Proceedings of EOMAS 2009

Enterprise Modeling in a Service Oriented Architecture

Peter Bollen

Maastricht University
Faculty of Economics and Business Administration
Department of Organization & Strategy

P.O. Box 616, 6200 MD Maastricht, The Netherlands
Tel: 31-43-3883715
Fax: 31-43-3884893
e-mail: p.bollen@os.unimaas.nl

Abstract: Service-oriented computing (SOC) is a new paradigm that allows organizations to tailor their business processes, in such a way that efficiency and effectiveness goals will be achieved by outsourcing (parts of) business processes to web-based service-providers. In order to find the computing service-providers that provide the organizations with the biggest benefits, it is paramount that the service-requesting organization (SRO) has a precise description of the service it wants to have delivered by the service delivering organization (SDO). In this paper we will illustrate how enterprises that play the SDO and SRO roles can be conceptually integrated by creating conceptual models that share the definitions of the business processes within the service oriented architecture (SOA) framework.

Keywords: service-oriented architecture, SOA, fact-orientation, ORM, object-role modeling

1 Introduction

In the service-oriented architecture (SOA) paradigm, a service requesting organization (SRO) basically outsources one or more organizational activities or even complete business processes to one or more service delivering organizations (SDOs). The way this is done currently, is that the SRO 'outsources' a given computing service to a 'third-party' SDO for a relative long period of time (3 months, a year). The selection and contracting activities are performed by organizational actors, i.e. managers responsible for the business processes in which the service(s) is (are) contained. Most of the current SDO's provide 'internet substitutes' [1] for functions that used to be performed by an (integrated) SRO's enterprise system, implying that the SRO's that use these process services are shielded from the intrinsic complexities of these 'substituted' functionalities [2, 3].

Current approaches for web services, have limitations on a semantic and ontological level (among others) [4]. The problem with current approaches is that they cannot handle the semantic and ontological complexities caused by flexible participants having flexible cooperation processes. Enterprise integration between participants (i.e. broker, SROs and SDOs) can only be achieved if the conceptual schema of the content, e.g. it's ontology can be expressed formally and explicitly [5].

In most business organizations the function that is responsible for information and knowledge management will have some kind of repository, schema or knowledge map that (ideally) defines the information objects (business repository or business ontology) and the semantic relationships between these business concepts (conceptual schema or a DDL of some sort). At best (large) companies have a business glossary in which business concepts are defined precisely. When it comes to processes we must conclude that at best descriptions of procedural knowledge might be documented in some type of DFD's or other process description logic (e.g. BPMN [6]). In most practical situations, however, the process logic is embedded in software code, and an explicit semantic description is lacking.

The application of the service-oriented paradigm that will lead to the most benefits for the SRO will be embedded in a semantic-web environment in which the 'outsourcing' decision in principle, can be made in real-time every time a service is requested [7]. This real-time level of decision making implies that the service-processes that are requested should be defined in such a way that the negotiation, contracting and execution of the service can take place in 'run-time' without 'design time' human intervention. In fact-oriented terminology we can say that a process is a fact-generating activity [8].

There currently exist a number of generic standards for expressing web ontologies, e.g. OWL [9] and for modeling web service ontologies, e.g. WSML [10] and web services execution, e.g. WFSL 1.0 [11] and BPEL4WS [12].

In this paper we will apply the fact-oriented conceptual modeling language (e.g. as documented in several variations in [13-17]), that will enable businesses to define their platform independent models for their service-oriented requirements and that will allow for an 'automatic' transformation from the platform independent model (PIM) to the platform specific model (PSM) levels [18, 19]. A well-written text-book on ORM is [20].

2 Related work

2.1 Related work on conceptual modeling approaches

In compliance with the emerging OMG standard on business rules: SBVR [21], we will consider a family of conceptual modeling approaches that traditionally have been used for the specification of relational databases [16] but have evolved into business rule modeling languages [22-26] and languages for subject matter modeling [27-30]. We will refer to this family of approaches as fact-oriented modeling approaches. The latest extensions of the fact-oriented modeling approaches cater for the declarative modeling of business processes [31-33] and business events [32, 34]. In an earlier article the application of fact-orientation for applying web ontologies has been discussed [35]

The (extended) fact-oriented approach structures verbalizable knowledge into the following elements [27]:

1. Knowledge domain sentences
2. Concept definitions and naming conventions for concepts used in domain sentences
3. Fact types
4. Fact type readings for the fact types
5. Population state (transition) constraints for the knowledge domain
6. Derivation rules that specify *how* specific domain sentences can be derived from other domain sentences.
7. Exchange rules that specify *what* fact instances can be inserted, updated or deleted.
8. Event rules that specify *when* a fact is derived from other facts or when (a) fact (s) must be inserted, updated or deleted.

The combined Knowledge Reference Model (KRM) consisting of elements 1 through 8 of the above captures a complete description of a domain- or application's conceptual schema, including the domain- or application ontology (elements 2, 3 and 4).

