Rule Driven Modeling of Business Collaborations

Bart Orriens

January 24, 2006

Abstract

Current composite web service development and management solutions, e.g. BPEL, do not cater for flexible and adaptive business collaborations due to their predefined and inflexible nature that precludes them accommodating business dynamics. In this report we propose a rule driven approach for adaptive business collaboration development in which rules drive and govern the development process. We briefly introduce the Business Collaboration Context Framework (BCCF), which provides enterprizes with the context required for business collaboration. Then, we shortly explain our model driven approach that based on the BCCF provides enterprizes with the means to capture their different collaboration behaviors both from a business and technical point of view. Subsequently, we set out how we utilize rules to drive the development and management of business collaboration models to make them more flexible and adaptive.

Infolab Technical Report Series, no. 29

January 2006

1 Introduction

No man is an island; and nor is the modern enterprize. Over the years enterprizes have constructed extensive networks of bridges to other organizations in order to deliver their corporate business services. Nowadays, however, enterprizes find themselves drifting in a global ocean of seemingly continuous change. And in order to stay afloat they need to be highly dynamic and adaptive to maintain their competitive edge. This has put pressure on enterprizes to provide business services that can adapt to changes.

Recently, there has been increasing focus on service oriented computing (specifically web services based technologies) as the means to realize such flexible and adaptable business services by utilizing existing business services cross organizational boundaries; where the idea is that enterprizes can easily and rapidly establish, change and demolish bridges to others, i.e. develop, manage and dissolve business collaborations.

Unfortunately, current web service development and management solutions including the defacto standard BPEL4WS [11] are too narrowly focused and not capable of addressing the requirements of business collaboration, which rely on agile and dynamic processes. As a result it is very difficult to develop and manage business collaborations with existing technologies and standards.

In order to realize the vision of flexible and adaptive collaborations, an enterprize requires a sophisticated environment which provides it with the means to:

1. Navigate the global ocean, that is:
   (a) Be able to assess its own position, i.e. analyze its private (internal) activities (both from a business and technical point of view) in order to obtain a clear view of its cooperation capabilities and needs.
   (b) Be able to quickly set out an appropriate course, i.e. establish its collaboration potential based on its own position, and subsequently rapidly determine the feasibility of engaging other enterprizes.
   (c) Be able to make contact with and build bridges to suitable enterprizes, i.e. negotiate business collaboration agreements and consequently collaborate under the defined terms.
2. Do so in a flexible and adaptive manner, that is:

(a) Be able to determine the appropriateness of a selected course, i.e. foresee how changes will influence the ability to cooperate with others, and consider alternative courses of action.

(b) Be able to assess how changes affect existing bridges, i.e. how they influence existing collaboration agreements, and what must be done to properly manage them (i.e., defined, verified and versioned and deliver consistent results when executed).

To address these requirements, we propose an approach grounded on three key notions: firstly, we advocate the modularization of the context in which business collaboration takes place; where the purpose is to reduce the inherent complexity of collaboration development and management. Modularization will achieve this as it will allow the development and management process to be sliced into multiple, more manageable chunks. This will enable enterprizes to focus on specific development and/or management activities whilst at the same time maintain a clear view of their purpose in relation to the overall process.

Secondly, we argue for a model driven approach to accurately and completely describe the business collaboration context; where the aim is to enable enterprizes to capture their cooperations with others in the form of models. This will allow enterprizes to make their own position, collaboration potential and agreements explicit in an unambiguous and well-defined manner. The resulting models can subsequently be utilized for communication and reference purposes, as well as constitute the basis for the actual execution of collaborations.

Thirdly, we promote a rule based approach in which rules are used to drive and constrain business collaboration model development and management; where the idea is to mimic the role of rules in real life as guiding and governing the behavior of enterprizes in order to make business collaboration flexible and adaptive. Flexibility will come from the fact that development of business collaborations is governed by rules (which further more can be chained and used for making complex decisions and diagnoses); whereas adaptability will be achieved as changes can be managed with minimum disruption to existing collaborations.

In the remainder of this report we focus on the third component of our approach, the model driven approach to capture the business collaboration context as defined in the Business Collaboration Context Framework. Accordingly, the report is structured as followed: we first introduce a running example based on a complex insurance claim handling scenario in section 2. Next, in section 3 we briefly introduce our framework for business collaboration context. Then, we explain in detail how this context is captured via models in section 4 After that we show in section 5 the manner in which we employ rules to facilitate the flexible development and adaptive management of these collaboration models. Finally we present our conclusions and outline directions for future research in section 7.[1]

2 Example

To exemplify the ideas presented throughout this paper an example inspired by the case study in [18] is used. The example describes a complex multi-party scenario, which outlines the manner in which a car damage claim is handled by an insurance company (AGFIL). AGFIL cooperates with several contract parties to provide a service level that enables efficient claim settlement. The parties involved are Europ Assist, Lee Consulting Services, Garages and Assessors. Europ Assist offers a 24-hour emergency call answering service to policyholders. Lee C.S. coordinates and manages the operation of the emergency service on a day-to-day level on behalf of AGFIL. Garages are responsible for car repair. Assessors conduct the physical inspections of damaged vehicles and agree repair upon figures with the garages. The scenario outline is as followed (more details are introduced in the remainder of this paper where needed):

