arrangement, called a *KanbanLoop*, for shipments of one *Item* from the *ShipFromParty* to the *ShipToParty*. The unit of shipment is a *Kanban*. In effect, the message authorizes shipment of one or more *Kanban*s to the customer site, by changing the *KanbanStatus* of those Kanbans.

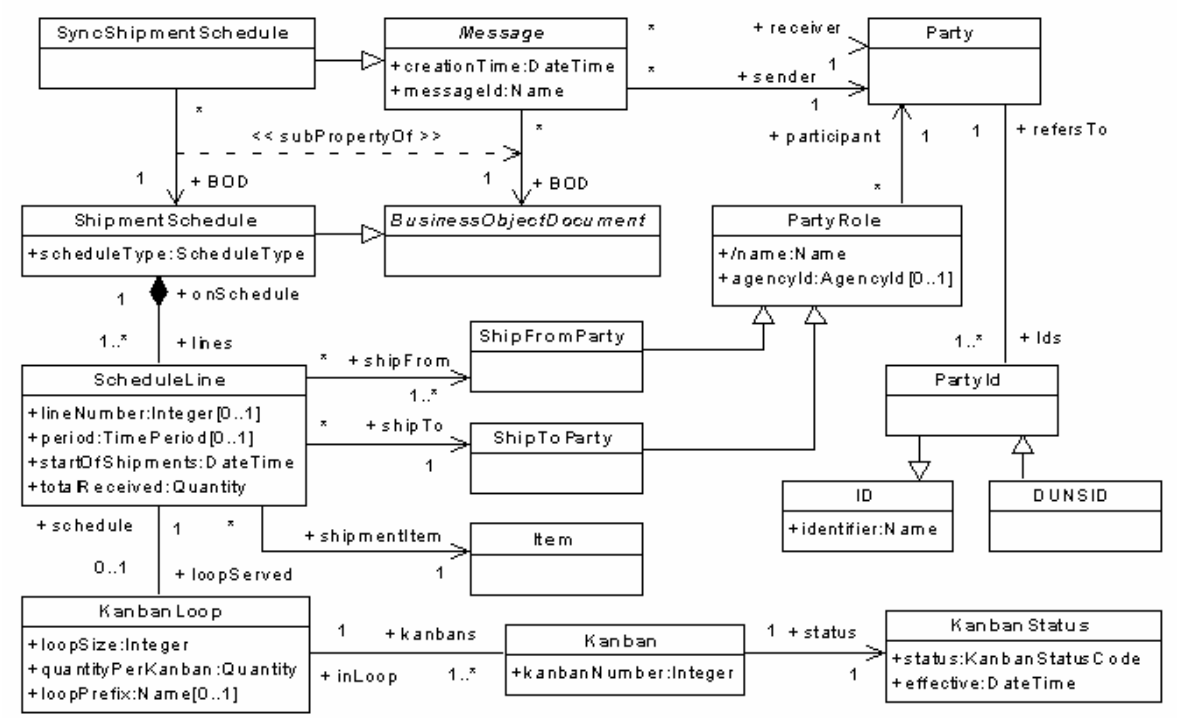

Figure 3 A portion of the IV&I e-Kanban Reference Ontology

4.2. Design Time

Before the run-time message exchange could be accomplished, the following steps were completed at design-time:

1) *Message schema transformation* (step A in Figure 2) of the standard *SyncShipmentSchedule* BOD and the GM-proprietary *AuthorizeKanban* message schema to corresponding RDF Schema-based representations.

2) Message schema *annotation* (step B in Figure 2) of the standard *SyncShipmentSchedule* BOD, and GM and Apolon *AuthorizeKanban* RDF Schemas using the ASTAR tool.

3) *Message reconciliation rules specification* (step C in Figure 2) using the ARGOS tool to create both (a) *forward* rules to reconcile data from the *SyncShipmentSchedule* BOD, and GM and Apolon *AuthorizeKanban* RDF instances to RO instances, and (b) *backward* rules to reconcile data from the RO instances to the *SyncShipmentSchedule* BOD, and Apolon *AuthorizeKanban* RDF instances.

4.2.1. Message Schema Transformation

The XSD2RDFS Tool enabled the business applications that currently use XML Schema-based interfaces to use the ATHENA RDF-based semantic mediation tools. For the application provider, this step was automatic and transparent. The XSD2RDFS tool transforms any given XML Schema into an RDF Schema.