A legend of the ORM-(I) notation used in this article is provided in the following. The 'role-based' ORM notation makes it easy to define static constraints on the data structure and it enables the modeler to populate ORM schemas with example sentence instances for constraint validation purposes. In ORM (and other fact oriented approaches) the fact construct is used for encoding all semantic connections between entities. Figure 1 summarizes the symbols in the ORM modeling language that we have used in this paper.

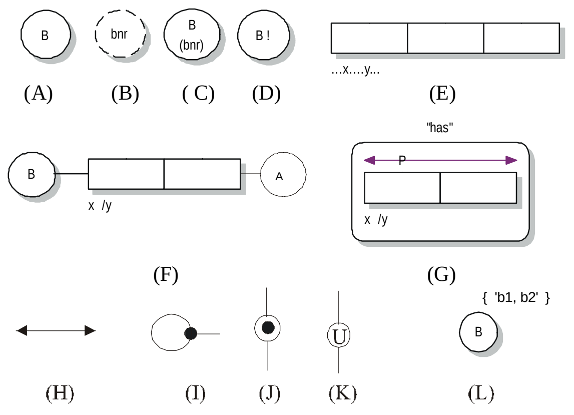


Fig. 1. Main symbols in Object-Role Modeling (ORM).

Atomic *entities* (figure 1A) or *data values* (figure 1B) are expressed in ORM as simple (hyphenated) circles. Instances of an entity type furthermore can exist independently (e.g. they are not enforced to participate in any relationship), which is shown by adding an exclamation point after the entity type's name (figure 1D). *Simple* reference schemes in ORM are abbreviated by putting the *value type* or *label type* in parenthesis beneath the name of the entity type (figure 1C). Semantic connections between entities are depicted as combinations of boxes (figure 1E) and are called *facts* or *fact types* in ORM. Each box represents a role and must be connected to either an *entity type*, a *value type* or *a nested object type* (see figure 1F). A fact type can consist of one or more roles. The number of roles in a fact type is called the fact type arity. The semantics of the fact type are put in the *fact predicate* (this is the text string *…x…y…* in figure 1E). A *nested object type* (see figure 1G) is a non-atomic entity type that is connected to a fact type that specifies what the constituting entity types and/or values types are for the nested object type. Figures 1H through 1L illustrate the diagramming conventions for a number of *static population constraint(s) (types)* in ORM. A double-arrowed line (figure 1H) that covers one or more 'boxes' of a fact type is the symbol for an *internal uniqueness constraint*. The symbol in figure 1K stands for an *external uniqueness constraint.* A(n) uniqueness constraint restricts the number of identical instances of a role combination 'under' the uniqueness constraint to *one*. A *mandatory role constraint* (figure 1I) can be added to a role. It specifies that each possible instance of such an object type must play that designated role at *all* times. A *disjunctive mandatory role constraint* (figure 1J) is defined on two or more roles and specifies that each possible instance of the object type connected to these roles must *at least* play *one* of these roles at *any* time. In figure 1L an example of a value constraint is given that enforces that each instance of the object type B either has the value *b1* or *b2*

2.2 Related work on process modeling

The research on the *process-oriented perspective* in the information systems research community was prominent in the late seventies and has resulted in numerous process-oriented information systems development methodologies like SADT [36] and SA/SD [37] that are still in use [38]. Although these development methodologies were process-oriented they also contained modeling constructs for the data-oriented perspective. In the eighties a number of system development methodologies were proposed that covered both the data-oriented, process-oriented and behaviour-oriented perspectives [39-41]. In the nineties a research school on 'workflow management' (see for a good literature review [42]). Around that time the business process reengineering 'paradigm' [43, 44] in combination with the increasing popularity of Enterprise Resource Planning packages (e.g. SAP, see [45]), lead to the development of domain-oriented analysis methods [46] of which the ARIS-based Business Process Modeling [47, 48] and BML [49] are good examples. A recent standard for expressing business processes is the business process modeling notation specification BPMN [6].

In addition to the body of work on process modeling in the above, there has been some considerable amount of research on process modeling in the fact-oriented community as well [8, 33, 50-52] [53, 54]. In figure 2 we have given the necessary documents for the three perspectives in the IFIP-CRIS architecture (based on [55]). In this paper we will focus on the documents in the middle column of figure 1 that represent the process-oriented perspective. In [54] the union of the meta process model and meta behaviour model is put into the architecure as 'program grammar'.