The policyholder (customer) phones Europ Assist using a free-phone number to notify a new claim. The claim is received by a call handler within Europ Assist's telephone assistance department. After verification of the customer' credentials to ensure that the provided policy details are valid and the occurred loss is covered, the call handler finds an approved repairer nearest to the customer's location. The customer is notified that this repairer will arrive at the scene shortly, if necessary with a replacement car and towing service. The call handler subsequently contacts the selected repairer to notify him of the incident. If the repairer is not available, another one will be selected and contacted. The customer is kept posted of such changes by phone. Once the repairer is on its way, the call handler contacts AGFIL to inform them of the made claim.

Upon receipt of the claim a claim handler will be assigned within AGFIL. The claim handler will gather all related claim information like customer records, claim history, etc. to Lee C.S. After that the claim handler will fill out the claim details on a claim form, which is subsequently stored pending further developments. Lee C.S. in the meanwhile has one of its consultants working on the claim. The first thing this consultant does, is contact the

[1] This work has been partially funded by the Netherlands Organization for Scientific Research (NWO)

garage to inquire about the status of the car. The garage has picked up the car while the previous was going on and has worked out an estimate of the car repair cost. If this cost was below $500 then the garage will have started repairs. But if the costs were higher, the consultant at Lee C.S. contacts an assessor to go to the garage and check out the car for him -or herself. This assessor makes an independent estimate of the repair costs and negotiates a final price with the garage.

The result of the assessment is next reported back to the consultant at Lee C.S. The consultant reads the report and approves repair. An approval notification is sent to the garage, which consequently starts repairs on the car. Lee C.S' consultant also informs the claim handler at AGFIL of the final repair cost estimate upon which the claim handler incorporates the new information in the claim form. Once the garage has completed its repairs on the customer's car, an invoice is communicated to the consultant at Lee C.S. The consultant checks the invoice to see if it matches the earlier received cost estimate. Once the invoice is approved, the consultant sends the invoice onwards to AGFIL. The claim handler receives the invoice and adds it to the claim form. Payment for the claim is also issued.

3 Business Collaboration Context Framework

At the heart of our approach stands the Business Collaboration Context Framework (BCCF). The BCCF captures the context in which business collaboration development and management takes place by adopting a three dimensional view. Through this three dimensional view modularization of the definition and management of business collaborations is achieved. Moreover, it facilitates different interest groups in business collaboration to communicate with each other. An overview of the framework is shown in Fig. 1; where for a complete overview the reader is referred to [21].

As the figure illustrates we modularize the business collaboration context along three dimensions in the BCCF, being *behavior*, *level* and *facet*. We briefly discuss these in the following, and explain how they can be used to modularize the business collaboration context; as such reducing the complexity of collaboration development and management.

3.1 Behavior

The first dimension, **behavior**, places emphasis on the different behaviors that an enterprize exhibits in business collaboration; where consequently the purpose and target of development and management varies.

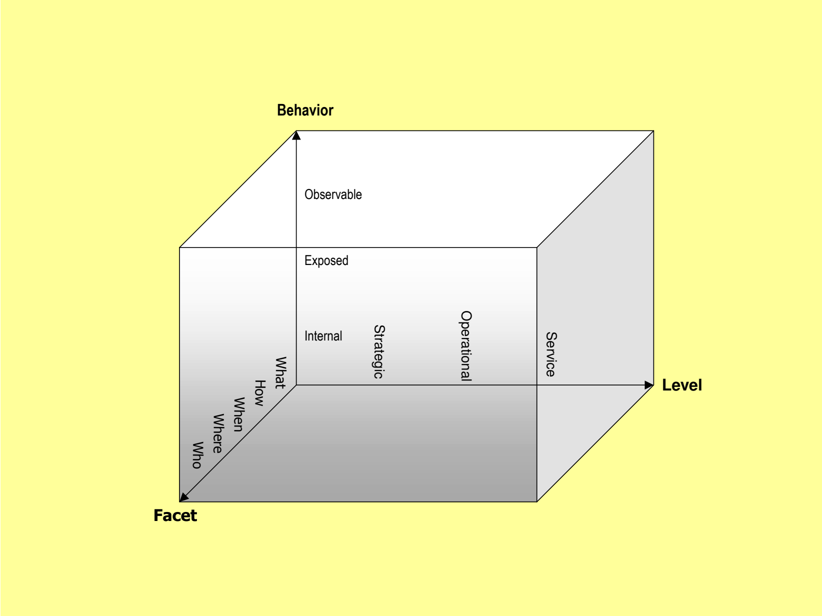

Fig. 1: Business Collaboration Context Framework (BCCF)

The behavior dimension encompasses three types of behavior captured in three corresponding so-called collaboration aspects (inspired by among others [15,30,35]): observable, exposed and internal behavior expressed in the *conversation*, *participant public behavior* and *internal business process* aspect respectively.