Listing 1 shows the XSD2RDFS transformation for a portion of the GM-proprietary *AuthorizeKanban* message. XML Schema elements (such are *gmSyncShipmentSchedule*,

sender, and *DUNS*, shown at the left side) are transformed into corresponding RDF Schema classes (*gmSyncShipmentSchedule*, *gmSyncShipmentSchedule_sender*, and *gmSyncShipmentSchedule_sender_DUNS*, shown at the right side). For each simple XML Schema element, a corresponding RDF property is created (e.g., *gmSyncShipmentSchedule_sender_DUNS_sValue* for the *DUNS* element). Each parent-child structural relation between XML Schema elements is transformed into a RDF property (e.g., *gmSyncShipmentSchedule_sender_DUNS_PROP* RDF property for the parent-child relation between the sender and DUNS XML Schema elements).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:tns="http://gm.com/gmSyncShipmentSchedule/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  name="gmSyncShipmentSchedule"
  targetNamespace=
  "http://gm.com/gmSyncShipmentSchedule/">
 <xsd:element name="gmSyncShipmentSchedule">
  <xsd:complexType>
   <xsd:sequence>
    ...
    <xsd:element name="sender"
        type="tns:partner"/>
    ....
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
<xsd:complexType name="partner">
  <xsd:sequence>
   <xsd:element name="DUNS" type="xsd:string"/>
  </xsd:sequence>
 </xsd:complexType>
 .....
</xsd:schema>
```

xsd2rdfs

```
<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" xmlns="http://www.nist-athena-ivi.com/rdfs#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
<rdfs:Class rdf:about="http://www.nist-athena-ivi.com/rdfs#gmSyncShipmentSchedule">
</rdfs:Class>
...
<rdfs:Class rdf:about=" http://gm.com/gmSyncShipmentSchedule/rdfs#gmSyncShipmentSchedule_sender">
</rdfs:Class>
<rdfs:Class rdf:about=" http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender_DUNS">
</rdfs:Class>
 ....
<rdf:Property rdf:about=" http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender_DUNS_sValue">
  <rdfs:domain rdf:resource= http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender_DUNS"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</rdf:Property>
...
<rdf:Property rdf:about=" http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender_PROP">
  <rdfs:domain rdf:resource=" http://gm.com/gmSyncShipmentSchedule /rdfs#gmSyncShipmentSchedule"/>
  <rdfs:range rdf:resource="http:// http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender"/>
</rdf:Property>
<rdf:Property rdf:about=" http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender_DUNS_PROP">
  <rdfs:domain rdf:resource=" http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender"/>
  <rdfs:range rdf:resource=" http://gm.com/gmSyncShipmentSchedule
/rdfs#gmSyncShipmentSchedule_sender_DUNS"/>
</rdf:Property>
...
</rdf:RDF>
```

Listing 1. A portion of an XSD2RDFS transformation

4.2.2. Message Schema Annotation

Semantic annotation is a means for defining the relationships between the elements in a message and the business meaning of those elements as captured in the reference ontology. So the critical drivers are the *definitions* of the terms used in the ontology and the *documentation* and usage guidelines for the message elements.

In the message schema annotation step, an expert on each software application studies the RO and assigns the appropriate RO semantics to each data element in the application interface, using the ASTAR tool. In addition, an expert on the IV&I standard performed this process for the standard BOD schema.

The ASTAR annotation method consists of four phases: (1) the terminological semantic annotation (TSA), (2) path semantic annotation (PSA), (3) the simple semantic annotation (SSA), and, (4) full semantic annotation (FSA) phase.

The TSA phase deals with terminological (*naming*) mismatches which occur when different terms are used to denote same concept. In this phase, the ASTAR compares the terms used in the application interfaces with the terminology of the RO concepts, and automatically detects terminological similarities among them. Then, correspondences between the terms are established manually, by identifying the RO concept whose

definition most closely matches the intent of the message element as indicated by its documentation (while the naming helps in finding these, it is not always a reliable indicator). The idea of the PSA phase is to relate structural paths in the message RDF schemas to relationship paths through the semantic network in the RO. The PSA phase reveals *structuring* mismatches which occur when different structures are used in organizing the same information set. The SSA phase deals with *content* mismatches, which are mostly about choices of representation for information units, such as named values, time intervals, and addresses. The corresponding, specialized, data transformations needed at run-time are denoted by attaching (usually pre-defined) *abstract operators* to these semantic associations. The FSA phase generates an OWL DL ontology (Web Ontology Language) for the semantic associations, and allows the user to note the actual relationship between the semantic concepts: equivalence, subsumption in either direction, or just overlap.

Semantic annotation also reveals real semantic mismatches, called *coverage* mismatches. Such a mismatch occurs when a concept in the application message has no match in the RO. The corresponding information will be lost on outbound messages and missing from inbound messages. When that information is optional and seldom used, it may not be a problem; but, when it is important to the application, it means that the standard is inadequate. The reverse case, in which the RO has a concept not used in the application message, is usually harmless – the RO may well support many different messages, only some of which use any given concept. But if it is a mandatory property of a required object, the application may not be suitable for the use envisaged in the standard.