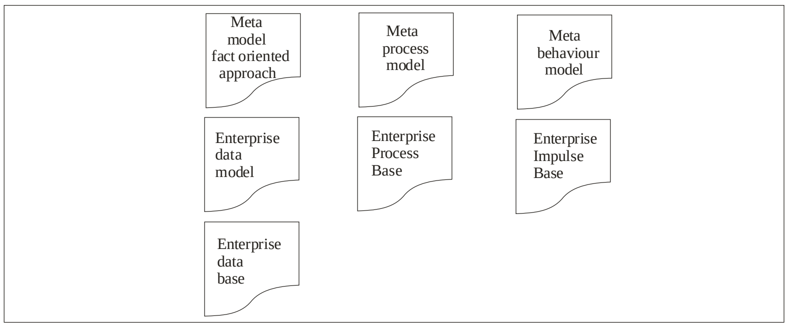


Fig. 2. Documents in the data-, process- and behaviour-oriented perspectives

In this paper we will focus on the 'enterprise process base' from the architecture in figure 2 and we will built upon the 5 document architecture as presented in [54].

2.3 Related work on service-oriented architecture

In [56] a SOA (service oriented architecture is provided). The basic elements from this service-oriented approach to distributed software design is given in figure 3.

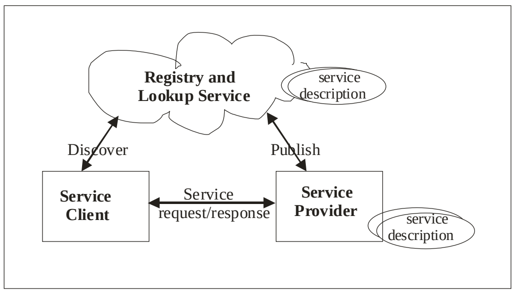


Fig.3. SOA architecture as given in [56] and [19]

In the SOA from figure 3, service delivering organizations (SDOs) or service providers use the registry service (or broker [5] or service repository [57]) to publish their identity and a description of services that they provide. When a service requesting organization (SRO), service requestor [57] or service client, needs a service, it queries the lookup service (service discovery [58]) which will initiate the communication between SRO and SDO to establish a commitment regarding the service delivery [58].

2.4 Related work on the modeling of service-oriented enterprise architectures

The specific paradigm that is center of the discussion in this article is the structuring of businesses according to services that are provided or needed [59]. Research that integrates conceptual (meta)modeling and the SOA paradigm mostly relates to the meta level, especially the meta process model according to the 7-document architecture in figure 2, e.g. as in Piprani et al. [60] and the ontology part of the meta model, e.g. Terlouw [61]. Other researchers investigate, how business rule management can be extended to the service-oriented paradigm [62]. We will take the SOA architecture from figure 3 as a basis for our further application of conceptual modeling on the SOA domain. We will thereby, explicitly distinguish three universes of discourse (UoD's): the client (SRO), the service provider (SDO) and the broker (or registry and look-up service).

3 Fact-oriented conceptual modeling of the SRO

We will extend the current modeling capabilities of the fact-oriented approach with modeling constructs for the modeling of business services in the context of the service-oriented paradigm by extending the concepts definitions and derivation/exchange rule modeling constructs to cater for 'business services' that can be provided by either the SRO itself or by one or more (external) SDO(s).

In order to use the semantic web for selecting and contracting SDO's for any business function that needs to be outsourced, business organizations need conceptual modeling tools that define these functions or (parts of) business processes. The commonly used process modeling approaches lack the capabilities to be used for this purpose [33]. In this section we will extend the fact-oriented conceptual modeling approach to cater for the definition of business functions (or parts of business processes) during design time in such a way that in a semantic web environment in which SRO and SDO's can interchange their domain ontologies and thereby in run-time can decide which of the relevant SDO's will be partner to deliver the requested service for a given business transaction.

We will present the elements from the fact-oriented knowledge reference model (KRM) and see how they can be applied in the situation in which SDOs are involved in (interorganizational) business processes. We will use as a running example for the UoD of the SRO, the (fictitious) ABC company, and focus on the carrier selection process for customer shipments.

3.1 The SRO UoD: The ABC company's carrier selection business process

ABC is a business that operates a number of 'brick-and-mortar' stores. Although the company does have an internet retail-website, it sometimes receives order request for deliveries via mail, e-mail or fax, outside the sales region it serves and in some cases even outside the country it operates in, and sometimes it receives 'overseas' order requests. Especially for the latter order category, ABC can make an additional profit by shipping the order using the cheapest carrier at any given point in time. The shipping fee, they charge to their customer is a constant fee. The customer has the choice between standard shipping and express shipping. The ABC company, has a logistics department in which 1 person is responsible for the shipment of continental and overseas orders. Since this person, has also other logistics responsibilities, he/she can not afford to spend too much time trying to search for the best transportation deals. It might be beneficial for ABC, to 'outsource' the carrier selection process to a third-party, in this case a service delivery organization (SDO).

We will start the presentation of our fact-oriented model by providing a list of structured concept definitions [23, 63]. This list of structured concept definitions, should facilitate the comprehension of knowledge domain sentences and comprise the business domain ontology [64].