The observable behavior constitutes the externally visible behavior between participants in a business collaboration. Expressed in the *conversation aspect*, the observable behavior specifies how its participants are expected to behave in the collaboration from a global (participant independent) point of view. As such the conversation aspect captures the behavior agreed upon by its participants, i.e. constituting a collaboration agreement. Agreements can be partitioned per participant in order to determine the behavior expected from individual participants. Moreover, in a multi-partner scenario the overall agreement can be split into binary sub-agreements describing the observable behavior per pair of participants.

Rather than being global in nature the exposed behavior is individual to each participant, yet visible to other participants. Captured in the *participant public behavior aspect* it describes how an individual participant can publicly behave in a business collaboration (i.e. its potential collaboration behavior). Similarly, the internal behavior is also individual to each participant. However, it is only of interest to this particular participant, i.e. it can not be observed by other participants. The internal behavior is thus private in nature, and is represented in the *internal business process aspect*.

The identified behaviors are not independent from one another; rather they are highly interrelated. The nature of a participant's internal business processes strongly influences the way in which it can interact with others; whereas in turn this potential collaboration behavior of participants greatly affects the make up of any collaboration agreements forged among them. Vice versa, the agreed upon behavior can prompt adjustment of a participant's exposed behavior; which consequently can necessitate alteration of its internal behavior.

3.2 Level

The second dimension, **level**, recognizes the fact that the different business collaboration behaviors of an enterprize take place at several levels; where consequently the domain, degree of abstraction and the type of developers in development and management varies. In the BCCF three layers of abstraction are identified (inspired among others by [20, 37]): the *strategic*, *operational* and *service* level spanning from high level requirements to technical realization of collaboration behaviors.

At the strategic level the focus is on behavior that is abstract in nature, describing the purpose and high level requirements an enterprize has with the behavior. This level is the domain of the strategic management. Depending on the collaboration aspect considered, developers here are interested in capturing the private strategies of the enterprize (internal behavior), the objectives it wishes to realize via collaboration with others (exposed behavior), and shared objectives to which it has committed itself in actual collaborations (observable behavior).

The operational conditions under which enterprizes exhibit their behavior are part of the operational level. This level establishes how high level strategic behavior (private, exposed and observable) will be operationalized. Business analysts are the people involved at this level of development and management. The technical realization of operational behavior is done at the service level by IT-specialists, who wish to describe how and what services the IT-infrastructure delivers to support the operational activities.

Similar to the different collaboration behaviors being related, dependencies exist between the same type of behavior at different levels. Strategic decisions can affect operational parameters, which in turn can influence demands on the services delivered by the IT-infrastructure. Vice versa, technological changes may prompt adjustment of operational activities, possibly leading to reconsideration of high level strategies.

3.3 Facet

The third dimension, **facet** captures the fact that the collaboration behaviors conducted by enterprizes affect many different parts. Facets represent these different parts of a business collaboration behavior that can be observed; and where consequently the focus and type of developer involved in collaboration development and management varies. Five facets are distinguished (inspired by among others [12, 34, 37]): *what*, *who*, *where*, *when* and *how* facet.

The *what* facet emphasizes the structural view of a collaboration behavior, focusing on what things are used to perform a collaboration behavior. The *how* facet takes a functional standpoint, and thus concentrates on how a collaboration behavior is conducted. The *who* facet concerns the participant(s) conducting the collaboration behavior. The location(s) at which the behavior is carried out are expressed in the *where* facet, whereas its temporal dimension is covered in the *when* facet.

Together the above facets completely encompass any collaboration behavior at any level; where their exact semantics are dependent on the specific level and collaboration aspect that they are part of. Moreover, although facets represent different parts of a business collaboration behavior, they also interact with the other facets. How things are done for example strongly relates to the temporal conditions that apply; whereas on a similar note who is involved may depend on the location at which things are to be done.

4 Modeling The Business Collaboration Context

The presented BCCF in the previous section helps reduce the level of complexity by partitioning the business collaboration context through usage of the dimensions of collaboration aspects, levels and facets. In order to actually develop and manage business collaborations enterprizes require a way to explicitly capture this context. In this section we first introduce the foundation of our model driven approach in subsection 4.1. After that in subsection 4.2 through 4.5 we discuss the different business collaboration models and their mappings to describe collaborations along the three BCCF dimensions.

4.1 Foundation

We employ two types of model in our model driven approach to capture the three dimensions of collaborations aspects, levels and facets: meta models and models, both of which are defined for individual levels. Meta models provide design guidelines in terms of classes and their relationships to capture the different collaboration behaviors in the same terms; where depending on the type of behavior being modeled specific constraints are placed on the meta-model. Models represent a particular behavior, where the dependencies among behaviors at different levels or different aspects are made explicit by vertical and horizontal mappings respectively. Models and mappings are derived by populating the *modeling description atoms* in or between meta models .