Table 1 shows some of the semantic relationships captured by annotations of the GM and Apolon schemas, including the abstract operators (AO) for the content mismatches. A *coverage* mismatch was found, between the GM schema and the RO. The GM schema has no equivalent for the KanbanStatus field. This is a critical field, but the GM usage is to imply the status value by the type of the message itself. So the semantic link is between the message type and the KanbanStatus, and it was possible to overcome this mismatch with a special reconciliation rule.

Table 1 Four-phases of ASTAR semantic annotation

Phase	App.	RDF Schema elements	Corresponding RO element	Identified Mismatch
TSA	GM	gmSyncShipmentSchedule	ShipmentSchedule_BusinessObject	*naming*
		gmSyncShipmentSchedule_sender	SenderParty_BusinessObject	
		gmSyncShipmentSchedule_sender_DUNS	Id_BusinessObject	
		gmSyncShipmentSchedule_sender_PROP	relTo_ SyncShipmentSchedule_Message_SenderParty	
		gmSyncShipmentSchedule_sender_DUNS_PROP	relTo_ SyncShipmentSchedule_Message_SenderParty_PartyId	
		gmSyncShipmentSchedule_sender_DUNS_sValue	has_Id_identifier	
		(not defined)	has_KanbanStatusCode	*coverge*
	Apolon	ShipmentSchedule	ShipmentSchedule_BusinessObject	*naming*
		forDocumentItem	relTo_ScheduleLine_ScheduleLine_loopServed_KanbanLoop	
		DocumentItem	KanbanLoop_BusinessObject	
		forKanban	relTo_KanbanLoop_hasKanban_Kanban	
		Kanban	Kanban_BusinessObject	
		kanbanStatus	has_KanbanStatusCode	
PSA		**RDF Schema path**	**Corresponding RO path**	
	GM	gmSyncShipmentSchedule. gmSyncShipmentSchedule_sender_PROP. gmSyncShipmentSchedule_sender. gmSyncShipmentSchedule_sender_DUNS_PROP. gmSyncShipmentSchedule_sender_DUNS. gmSyncShipmentSchedule_sender_DUNS_ sValue : STRING	ShipmentSchedule_BusinessObject. relTo_ShipmentSchedule_message_SyncShipmentSchedule. SyncShipmentSchedule_Message. relTo_ SyncShipmentSchedule_messageSender_SenderParty. SenderParty_BusinessObject. relTo_ SenderParty__partyIds_PartyId. PartyId_BusinessObject. has_Id_identified:STRING	*structuring*

		RDF Schema path	AO	Corresponding RO path	
SSA	GM	gmSyncShipmentSchedule. gmSyncShipmentSchedule_sender_PROP. gmSyncShipmentSchedule_sender. gmSyncShipmentSchedule_sender_DUNS_PROP. gmSyncShipmentSchedule_sender_DUNS. gmSyncShipmentSchedule_sender_DUNS_ sValue: STRING	=	ShipmentSchedule_BusinessObject. relTo_ShipmentSchedule_message_SyncShipmentSchedule. SyncShipmentSchedule_Message. relTo_SyncShipmentSchedule_messageSender_SenderParty. SenderParty_BusinessObject. relTo_SenderParty__partyIds_PartyId. PartyId_BusinessObject.has_Id_identified:STRING	
	Apolon	ShipmentSchedule. forDocumentItem. DocumentItem. forKanban. Kanban.kanbanStatus:STRING	φ*	ShipmentSchedule_BusinessObject. relTo_ShipmentSchedule_ShipmentSchedule_lines_ScheduleLine. ScheduleLine_BusinessObject. relTo_ScheduleLine_ScheduleLine_loopServed_KanbanLoop. KanbanLoop_BusinessObject. relTo_KanbanLoop_hasKanban_Kanban. Kanban_BusinessObject. relTo_Kanban_Kanban_status_KanbanStatus. KanbanStatus_BusinessObject. has_KanbanStatusCode:ENUMERATION [authorized, empty, full, shipped]	*content*

		RDF Schema path	Relation	OWL DL Expression
FSA	GM	gmSyncShipmentSchedule. gmSyncShipmentSchedule_sender_PROP. gmSyncShipmentSchedule_sender. gmSyncShipmentSchedule_sender_DUNS_PROP. gmSyncShipmentSchedule_sender_DUNS. gmSyncShipmentSchedule_sender_DUNS_sValue:STRING	=	STRING ∩(∃ *inverseOf* has_Id_identified.(PartyId_BusinessObject ∩(∃ *inverseOf* relTo_SenderParty_partyIds_PartyId.(SenderParty_BusinessObject ∩(∃ *inverseOf* relTo_SyncShipmentSchedule_messageSender_SenderParty. (SyncShipmentSchedule_Message ∩(∃ *inverseOf* relTo_ShipmentSchedule_message_SyncShipmentSchedule. (ShipmentSchedule_BusinessObject)))))))

* 'φ' - a unary abstract operator, defined on a single path from the RO. ASTAR also defines a binary abstract operator '+', to be used to compose RO paths

4.2.3. Message Reconciliation Rules Specification

The message reconciliation rules specification was based on the semantic model defined in the RO and the semantic annotations. The software application expert specifies the reconciliation rules – the instructions for copying and converting data elements, using

the ARGOS tool. The specification was performed semi-automatically, by selecting, from a list provided by the tool, the appropriate rule templates and conversion functions for each semantic match. The tool then created the declarative run-time rule by substituting the matching schema path and RO path into the template.