3.2 List of concept definitions

Table 1. List of concept definitions for SRO

Concept	**Definition**
Organization	A business entity that delivers services and/or goods to customers and/or other business entities.
Organization code	A name from the *organization code* name class that can be used to identify an [organization] among the set of [organization]s.
Service requesting organization (SRO)	An [organization] that potentially can request a service from a third party organization.
Service delivery organization (SDO)	A [service delivery organization] that delivers a service to a [SRO]
Cargo	A product shipment from a [SRO] to a customer
Dimension	Size of [cargo] as length* width * height
Dimension code	A name from the *dimension code* name class that can be used to identify a [dimension] among the set of [dimension]s
Size	Depicts the extent in meters of any of the three elements of a [dimension]
# of meters	A name from the two-decimal number name class that can be used to identify a [size] among the set of [size]s.
Volume	Depicts the extent in cubic meters of a three- [dimension]-al package
# of cubic meters	A name from the two-decimal number name class that can be used to identify a [volume] among the set of [volume]s.

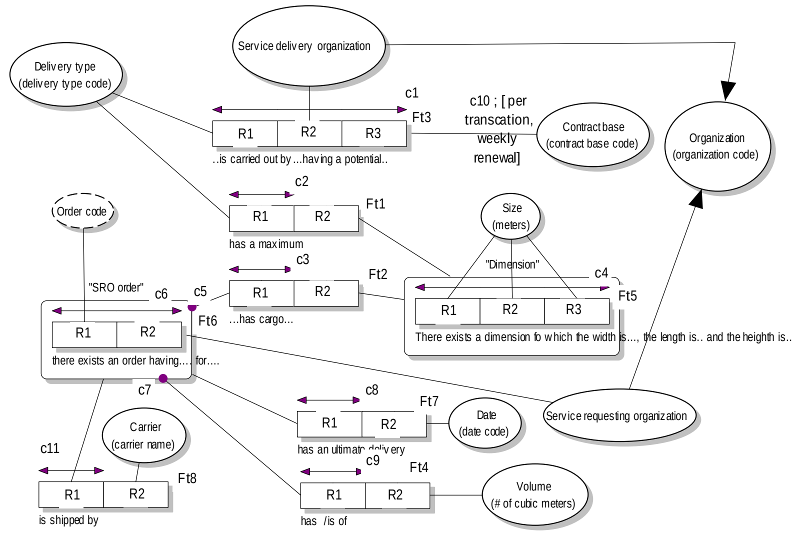
Delivery type	A generally agreed upon type of delivery by a [service requesting organization] and a service registry organization or broker that is characterized by a maximum [dimension]
Delivery type code	A name from the delivery type code name class that can be used to identify a [delivery type] among the set of [delivery type]s.
Contract base	Type of commitment between a [service delivery organization] and a [SRO]
Contract base code	A name from the *contract base code* name class that can be used to identify a [contract base] among the set of [contract base]s.
'Per transaction' contract base	A specific value for a [contract base code] that means that a contract between a [SDO] and a [SRO] change per transaction on the discretion of a [SRO].
'Weekly renewal' contract base	A specific value for a [contract base code] that means that a contract between a [SDO] and a [SRO] can change per week on the discretion of a [SRO].
is shipped by	Depicts that a package is tranpsorted from an originator's door to a receiver's door
Order	A request to ship a package to a customer
Order code	A name from the *order code* name class that can be used to identify a [order code] among the set of [order code]s.
Carrier	A third party logistics organization that ships packages for an [order] from a [SRO] to a client of the [SRO]
Carrier name	A name from the *carrier name* name class that can be used to identify a [carrier] among the set of [carriers]s that exist in the world.
Date	Depicts a specific day
Date code	A name from the *date code* name class that can be used to identify a [date] among the set of [date]s.

3.3 Fact types and fact type readings, Population state (transition) constraints for the knowledge domain

In case a standard between the SDO's and SRO's and service broker has been implemented, in which it is agreed upon that*: for any (predefined) delivery type at most one maximum dimension can exist*, we can show this as a uniqueness constraint of fact type Ft1 that covers the role R1. A further formalization of the allowed communication within the SRO's UoD's is the convention that *a given order by a given SRO must have exactly one dimension*. The latter business rule is encoded in the ORM fact type model in figure 3 as a uniqueness constraint spanning role R1 of fact type Ft2 in combination with a mandatory role on the nested entity type SRO-order. Finally, in role R3 of fact type Ft3 we can define a value constraint in which the allowed and enumerable set of values can be listed.

3.4 Derivation rules

In addition to the business rules that can be expressed as population state constraints, we can add business rules that can derive ‘new’ fact instances from ‘old’ fact instances. An example of such a derivation rule can be applied for fact type *Ft4*. We assume that *a volume is the multiplication of the three dimensions* figures that are modeled in fact type *Ft2/Ft5*. This derivation rule can be modeled as a derivation rule in figure 4 in which formula: Ft4.R2 = Ft5.r1*Ft5.r2*Ft5.r3 is contained.