We identify five types of modeling description atom, being *model*, *element*, *property*, *link*, and *mapping*; where the meta models define under what conditions these atoms can be combined. Their semantics and formal definition are as follows (where we use first order logic for the formalization):

1. *model*; captures the specifics of a business collaboration model. A model has a name, level and aspect. Level must be equal to 'strategic', 'operational' or 'service', and aspect equal to 'internal business process', 'public participant behavior' or 'conversation'). A model **m** is formally defined as a tuple M(w,x,y); where 'w' is the name of the model, 'x' the level, and 'y' the aspect.

2. *element*; represents a facet of a collaboration behavior, i.e. what, how, who, where, and when facet. An element has a uniquely identifying name and a type. The element type reflects the kind of facet being represented at a particular level. Each element has one or more properties. An element **e** is formally defined as E(w,x,y); where 'w' is the name of the model, 'x' the type, and 'y' the model reference.

3. *property*; defines a characteristic of an element, enriching the description of a facet. Each property has a name, type and value. The name provides an unique identifier, whereas the type reflects the kind of characteristic being defined. Value defines the value of the property. A property **p** is formally defined as $P(w,x,y,z,w_1)$; where 'w' is the name of the model, 'x' the type, 'y' the value, z the element reference, and w_1 the model reference.

4. *link*; expresses connections between elements belonging to the same model. Links have a name and type. The name is for identification purposes, whereas the type indicates the kind of relationship being

established. A link **l** is formally defined as L(w,x,y,z,w_1); where 'w' is the name of the model, 'x' the level, 'y' the source element, 'z' the target element and 'w_1' the model reference.

5. *mapping*; specify relations between elements from different models. A mapping has a name and type. The name gives an unique label to the mapping; the type signifies the kind of mapping defined. A mapping can be 'vertical' in nature linking elements from models at different levels, or 'horizontal' connecting elements from models at different behaviors. A mapping **c** is formally defined as C(w,x,y,z) where 'w' is the name of the model, 'x' the level, 'y' the source element, and 'z' the target element.

Based on the above we formally define a model **MS** using set theory as a collection of elements, properties, and links:

MS: m $\cap$ ES $\cap$ PS $\cap$ LS

stating that a model is the conjunction of a model tuple m, the set of elements ES, the set of properties PS and the set of links LS, where these are defined as:

ES: the set of elements defined as $\{e_0 ... e_n\}$.
PS: the set of properties defined as $\{p_0 ... p_n\}$.
LS: the set of links defined as $\{l_0 ... l_n\}$.

Subsequently, we can formally define a mapping MAP as:

MAP: $MS_1 \cap MS_2 \cap CS$

stating that a mapping is the conjunction of two models MS_1 and MS_2, and the set of mappings CS; where CS is defined as $\{c_0 ... c_n\}$.

On top of this foundation we have defined meta models for modeling the strategic, operational and service level in the BCCF. In the following subsections we discuss the different models and mappings between them; where snippets of exemplary models for the AGFIL application describing an interaction between `Garage Inc` and `Lee C.S` are illustrated in Fig. 2, showing the strategic, operational and service representation of this interaction respectively. The reader is referred to [22] for a complete discussion of the meta models as well as the examples.

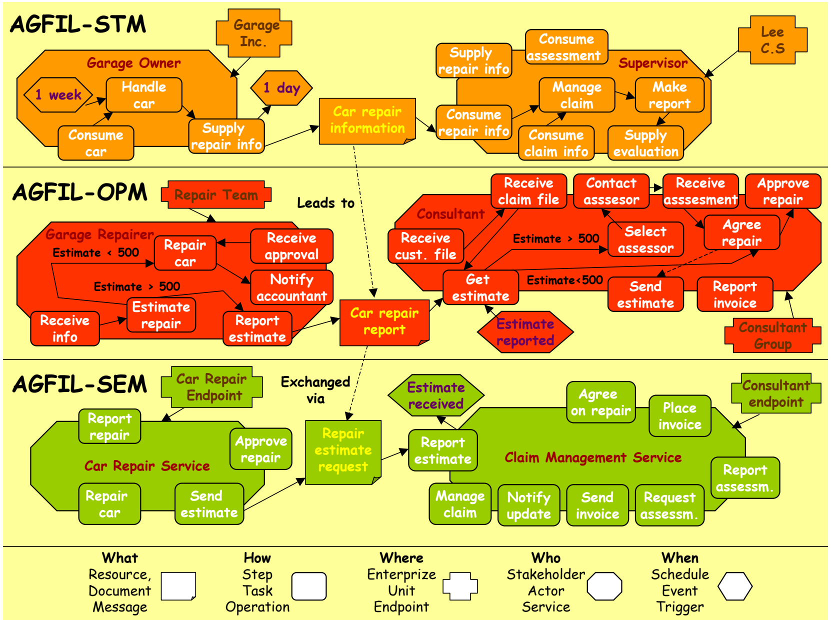

Fig. 2: AGFIL Collaboration Models

4.2 Strategic Models

At strategic level, strategic models like the AGFIL-STM in Fig. 2 capture purpose and high level requirements of business collaborations, akin to requirements analysis [6, 35]. As illustrated in Fig.2 these strategic models are expressed in terms of resources, steps, stake holders, enterprizes, and schedules. Resources make up the structural view of a strategic collaboration, where each resource provides an abstraction mechanism for means such as financial, human and informational capital. Resources have a certain value, capacity and description.