The rule generation process is performed twice for each application interface: once to generate the forward rules and once to generate the backward rules.

Table 2 shows examples of map, set value and map table rule templates. The GM sender-DUNS path was reconciled with RO senderParty-ID_identifier path using the map rule template, which simply copies the data value. The set value rule was used to set the RO KanbanStatusCode value to the constant Authorized, overcoming the coverage mismatch. The map table rule template was used to create the rules for reconciling the content mismatch.

Table 2 Examples of defined reconciliation rules

	GM RDF Schema path	RO path	Rule
Forward	gmSyncShipmentSchedule. gmSyncShipmentSchedule_sender_PROP. gmSyncShipmentSchedule_sender. gmSyncShipmentSchedule_sender_DUNS_ PROP. gmSyncShipmentSchedule_sender_DUNS. gmSyncShipmentSchedule_sender_DUNS_ sValue:STRING	ShipmentSchedule_BusinessObject. relTo_ShipmentSchedule_message_SyncShipmentSchedule. SyncShipmentSchedule_Message. relTo_ SyncShipmentSchedule_messageSender_SenderParty. SenderParty_BusinessObject. relTo_SenderParty_partyIds_PartyId. PartyId_BusinessObject.has_Id_identified:STRING	map
	(not defined)	ShipmentSchedule_BusinessObject. relTo_ShipmentSchedule_ShipmentSchedule_lines_ ScheduleLine. ScheduleLine_BusinessObject. relTo_ScheduleLine_ScheduleLine_loopServed_ KanbanLoop. KanbanLoop_BusinessObject. relTo_KanbanLoop_hasKanban_Kanban. Kanban_BusinessObject. relTo_Kanban_Kanban_status_KanbanStatus. KanbanStatus_BusinessObject.has_KanbanStatusCode: ENUMERTION [authorized, empty, full, shipped]	set value
	RO path	Apolon RDF Schema path	
Backward	ShipmentSchedule_BusinessObject. relTo_ShipmentSchedule_ShipmentSchedule_ lines_ScheduleLine. ScheduleLine_BusinessObject. relTo_ScheduleLine_ScheduleLine_loopServed_ KanbanLoop. KanbanLoop_BusinessObject. relTo_KanbanLoop_hasKanban_Kanban. Kanban_BusinessObject. relTo_Kanban_Kanban_status_KanbanStatus. KanbanStatus_BusinessObject. has_KanbanStatusCode:ENUMERATION [authorized, empty, full, shipped]	ShipmentSchedule. forDocumentItem. DocumentItem. forKanban. Kanban. kanbanStatus:STRING	map table

4.3. Run-Time

A Coordinator run-time engine converted messages, in both directions, from the proprietary to the standard IV&I form and from the standard form to the proprietary form. Each message transformation involved two rulesets: for the sending application, the forward ruleset for the application and the backward ruleset for the standard; and for the receiving application the forward ruleset for the standard and the backward ruleset for the application. Each application had its own appropriately configured Coordinator instance.

Inside the Coordinator, several message transformations and reconciliations were necessary, as described below, but these steps are virtually transparent to the software application developer – the application sends and receives messages in its proprietary format. And the Coordinator provided the robust webservice interface needed for supply-chain transactions that use the Internet – webservice addressing, reliability and security.

The XML2RDF service accepts the XML message and generates RDF-based version of message corresponding to the RDF Schema produced by the XSD2RDFS transformation. This was needed for the GM interface, and for the standard, but not for the Apolon interface.

Next, the ARES rule execution engine is used to apply a forward and backward reconciliation rulesets. Finally, where was needed, the RDF2XML tool converted the RDF version of the target message to its XML form – a valid instance of the XML schema for the interface.