Service delivery organization
Delivery type (delivery type code)
c1
Ft3
R1 R2 R3
..is carried out by ...having a potential..
c10 ; [per transcation, weekly renewal]
Contract base (contract base code)
Organization (organization code)
Order code
c2
Ft1
R1 R2
has a maximum
Size (meters)
"Dimension"
c3
Ft2
...has cargo...
c4
Ft5
There exists a dimension fo which the width is...., the length is.. and the heighth is..
"SRO order"
c6
c5
Ft6
there exists an order having.... for....
c7
c8
Ft7
has an ultimate delivery
Date (date code)
Service requesting organization
Carrier (carrier name)
c11
Ft8
is shipped by
c9
Ft4
has /is of
Volume (# of cubic meters)

Define Order has Volume (cubic meters)**as** Order has cargo Dimension **and**
There exist a dimension for which the width is Size_1 and the length is Size_2 and the heighth is Size_3 **and**
Volume= $\text{Size}_1 * \text{Size}_2 * \text{Size}_3$

Fig.4. Complete conceptual schema for SRO (in combination with table 1)

We note that in a service oriented architecture, derivation rules play an important role because SRO’s ‘outsource’ the execution and management of these rules to SDO’s. It’s therefore, paramount to incorporate the definition of these derivation- and exchange processes into the list of concept definitions. If we inspect the conceptual schema for our example SRO we see that the process *calculate volume* is implemented within the sphere of influence of the organization itself. The process is made explicit in the form of derivation rule: *Define order has Volume (cubic meters)*, that is listed at the bottom of figure 4. The process determine carrier for order, however is outsourced to some SDO.

Process: Calculate Volume	A process that has a a result: a rough indicator of the cubic [volume] of a package which is determined by multiplying its width, heigth and length. **<Create(s) instance(s) of Ft4>**
Process: Add order	A transaction in which the [order] and the [dimension] and [delivery date] of the [order] are added to the information system. **<Create(s) instance(s) of Ft2 and Ft7>**
Process: Determine carrier for order	This process leads to the selection of a specific [SDO] for the shipment of an [order] under the best possible conditions for [delivery time] and [shipment price] **<Create(s) instance(s) of Ft8>**

4 Fact-oriented conceptual modeling of the SDO

In this section we will look at the Universe of Discourse of a web-service that provides carrier selection services for SRO’s. One of the main processes within this UoD’s is the up-to-date acquisition of carrier data regarding latest offers, in terms of shipment conditions, and prices for each delivery type and possibly delivery (sub)-types depending upon each individual carrier. This web-service organization has as objective to match SRO’s with carriers normally for a small fee per transaction. We will see that ontological commitments need to be established between SRO’s and SDO’s on a ‘design’-time level. This means that key concepts for web-based service transactions will be harmonized (as can be checked for example in the list of concept definitions in tables 1 and 2, for the concept *delivery type* and *carrier*). On the other hand, promotional concepts and other rating schemes can be introduced on the fly, at any time by a carrier. For many of these promotional campaigns and or new tariff schemes, it will not be feasible to establish ontology harmonization between the SDO and these carriers at all times. To cater for this, we need modeling constructs that allow us to deal with the runtime changes in domain concepts as used by SDO’s in their carrier selection processes on behalf of their SRO customers. We will show now in our example list of definitions and conceptual schema for the UoD of the SDO can be modeled for these short-term runtime definitions of domain concepts. We note that a ‘snaphot’ of delivery types for every carrier that that is considered by a carrier-selection SDO will be modeled as a populations of fact type Ft1 in the conceptual schema of the carrier selection SDO in figure 5. We now see that the carrier selection broker service not only provides the best deal for a service requesting organization, but also performs the role of ‘ontological harmonizer’ between the SRO’s and the carriers by introducing and defining the concepts of *local delivery type* and *carrier delivery type*.

In table 2 we have provided the extended list of concept definitions for this example UoD of a service delivery organization in which the definitions of the fact generating processes are incorporated.