Resources are used and produced by steps which represents high level functions such as `manage claim`. Steps can be dependent on one another, and are of type 'internal' (like `handle car`); i.e. private to an enterprize (presented inside the stakeholder boundary in Fig.2), or type 'communication' representing resource supply and consumption e.g. `consume repair information`. Stake holders like `garage owner` describe the participants involved who are responsible for carrying out defined steps. Each stake holder belongs to an enterprize, where an enterprize records organizational information. Stake holders and their enterprizes are bound by schedules, like the deadline of `1 week` for `handle car`.

In addition, non-functional (NF) characteristics can be modeled as properties, currently with regard to assessment, payment, quality of service (QoS) and security. Assessment properties associated with schedules depict how their progress is to be measured; payment properties reflect what payment criteria apply to a step; quality of service properties represent high level objectives regarding the reliability, responsiveness, efficiency, accuracy, availability and accessibility of steps; whereas the security properties address the most common threats of masquerading and unauthorized access for steps, and disclosure and modification of resources. For more information the reader is referred to [23, 24, 25, 26].

Using the described modeling elements the different strategic collaboration behaviors can be defined. The conversation aspect (all modeling elements external to or on boundary of stake holders) defines the exchange of resources between enterprizes to achieve shared strategic objectives, i.e. a strategic agreement (akin to [6, 35]. The strategic potential of a participant (all elements at border of a single stakeholder) is defined in terms of the resources it can exchange; whereas the private behavior identifies the high-level steps performed to realize this potential.

4.3 Operational Models

At operational level, operational models like the AGFIL-OPM in Fig. 2 depict how high level strategic behavior is realized in terms of operational activities. These are expressed in terms of documents, tasks, actors, units, and events; expressing what, how, who, where, and when facet at operational level respectively. Documents (like `car repair report` represent the flow of information in a collaboration behavior, i.e. capturing its structural view in operational semantics. Documents provide characteristics of the information e.g. in terms of semantics and syntax used.

Documents are used and produced by tasks, which represent specific business functions. Tasks are of type 'internal' or 'communication' like `collect claim form` and `report invoice` (represented inside or on the boundary of the actors respectively). Actors such as `garage repairer` and `consultant` are responsible for carrying out tasks. Actors belong to units such as `repair team` unit, representing an organizational construct. In order to assess progress, keep logs to ensure non-repudiation, and etceteras, events capture business occurrences with properties such as 'date', 'time', 'severity'.

With regard to definition of NF-characteristics in operational models, specification of assessment, payment, quality of service and security properties is supported. Assessment properties indicate which events need to be monitored, logged and published; whereas payment properties capture the cost of tasks and under what conditions this is to be paid. With regard to quality of service a wide range of properties are associated with tasks, e.g. to measure failure rate, recovery rate, latency, and throughput. In a similar fashion security properties can be set for tasks and documents to define the mechanisms to be used for authentication and authorization, and confidentiality and integrity respectively (see [23, 24, 25, 26] for more information).

In terms of the collaboration aspects an operational model constitutes the following: in the conversation aspect (elements on or outside actor borders) operational models constitute agreements which define the observable flow of information between actors; comparable to RosettaNet [31] or ebBPSS [16] models. In the participant public behavior aspect (elements on border of a single actor) an operational model constitutes the documents an actor can exchange; like ebCPP based models [16]. In the internal business process aspect (elements within actor) the resulting models specify work flow like business process descriptions (similar to e.g. BPML [8]).

4.4 Service Models

At service level, operational models are translated into service models that specify how the described operational behavior is realized using the services offered by the IT-infrastructure. Service models are defined in terms of messages, operations, services, endpoints, and triggers. Messages represents containers of information (e.g., `repair estimate request`), consisting of meta-data and actual data. Meta-data comprises the information required to deliver the message and enable its processing. Payloads contain any content of the message not conveyed in its meta-data (like text documents, images, video files, etc).

Messages function as the inputs and outputs of operations such as `place invoice`. Operations represent specific technical functions, and are described in terms of their access details. Operations, just as steps and tasks at strategic and operational level respectively, can be dependent on one another. Additionally, they can be of type 'internal' or 'communication'. Operations are grouped in services (e.g. `car repair service`, representing collections of logically related operations. Services are provided by endpoints (like `claim handling endpoint`) and have properties 'network location' and 'type'. To express technical occurrences triggers like `claim request acknowledged` are defined.