5. Experimental Results and Findings

As a proof of concept, the experimental results showed that it was possible to do the conversions between application-specific messages and the IV&I standard messages using the semantic-mediation approach. The scenario offered evidence that the ATHENA toolset provided adequate representational support for the selected data exchange scenario.

The experimental results also gave some insight into the complexity of the reconciliation problem for an intentionally simple use case. Note that the mappings for the standard IV&I interface are publicly available; an application expert need only be concerned about the mappings for his application.

Table 3 summarizes the statistics for the semantic matches between the RO and the three interface schemas – the IV&I standard and the two proprietary interfaces – for the single SyncShipmentSchedule message. The ASTAR tool provided computational support for finding lexical-terminological similarity among the names used, but each of these annotations required an expert on the specific interface to identify the valid correspondences. An expert is needed because the semantic correspondences depend on the meaning, not on the term used. All message schemas concepts were annotated successfully.

Table 3. Statistics for the semantic matches between the RO and the three interface schemas

Interface Schema	GM	Apolon	eKanban BOD	Total
Number of concepts	33	60	76	169
Concept matches established	48	60	76	184
Path matches established	12	27	29	68
Content mismatches identified	0	1	1	2
Coverage mismatches identified	1	0	9	10

Table 4 shows the number and types of reconciliation rules created for each interface schema. All identified mismatches were successfully reconciled. The *map* template, which is used for one-to-one mappings between matching elements with no data conversion, is used for 85% of the runtime reconciliation rules. And the remaining 15% are identified by

the ASTAR annotations as special cases. Because the ARGOS tool is semi-automatic, the human expert must select every rule used, but the instantiations of the *map* template could be done automatically, reducing the time for this phase up to 85%.

Table 4. Number and types of reconciliation rules created for each interface schema

Interface Schema / Reconciliation rule template	GM	Apolon	eKanban BOD	Total	Total (%)
map	12	27	29	69	85%
map table	0	1	1	2	2%
set value	1	0	9	10	13%
Total	13	28	39	80	100%

While the basic functionality of the tools was shown to be appropriate, the semantic annotation and reconciliation tasks proved to be challenging for the current toolset, and for the users of those tools. In particular, the path-matching mechanism used in ASTAR was barely adequate and unnecessarily time-consuming. The semantic-mediation approach has shown adequate representational capabilities, but significant rework of the tooling is needed for industrial use. The experimental investigation highlighted three directions for future work that may significantly advance the existing toolsets to support the methodology:

- *Industrial message model.* Develop and use a message platform-independent model that faithfully represents the concepts in webservices and XML Schema, rather than an RDF Schema-based model. This makes the design-time activities easier for the human expert and improves capabilities for the manipulation of messages in run-time rules.
- *Efficient annotation support.* For path matching, do not generate any path through the RO a priori. In general, allow the engineer to steer the path development rather than presenting him with a long list of possible paths. Generate the full candidate path when it can be deduced unambiguously from the terminological annotations.
- *Automated reconciliation support.* Use the semantic annotations to derive most reconciliation rules automatically. Involve the application expert only in the special cases.

6. Survey of Related Semantic-Mediation Approaches

Mediators were first introduced as parts of distributed information systems in the late eighties. Since then, many application areas have adapted mediators as the approach for overcoming heterogeneity of applications and data sources. Presently, the ontologies, as a reference semantic and shared vocabulary of specific business domains, are becoming a primary interface to the data mediation.

There are several alternative architectural approaches for ontology-based semantic mediation, depending on the type and number of ontologies involved, and on a reconciliation approach applied. In a semantic-mediation, ontologies may appear either as individual local ontologies (LO), which represent local conceptual models of data sources,

or as reference ontologies, which represent common standardized conceptual model of data in some particular business domain.

Hameed (2004) suggested three models for ontology-based mediation systems, depending on between which ontologies reconciliation takes place: (1) Local ontologies (LOs) pair-wise reconciliation model (i.e., *any-to-any model*) that reconciles LOs among themselves as needed. (2) Single reference ontology (RO) model (i.e., *any-to-one model*) where RO serves as an "interlingua" that any instance of LO may be translated to and vice-versa. (3) A hybrid model that employs multiple ROs, forming clusters of interrelated ontologies and where reconciliation is specified between the LOs and an RO in one cluster, and among the ROs of different clusters.

Further distinction is based on the reconciliation approach applied – merging or mapping. Merging unifies two or more ontologies with overlapping parts into a single ontology that includes all information from the sources. Mapping builds the mapping statements that define relationships between concepts of different ontologies and mapping rules that specify transformation from one ontology (source) to the other ontology (target).