Table 2. List of concept definitions for SDO

Concept	**Definition**
Carrier	A third party logistics organization that ships packages for an [order] from a [SRO] to a client of the [SRO]
Carrier name	A name from the *carrier name* name class that can be used to identify a [carrier] among the set of [carriers]s that exist in the world.
Local delivery type	A label to refer to a specific type of service provided by a specific [carrier]
Carrier delivery type	A [local delivery type] that is offered by a [carrier]
Period length in days	A period or slice in time having a duration
Natural number	A name from the *natural number* name class that can be used to identify a [period length in days] among the set of [period length in days].
Money amount	A specific quantity of money
Dollars	A name from the *dollar* name class that can be used to identify a [money amount] among the set of [money amount]s.
Promotional price	A price that is charged per kg for a delivery service during a number of [week]s in a promotional period
Standard price	A price that is charged in a [week] for which no [promotional price] is charged
Maximum dimension	The maximum [size] for length * the maximum [size] for width * the maximum [size] for height of an [order] for which a given [delivery type] is still valid
Maximum delivery period	The maximum value for [Period length in days] it takes to deliver a package to a client of a [SRO]
Year	A period or slice in time consisting of 365 days according to the Roman calendar
a.d.	A name from the *a.d.* name class that can be used to identify a [year] among the set of [years] in the roman calendar.
Week	A period or slice of time consisting of 7 days.
Weekcode	A name that can be used to identify a [week] within a given [year]
Process: Classify service offering	A process that has a result a classification for a [local delivery type] offered by a [carrier] in terms of an instance [delivery type] that has been defined by a [SRO] and [SDO]. **<Create(s) instance(s) of Ft108>**
Process: Add service offering delivery length	A process that has a result that a maximum delivery length for a [carrier delivery type] is entered into the information base. **<Create(s) instance(s) of Ft106>**
Process: Add service offering standard price	A process that has a result that a [standard price] for a [carrier delivery type] is entered into the information base **<Create(s) instance(s) of Ft101>**
Process: Add service offering promotional price	A process that has a result that a [promotional price] during one or more [weeks] for a [carrier delivery type] is entered into the information base **<Create(s) instance(s) of Ft102 and Ft107>**

In figure 5 we have given the complete conceptual schema for the example carrier selection process within the UoD of the SDO.

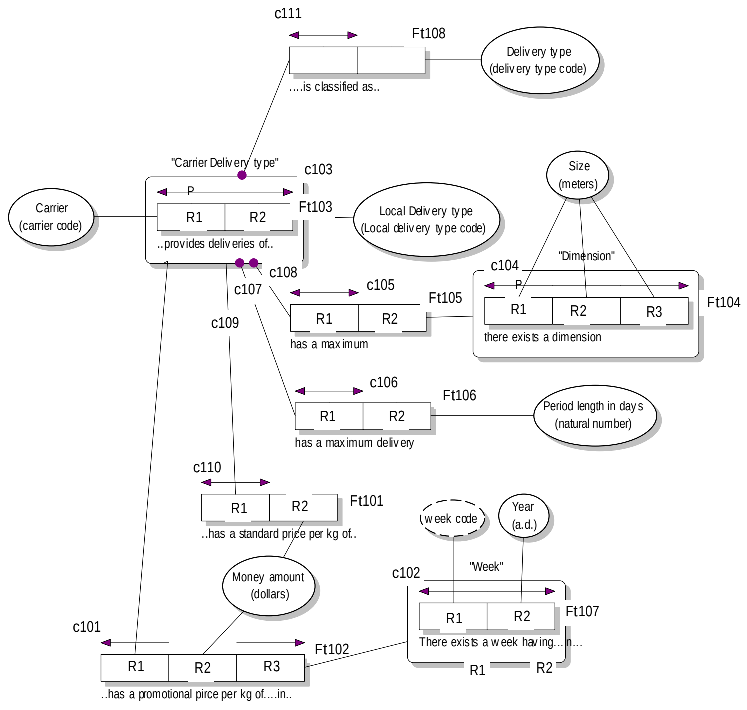


Fig.5. Conceptual schema for SDO (in combination with table 2)

So, if we inspect the conceptual schema for our fictitious example SDO in figure 4, we can say that one of the 'core business processes' for the SDO, is the establishing, ontological harmonization, in 'run-time' this means that the SDO will populate fact type Ft108 in figure 5, by continuously scanning for recent service offerings provided by existing and new carriers. This business process mainly scans and interprets these service offerings, and as a result will 'label' these offerings and subsequently classify them, in the terminology, that was established between the SRO's and SDO's, via a broker or registry service. In the running example of this article, we have limited ourselves to only depict a few relevant fact types that will be used in practice. In a real-life conceptual schema 100's of fact types might actually be used in the communication, between SDO, SRO and registry service.

5 Conclusion

In line with semantic web developments, the conceptual schema needs a communication part that contains 'definition' instances to be shared with the potential agents in order for them to be able to communicate effectively and efficiently with a ('web-based') business application. Adding these semantic definitions of business processes is a requirement for achieving enterprise integration. This will significantly increase the quality and ease-of-use of such a (web-based) application, since it has established a semantic bridge with the potential external users, allowing them to communicate in a direct way with the business application, by preventing semantic ambiguities from occurring in the first place. Another advantage of applying fact-orientation for capturing an application or a (relatively complex) domain's ontology is in its flexibility to use it even to model communication between agents in which (explicit) ontological harmonization at a type or schema level is not possible or desirable. By adding 'run-time' concepts as populations of (typed) concepts for which an ontological harmonization already has been established.

Another advantage of using (extended) fact-oriented modeling languages is that a business organization is not forced to remodel the application or domain ontology every time a new 'implementation' standard has been defined. Business organizations can capitalize on the fact-oriented conceptual modeling investment, for the foreseeable future by applying the appropriate mappings between a fact-oriented application ontology and the implementation standard of the time.