Messages, operations, and triggers are augmentable with NF properties concerning assessment, payment, quality of service and security. Assessment properties are part of a trigger's definition and define how long it is to be stored, published using a push mechanism, and so on. Payment properties specify pricing information for an operation (like minimum and maximum price), what payment instrument is used (such as cash, credit card), etc. Quality of service properties identify the variables measured to determine the QoS level like no. of failures, and no. of accepted requests. Lastly, security properties depict the measure used to enforce security, e.g. the encryption protocol RSA to ensure confidentiality (also see [23, 24, 25, 26]).

By combining messages, operations, services, endpoints and triggers enterprizes can define their observable, exposed and internal behavior. The observable behavior service model (the elements on or outside the border of services) is akin to the notion of choreography [30] defining the agreed upon exchange of messages among services. Models of the exposed behavior are formed by elements placed on the border of individual services, and depict the operations a service can offer and the conditions under which this can be done (akin to e.g. WSDL [10]). Within a service the modeling elements depict internal behavior models akin to orchestration [30] (not shown in Fig. 2).

4.5 Mappings Between Models

For the specification of dependencies between different collaboration behaviors at different levels and aspects, we employ vertical and horizontal mappings respectively. *Vertical mappings* are realized by providing links between the classes in different meta-models and instance models at different perspectives. The vertical mappings are based on the implicit links that exist between classes that describe the same facet at different levels in the same collaboration behavior.

The mappings between a strategic and operational behavior express how strategic behavior is realized in terms of operational activities. One strategic behavior can be mapped to multiple operational behaviors. Vice versa, an operational behavior can support multiple strategic behaviors. An operational behavior is itself mapped to one or more service behaviors expressing how operational activities are realized using services from the IT-infrastructure; where each service behavior can help realize multiple operational behaviors.

For example, stake holder `garage owner` is mapped to actor `garage repairer` via a *controlledBy* mapping, reflecting that the latter is employed by the former. `garage repairer` itself is *representedAs* `car repair service` incorporating this actor as an entity in the IT-infrastructure providing services. Note that this allows enterprizes to develop and manage services offered both by human and computerized actors in the same manner.

Horizontal mappings define links between instantiations of the same class in each meta-model, which are part of models describing different collaboration behavior. The mappings are grounded on the implicit generic relations that exist among collaboration behaviors at an individual level.

The one-to-one mapping from an internal to an exposed behavior expresses how a private process communicates with the outside. Vice versa, an exposed behavior behavior can correspond to one or more internal behaviors; in which case the private portions of the latter behaviors vary. Between an exposed and observable behavior there exists a subsumption relationship; that is, how a participant can behave subsumes (or is equal to) of how it is expected to behave; where this relation is one-to-many, and one-to-one from exposed or observable behavior standpoint respectively.

To illustrate, `car repair report` in `Garage Inc`'s private process is *sendAs* a document with the same name in its exposed behavior. Observe that the characteristics of the two documents can vary, for example regarding content, syntax and language. `car repair report` in exposed behavior is itself *exchangedAs* `car repair report` in observable behavior; where once again their definitions need not be the same.

Note that in the above we have not discussed the matter of mapping individual properties at one level to those on another level, or from one type of collaboration behavior to another. Obviously such dependencies do exist, e.g. that schedule `1 day` mandated in the strategic agreement between `Garage Inc` and `Lee C.S.` must not be more strict than what can be supported by the garage in its exposed behavior. However, for reasons of brevity these are not covered here; more information can be found in [23, 24, 25, 26].

5 Rule Driven Business Collaboration Development and Management

In section 4 we discussed how enterprizes can utilize a model driven approach to capture the business collaboration context in which they operate. This discussion illustrated that development and management of the necessary business collaboration models requires an intensive modeling effort on the part of the enterprizes involved. Fig.3 visualizes this effort from the point of view of an individual enterprize.

As Fig.3 shows enterprizes are faced with the challenge of capturing the three different types of collaboration behavior that they exhibit, at three layers of abstraction each of which covers five different facets (as well as