Anicic (2006) and OntoMerge (2008) demonstrated an *any-to-any merging model*. Anicic demonstrated how local OWL ontologies (representing local message schemas), which are, however, based on the same generalized terminology, are merged together. He showed that the source ontology individuals may be classified into the target ontology individuals using automated reasoners (i.e., individual classification computation). Anicic demonstrated the semantic integration of automotive supply-chain applications that use AIAG and STAR (2008) message specifications standards, both based on the OAG standard (OAG 2008). OntoMerge introduced bridging axioms between two related ontologies, which, however, are not required to be based on the same generalized terminology. OntoMerge implemented translation using an inference engine that computed inferences on merged ontology axioms and bridging axioms.

The any-to-any model based on merging, however, has shortcomings: (1) increased size and complexity of the merged ontology when merging many LOs (especially when bridging axioms are used), and (2) incapability of non-trivial relationships computation (e.g., split and join relationships) with automated reasoners.

Artemis project demonstrated an *any-to-any mapping model* by devising executable mapping rules among two LOs (Bicer *et.al* 2005). Artemis defined the message exchange framework for clinical data-exchange among healthcare institutes and demonstrated mediation between two local OWL ontologies that represented EDI-based HL7.2 and XML-based HL7.3 messages. Artemis introduced and used the OWLmt tool, which is of the same purpose as the ARGOS tool, for establishing the mappings between OWL ontologies and mapping rules execution. OWLmt tool provides a so-called Mapping Schema Definition ontology (encoded in OWL) for establishing the mappings between two OWL ontologies by predefined mapping pattern axioms from the Mapping Schema Definition. Mapping patterns are further interpreted by the OWLmt engine for handling the mapping process.

The shortcoming in the any-to-any mapping model is a large number of mapping rules, which may go up to n(n-1) mappings if n LOs are mapped to each other in a bidirectional way. The large number of mapping rules leads to the time-consuming creation and error-prone maintenance of the mappings.

On the other side, the any-to-one mapping model significantly reduces the number of mappings. The AMIS and Harmonize projects demonstrated the *any-to-one mapping model*. AMIS provided a reference model for semantics-based integration infrastructure for supply chains (Libel *et.al.* 2004). AMIS considers RO as "interaction ontology formation" that captures the business concerns and concepts, and engineering interaction concerns also. The Harmonize project (Dell'Erbaa *et.al.* 2002) demonstrated semantic-mediation in a tourism harmonization network to allow participating tourist organizations to keep the proprietary XML-based application interfaces and still be able to exchange message data. Harmonize defined a reference Harmonization Ontology and RDF interchange format. Harmonize didn't require a standard-based XML message exchange. Harmonize used the MAFRA tool (Maedche *et.al.* 2002) for establishing the mappings between RDFS-based ontologies and mapping rules execution. MAFRA provides a proprietary mapping ontology, called Semantic Bridge Meta-Ontology (SBO), for establishing the mappings between ontologies by bridging axioms, which is an idea similar to the OWLmt. MAFRA maps RDF ontologies, and transforms source ontology instances to the target ontology instances by evaluating the defined bridge axioms and executing pre-defined functions assigned to each bridge axiom.

The merging is not a desirable reconciliation approach for the any-to-one model, as the actual merge of a LO and RO modifies RO, which is an unwanted behavior for the any-to-one model and RO idea. Practically, the merging could be applied in the any-to-one model only by establishing the 'virtual' merge axioms among an LO and a RO, but, however, such 'virtual' merge axioms are nothing more than mappings.

The work presented in this paper is aligned with an any-to-one mapping model and it mostly follows from the AMIS project. AMIS reference model is adapted to the standard-based interoperability; however, the novel semantic-mediation architecture proposed in this paper sees RO as a capture of business concerns and concepts only, which is different than the AMIS view of the RO. Furthermore, RO has to be developed in a top-down manner - after designing the business process, gathering the business requirements, and specifying standard message exchange - and business applications ultimately exchange standard-conformant messages, which are additional requirements comparing the other approaches mentioned above.

The proposed architecture provides semantic mediation for the web-service based supply-chain applications where the business data are exchanged as SOAP (2007) wrapped XML messages. Similar semantic-mediation architectures for the web-service based supply-chain applications were demonstrated by Ye and Yang (2007), and Oh and Yee (2007). Ye and Yang introduced a general Supply Chain Ontology (SCO) (Ye *et.al.* 2008) that captured concepts and relationships common to the supply chain management, and also used a rule-based approach to map semantically similar terminologies between the SCO and local application ontologies. The mapping rules in their work are represented in SWRL rule description language (Horrocks et.al., 2004). Oh and Yee demonstrated rule-based mapping between two local RDFS ontologies in a web-services based manufacturing applications environment, however, with no presence of reference ontology.