References

1. Siau, K. and Y. Tian, *Supply chains integration: architecture and enabling technologies.* Journal of Computer Information Systems, 2004. **45**(Spring): p. 67-72.
2. Estrem, A., *An evaluation framework for deploying web services in the next generation manufacturing enterprises.* RCIM, 2003. **19**: p. 509-519.
3. Baina, K., K. Benali, and C. Godart, *Discobole: a service architecture for interconnecting workflow processes.* Computers in Industry, 2006. **57**.
4. Shen, W., et al., *An agent-based service-oriented integration architecture for collaborative intelligent manufacturing.* RCIM, 2007. **23**: p. 315-325.
5. Yue, P., et al., *Semantics-based automatic composition of geospatial web service chains.* Computers & Geosciences, 2007. **33**: p. 649-665.
6. OMG, *Business process modelling notation (BPMN) specification*. 2007, OMG.
7. Menascé, D., H. Ruan, and H. Gomaa, *QoS management in service-oriented architectures.* Performance Evaluation, 2007. **64**: p. 646-663.
8. Bollen, P., *Conceptual process configurations in enterprise knowledge management systems*, in *Applied computing 2006*. 2006, ACM: Dijon, France.
9. Bechhofer, S., et al., *OWL web ontology language reference*, in *W3C*. 2004, W3C. p. 62.
10. Bruijn, J.d., et al., *WSML working draft 14 march 2005*, J.d. Bruijn, Editor. 2005,
11. Leymann, F., *Web Service Flow Language*. 2001, IBM.
12. Andrews, T., et al., *Business Process Execution Language for Web Services*. 2003, BEA systems.
13. Verheijen, G. and J. van Bekkum. *NIAM: An Information Analysis method*. in *IFIP TC-8 CRIS-I conference*. 1982: North-Holland, Amsterdam.
14. Bakema, G.P., J.P. Zwart, and H. van der Lek, *Fully communication oriented NIAM*, in *NIAM-ISDM 1994 Conference*, G. Nijssen and J. Sharp, Editors. 1994: Albuquerque NM.

15. Nijssen, G. and T. Halpin, *Conceptual schema and relational database design: A fact based approach*. 1989, Englewood Cliffs: Prentice-Hall.
16. Halpin, T., *Information Modeling and Relational Databases; from conceptual analysis to logical design*. 2001, San Francisco, California: Morgan Kaufmann.
17. Lemmens, I., M. Nijssen, and G. Nijssen, *A NIAM 2007 conceptual analysis of the ISO and OMG MOF four layer metadata architectures*, in *OTM 2007/ ORM 2007*. 2007, Springer: Vilamoura, Algarve, Portugal.
18. OMG, *MDA guide version 1.0.1*, J. Miller and J. Mukerji, Editors. 2003, OMG.
19. Jardim-Goncalves, R., A. Grilo, and A. Steiger-Garcao, *Challenging the interoperability between computers in industry with MDA and SOA*. Computers in Industry, 2006. **57**: p. 679-689.
20. Halpin, T. and T. Morgan, *Information Modeling and Relational Databases; from conceptual analysis to logical design*2nd ed. 2008, San-Francisco: Morgan-Kaufman.
21. OMG, *Semantics of Business Vocabulary and Business Rules (SBVR), first interim specification*. 2006. p. 392.
22. Nijssen, G., *Semantics for Business: a process to specify the most important business rule in SVBR*. Business Rules Journal, 2007. **8**(12).
23. Nijssen, G., *SBVR: Semantics for Business*. Business Rules Journal, 2007. **8**(10).
24. Nijssen, G., *Kenniskunde 1A*. Vol. PNA Publishing Heerlen. 2001.
25. Halpin, T., *Business Rule Modality*, in *Proc. EMMSAD'06: 11th Int. IFIP WG8.1 Workshop on Exploring Modeling Methods in Systems Analysis and Design*. 2006.
26. Halpin, T., *A fact-oriented approach to business rules*, in *ER 2000*. 2000.
27. Nijssen, G. and R. Bijlsma. *A conceptual structure of knowledge as a basis for instructional designs.* . in *The 6th IEEE international conference on Advanced Learning Technologies, ICALT 2006*. 2006. Kerkrade, the Netherlands.
28. Bollen, P. *Using Fact-Orientation for Instructional Design*. in *ORM 2006 Workshop*. 2006. Montpellier, France: Springer Verlag.
29. Vos, J., *Is there fact orientation life preceding requirements ?*, in *OTM 2007/ ORM 2007*. 2007, Springer: Vilamoura, Algarve, Portugal.
30. Nijssen, G. and P. Bollen, *Universal Learning: a science and methodology for education and training*, in *EDINEB: the case of problem-based learning*. 1995, Kluwer: Maastricht.
31. Bollen, P. *Fact-Oriented Business Service Modeling*. in *EMSSAD '07*. 2007. Trontheim, Norway.
32. Bollen, P., *Fact-oriented Business Rule Modeling in the Event Perspective*, in *CAISE 2007*. 2007: Trondheim, Norway.
33. Morgan, T., *Business Process Modeling and ORM*, in *OTM 2007/ORM 2007*. 2007, Springer: Vilamoura, Algarve, Portugal.
34. Bollen, P., *Fact-oriented modeling in the data-, process- and event perspectives*, in *OTM 2007, ORM 2007*. 2007, Springer: Vilamoura, Algarve, Portugal.
35. Bollen, P., *Natural Language Modeling for Business Application Semantics*. Journal of Information Science and Technology, 2005. **2**(3): p. 18-48.
36. Connor, M., *Structured analysis and design technique*. 1980: Waltham.
37. Yourdon, E. and L. Constantine, *Structured design*. 1979: Prentice-Hall.
38. Shen, H., et al., *Integration of business modeling methods for enterprise information systems analysis and user requirements gathering*. Computers in Industry, 2004. **54**: p. 307-323.
39. Flynn, D. and R. Warhurst, *An empirical study of the validation process within requirements determination*. Information Systems Journal, 1994. **4**: p. 185 - 212.