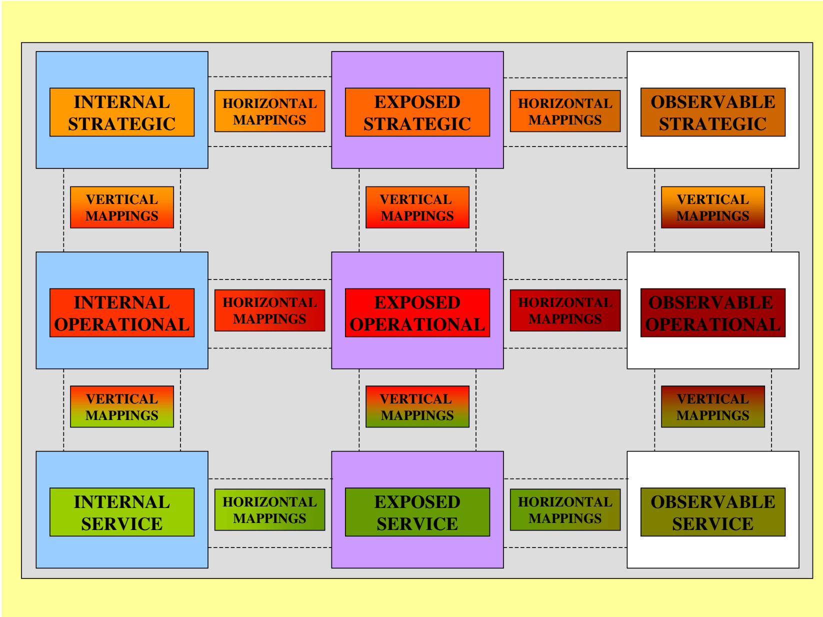

Fig. 3: Modeling Business Collaborations

various domains of a non-functional nature such as security). Even when taking into account the benefits of having highly modularized the development and management of business collaborations, this still leaves enterprizes with the task of creating and maintaining all the different models and the dependencies between them; whilst at the same time ensuring that individual models are consistent, models at different levels are aligned, and models capturing different behavior do not contradict one another (i.e. that the vertical and horizontal mappings respectively are valid).

In addition, enterprizes must ensure that the modeled behaviors are compliant with the rules applicable to them. Such rules are wide ranged in scope, and govern the manner in which an enterprize conducts its business. They can be external in nature, like relevant legislation and industry mandated regulations, as well as be specific to individual enterprizes such as organizational policies, work procedures, and so on. To further complicate matters these kinds of rules are frequently subject to change, and consequently the model(s) capturing affected behavior(s) must be adjusted accordingly. When combined these two issues result in the rather daunting prospect of having to create and manage a large number of collaboration models in a consistent manner; where at the same time the definition of these models will need to be frequently updated in a timely and orderly manner.

To give one simple illustration of this: suppose that at the service level of an internal behavior a critical internal service adopts an updated messaging protocol because of new technical regulations. However, because of this the service can no longer communicate with other internal services. This has several consequences: one is that the availability of exposed services using the affected internal ones becomes compromised. This in turn leads to the violation of made agreements with other enterprizes. The other consequence is that at the operational level one or more business tasks in the mapped internal behavior can no longer be successfully completed. This then results in the violation of made operational agreements as well as jeopardizing the high level strategic objectives.

As even this simple example already shows a change in some part of a business collaboration can have an enormous, cascading affect on the collaboration as a whole. Therefore, given the complexity and sheer scope of the modeling effort it is simply not feasible in our view to perform business collaboration development and management manually; whilst at the same time be able to do this in a flexible and adaptive manner. This would constitute a violation of the second set of requirements identified in section 1 regarding flexibility and adaptability.

We therefore submit that enterprizes need a mechanism which assists them in the flexible development and adaptive management of business collaboration. We have developed a rule driven mechanism for this purpose that is twofold in nature: 1) we use rules to drive the development and management process, that is, to create and maintain business collaboration models; 2) we employ rules to maintain the consistency of and among these collaboration models. This is visualized in Fig.4.

The figure shows that each type of collaboration model and the mappings between them now has a set of so-called **management rules** associated with it (represented by ellipses in Fig.4). The validity of the combination of a model and its rules, as well as of the relationships between models, is controlled by **construction rules**. In both cases when we talk about rules we refer to ”precise statements that describe, constrain and control the structure, operations and strategies of a business” [32]. In the following we discuss the different types of management and construction rules. Subsequently we explain how they can be used in conjunction to develop and manage business collaboration models.

5.1 Management Rules

Management rules express the peculiarities, originality and values of in-

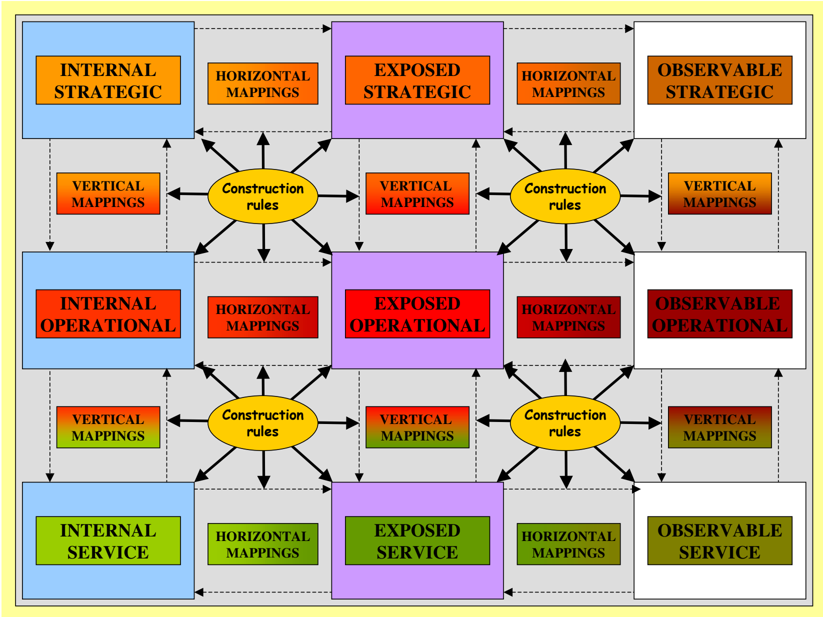

Fig. 4: Using Rules For Modeling Business Collaborations

dividual enterprizes. Management rules represent all the knowledge about the behavior (or mapping between behaviors) being modeled that is likely to be the subject of change. Extraction of this knowledge and making it explicit enables enterprizes to not only manage it, but also to link modeling decisions to their underlying rationale. Moreover, by defining different sets of rule for different circumstances, enterprizes are able to change their behavior (or relations between behaviors) in accordance with the exact context in which it is to take place.