Ontology-based semantic mediation can also be used for web-service based supply-chain applications where web services do not exchange XML messages but where they are exposed as remote services operations and where operations' parameters (defined as primitive or user data-types) carry business data. Shen et.al (2007) categorized the

heterogeneity of such network web services composition in manufacturing in several conflict types and proposed three types of ontologies that are used together to resolve conflicts and to achieve interoperable network web services composition: 1) the web services Domain Knowledge Ontology that specifies a business process in the terms of process's name, participating parties, related services, input and output parameters, 2) Domain Industry Knowledge Ontology that uniformly defines semantic terms used by service providers and service customers to describe web services, services' operations and parameters, and 3) Mapping Knowledge Ontology that defines positive (forward) mapping functions and inverse (backward) mapping functions.

Semantic web services also have become a key technology for enabling the workflow integration of supply chains (McIlraith et.al., 2001). The semantic web services may provide automated web service discovery, execution, composition and interoperation in the manufacturing supply-chain environments (Kulvatunyou *et.al.* 2005, Yang *et.al.* 2005). The semantic-mediation architecture proposed in this paper is, however, concerned with the semantic interoperability in the traditional web-service based supply-chain environments.

The work presented in this paper is mainly focused on bringing the technologies together to demonstrate supply-chain applications interoperability in a novel and effective way, by adopting an approach of semantic-mediation through a standard ontology and applying the semantic-mediation for achieving the standards-based interoperability between supply-chain applications.

7. Conclusions

The proposed semantic-mediation architecture was demonstrated (1) to be able to handle multiple message formats from independent, heterogeneous manufacturing supply-chain applications and (2) to provide required transformations and model-based reconciliations to result in interoperable message exchanges over the Internet.

The ATHENA-developed toolset, although in its prototype version, successfully enabled this architecture. The toolset provided support for reference-ontology development, semantic-annotation capabilities with semi-automated assistance based on lexical similarities, and support for the definition of executable reconciliation rule sets with semi-automated assistance based on the formally captured semantic annotation.

The proposed architecture may positively impact situations where new partners are frequently added to the supply-chain collaboration. A new tool provider may achieve significant reduction of time when specifying and implementing a standard interface. The development of the reference ontology and the associated artifacts for the standard itself is a one-time task for the standards community. Those are available to every new provider. So each new provider need only do the mappings for his existing interfaces.

The new architecture has the potential to move standards development from a syntax-based approach to a semantics-based approach, and from informal expressions of business intent to formal and machine-understandable ones. This architecture moves several implementation tasks associated with standards compliance to a model-based approach and thereby from hard-coded, subjective and inconsistent implementations to clear, documented semi-automated and consistent implementations. In addition, companies that are successful in using the advanced technologies may significantly reduce time-to-market for conforming software products.

Disclaimer

Certain commercial software products are identified in this paper. These products were used only for demonstration purposes. This use does not imply approval or endorsement by NIST, nor does it imply these products are necessarily the best available for the purpose.

References

Anicic, N. et.al, 2006. *An Architecture for Semantic Enterprise Application Integration Standards*, Proc. Int'l Conf. Interoperability of Enterprise Software and Applications, Springer-Verlag, 25-34.

Automotive Industry Action Group (AIAG) Web Page, 2008. [Online]. Available at: http://www.aiag.org/, [accessed August 2008]

IV&I Automotive Industry Action Group (AIAG) Web page, 2007. *Inventory Visibility & Interoperability Oversight Committee* [online] Available at: http://www.aiag.org/staticcontent/committees/index.cfm?committeeid=IVOS [accessed July 2007]

ATHENA Integrated Project Web Site, 2007. [online] Available at: www.athena-ip.org [accessed July 2007]

Barkmeyer, E., Kulvatunyou, B., 2007. *An Ontology for the e-Kanban Business Process* [online], NIST Internal Report 7404, Available from: http://www.mel.nist.gov/msidlibrary/doc/NISTIR_7404.pdf [accessed July 2007]

Bicer, V. et.al., 2005. *Artemis Message Exchange Framework: Semantic Interoperability of Exchanged Messages in the Healthcare Domain*, SIGMOD Record 34(3), 71-76.

Dell'Erbaa, M. et.al., 2002. *HARMONISE: A Solution for Data Interoperability*, Proc. of IFIP I3E 2002 Conf., 114-127.