40. Olive, A., *Dades- a methodology for specification and design of information systems design and management*, in *Information systems design methodologies* 1982, N-Holland.
41. Rolland, C. and C. Richard. *The REMORA methodology for Information Systems Design and Management*. in *information systems design methodologies.* 1982: North-Holland.
42. Salimifard, K. and M. Wright, *Theory and Methodology: Petri net-based modelling of workflow systems: an overview*. Eur. J.of Operations Research, 2001. **134**: p. 664-676.
43. Davenport, T. and J. Short, *The new industrial engineering: information technology and business process redesign*. Sloan Management Review, 1990(summer): p. 11-27.
44. Hammer, M., *Reengineering work:don't automate, obliterate.*HBR,1990(July): p.104-112.
45. Curran, T. and A. Ladd, *SAP R/3 business blueprint*. 2000: Prentice-Hall.
46. Nissen, H. and M. Jarke, *Repository for multi-perspective requirements engineering.* Information Systems, 1999. **24**: p. 131-158.
47. Scheer, A., *Business Process Engineering,* 1998, Berlin: Springer.
48. Scheer, A., *ARIS_ Business process modeling, 2nd edition*. 1999, Berlin: Springer.
49. Johannesson, P. and E. Perjons, *Design principles for process modeling in enterprise application integration*. Information Systems, 2001. **26**: p. 165-184.
50. Balsters, H., et al., *Modeling dynamic rules in ORM*, in *OTM 2006/ORM 2006*. 2006, Springer: Montpellier, France.
51. Brinkkemper, S. and A. ter Hofstede, *The conceptual task model: a specification technique between requirements engineering and program development.*, in *Technical report 89-15*. 1989, Dept. of informatics. University of Nijmegen.
52. Hofstede, A.t. and S. Brinkkemper, *Conceptual task modeling* in *Technical report 89- 14.* 1989, Dept. of informatics. University of Nijmegen.
53. Prabhakaran, N. and E. Falkenberg, *Representation of Dynamic Features in a Conceptual Schema.* Australian Computer Journal, 1988. **20**(3): p. 98-104.
54. Nijssen, G., *An axiom and architecture for information systems*, in *Information systems concepts: an in-depth analysis*. 1989.
55. Olle, T.W., et al., *Information Systems Methodologies- A Framework for Understanding*. 1988: North-Holland, Amsterdam.
56. McIntosh, R., *Open-source tools for distributed device control within a service-oriented architecture.* Journal of the Association for Laboratory Automation, 2004(9): p. 404-410.
57. Mokhtar, S.B., et al., *Easy: efficient semantic service discovery in pervasive computing environments with QoS and context support.* The Journal of Systems and Software, 2007.
58. Cotroneo, D., et al., *Securing services in nomadic computing environments.* Information and Software Technology, 2007.
59. Engels, G., et al., *A method for engineering a true service-oriented architecture*, in *10th International Conference on Enterprise Information Systems*. 2008: Barcelona, Spain,.
60. Piprani, B., C. Wang, and H. Keqing, *A metamodel for enabling a service oriented architecture*, in *ORM 2008.* 2008: Monterrey, Mexico.
61. Terlouw, L., *Ontology-based change management of composite services* in *Workshop on pervasive systems* 2006: Montpellier, France.
62. Weigand, H., W.-J. van den Heuvel, and M. Hiel, *Rule-based service composition and service-orienetd business rule management*, in *ReMoD'08*. 2008: Aachen, Germany.
63. Bollen, P., *On the applicability of requirements determination methods*, in *Management and Organization*. 2004, University of Groningen: Groningen. p. 219.
64. Bollen, P., *Extending the ORM conceptual schema design procedure with the capturing of the domain ontology*, in *EMMSAD '07*. 2007, tapir Academic Press: Trondheim, Norway.