Gould, L., 2007. *What's New In Automotive Supply Chains?* [online], Auto Field Guide, Available from: http://www.autofieldguide.com/articles/020407.html [accessed July 2007]

Libes, D., et. al., 2004. *The AMIS Approach to Systems Integration* [online], NIST Internal Report 7101, Available from: http://www.mel.nist.gov/msidlibrary/doc/nistir7101.pdf [accessed July 2007]

Hameed, A. et.al., 2004. *Ontology Reconciliation*, Handbook on ontologies, Springer-Verlag, 231–250.

Horrocks, I., Patel-Schneider, et.al., 2004. *SWRL: a semantic web rule language combining OWL and RuleML*, WWW Consortium Member Submission, [online]. Avaiable at http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/ . [accessed September 2008]

Maedche, A., et.al., 2002. *MAFRA— A MApping FRAmework for Distributed Ontologies*, In Proceedings of EKAW 2002, LNCS 2473, Springer- Verlag, 235–250.

McIlraith, S.A., Son, T.C. et.al., 2001. *Semantic Web services*, Intelligent Systems, IEEE 16(2), 46-53

Miletic, I., Vujasinovic, M., et al., 2007. Enabling Semantic Mediation for Business Applications: XML-RDF, RDF-XML, and XSD-RDFS Transformation, *Proc. Int'l Conf. Interoperability of Enterprise Software and Applications*, Springer-Verlag, 483-494

Missikoff, M., and Taglino, F., 2006. *D.A3.2 - Updated version of the Ontology Authoring and Management System with semantic search functions* [online], Deliverable of the Athena

project, February 2006; Available from: http://leks-pub.iasi.cnr.it/Athos or www.athena-ip.org [accessed July 2007]
Missikoff, M., et. al., 2006. *D.A3.3 - Semantic Annotation language and tool for information and Business Processes* [online], Deliverable of the Athena project, February 2006; Available from: http://leks-pub.iasi.cnr.it/Astar or www.athena-ip.org [accessed July 2007]
Coscia, E., et. al., 2006. *D.A3.4 - A system for reconciliation rules specification, storage and management* [online], Deliverable of the Athena project, Available from: https://services.txt.it or www.athena-ip.org [accessed July 2007]
Pondrelli, L., et. al., 2006. *D.A3.5 - A reconciliation and mediation engine, capable to efficiently process semantic mediation and reconciliation rules* [online], Deliverable of the Athena project, Available from: www.athena-ip.org [accessed July 2007]
Simple Object Access Protocol (SOAP) 1.1, 2007. [online] Available at http://www.w3.org/TR/soap/, [accessed September 2008]
Shen, D., Yu., G., et.al., 2007. *Resolving heterogeneity of Web service composition in network manufacturing based on ontology*, International Journal of Computer Integrated Manufacturing, 20(2), 222-233
Standards for Technology in Automotive Retail (STAR) Web Site, 2008. [Online]. Available at: http://www.starstandard.org/, [accesed August 2008]
Vayssière, J., et.al., 2006. *Rapid Prototyping for Service-Oriented Architectures* [online], 2nd Workshop on Web Services Interoperability at IESA Conf., Bordeaux 22nd - 24th March, 2006; Available from: http://www.athena-ip.org/dmdocuments/pu/Rapid_Prototyping_for_SOAs.pdf [accessed July 2007]
Novicic, I., et al., 2007. *A Case Study in Business Application Development Using Open Source and Semantic Web Technologies*, Proc. Int'l Conf. Interoperability of Enterprise Software and Applications, Springer-Verlag, 721-724.
Open Applications Group (OAG) Web Site, 2008. [online] Available from: http://www.openapplications.org/, (current November 2004)
OntoMerge, 2007. *OntoMerge: Ontology Translation by Merging Ontologies*, [online] Available from: www.cs.yale.edu/~dvm/daml/ontology-translation.htmltranslation.html [accessed July 2007]
Oh. S., and Yee, S., 2008. *Manufacturing interoperability using a semantic mediation*, International Journal Advanced Manufacturing Technology, 39(1-2), 199–210
Yang, Z., Gay, R., et.al., 2005. *Automating integration of manufacturing systems and services: a semantic Web services approach*, Industrial Electronics Society IECON 2005. 31st Annual Conference of IEEE 6-10 Nov. 2005
Ye, Y., Yang, D., et. al., 2007. *An ontology-based architecture for implementing semantic integration of supply chain management*, International Journal of Computer Integrated Manufacturing, 21(1), 1-18
Ye, Y., Yang, D., et. al., 2008. *Ontology-based semantic models for supply chain management*, International Journal of Advanced Manufacturing Technology, 37(11-12), 1250-1260
Kulvatunyou, B., Cho, H., et.al., 2005. *A Semantic Web Service Framework to Intelligent Distributed Manufacturing*, International Journal of Knowledge-based and Intelligent Engineering Systems, 9(2), 107-127