Management rules can be associated with an individual model, like "All customers with status 'special' get a 20% discount", or more complex such as "The availability rate of a money transfer order operation must be higher than 95% between 09.00 am and 5.00 pm". Management rules can also be associated with a mapping, e.g. that a business event at operational level will only be raised if a specific combination of service level triggers occurs. Another example of a mapping rule is that the price for an operation as specified in the exposed service behavior is at least 150% of its cost in the internal service behavior.

Management rules of individual collaboration behaviors are subdivided in accordance with the three dimensions of BCCF. Classified along collaboration aspect they constitute *private regulations*, *offered conditions* and *agreed on conditions* in internal, exposed and observable behavior respectively. Along level management rules are sub-categorized in *goals*, *business rules* and *constraints* at strategic, operational at service level respectively. Based on facet management rules are grouped in *structural rules*, *functional rules*, *participant rules*, *geographical rules*, and *temporal rules* in what, how, who, where, and when facet respectively.

Management rules governing the relations between behaviors, i.e. mapping rules, are subdivided into *vertical mapping rules* and *horizontal mapping rules*. These can be further classified based on what two behaviors the associated mappings relate. This results vertically in *strategic-operational mapping rules* and *operational-service mapping rules*, and horizontally in *internal-exposed mapping rules* and *exposed-observable mapping rules*.

Interpreted in the context of modeling, management rules represent all the knowledge about the behavior (or mapping between behaviors) being modeled that is likely to be the subject of change; whereas the model/mapping captures the part of the behavior that is relatively stable. Extraction of this knowledge and making it explicit enables enterprizes to manage it accordingly. More importantly though it allows them to link modeling decisions to their underlying rationale, where this is twofold: 1) it can be verified that every modeling decision following from the applicable management rules has been made, i.e. the modeled behavior is *complete*; and 2) that every modeling decision made can be justified in terms of these rules, i.e. the modeled behavior is *minimal*. If the modeled behavior is both complete and minimal, the enterprize then knows it is *conform* to its associated management rules.

In order to maximize these benefits we associate rules with individual modeling description atoms; where a rule belongs to a particular description atom if its conclusion refers to this atom. The advantage is that when each atom has its own set of rules, they can be combined in a plug and play manner; as such maximizing the degree of flexibility and adaptability. An atom has three different management rule sets governing how it can be combined a) within an individual model, called *individual management rules*; b) within a vertical mapping, called *alignment management rules*; and c) within a horizontal mapping, called *compatibility management rules*. Rule sets are formalized as follows:

$\mathbf{RS}_{PIM}$: the set of individual management rules for a property p defined as $\{\mathbf{r}_{PIM0}...\mathbf{r}_{PIMn}\}$.

The two other rule sets of a property as well as those of models, elements, links, and mappings can be defined in a similar manner.

Each rule in the rule set RS_{PIM} (and all other rule sets) is expressed in terms of the modeling description atoms (i.e. models, elements, attributes, links, and mappings) as First Order Logic (FOL) material implications. This allows for an uniform definition of the different types of management rule that can be identified in the business collaboration context in accordance with its dimensions (for a complete rule classification the reader is referred to [27]). Formally a rule r is defined as:

$\mathbf{r}_{label}$: $m_1 \sigma e_0 \sigma ... \sigma e_k \sigma p_0 \sigma ... \sigma p_l \sigma l_0 \sigma ... \sigma l_m \sigma c_0 \sigma ... \sigma c_m \rightarrow p$; representing a property individual construction rule named 'label' with k,l,m $>$ 0; and $\sigma \in \{\wedge \vee \oslash\}$ are standard logical operators

stating that a rule is divided into *conditions* and *conclusions* before and after the $\rightarrow$ respectively; where the conditional part constitutes a series of conjunctive, weak disjunctive or strong disjunctive statements of modeling description atoms (optionally grouped using '(' and ')', and where statements can be negated using $\neg$); and the conclusion part consists of a single, non-negated atom.

An example of a rule r_{PIM} is $r_{G_modification}$ (where the 'G' reflects the fact that it is a goal, i.e. a rule at strategic level):

$r_{G_modification}$: E(carRepairInformation,resource) $\rightarrow$ P(ModProp,modification,true,carRepairInfo

which states that if there is an element named `carRepairInformation` of type 'resource' its property 'modification' must be set to 'true'; where `ModProp` is a placeholder for actual property names indicated by the capital letter.

Now, as we build complex constructions out of individual modeling description atoms, we can combine their rules in more complex sets as well. For example, given that a model constitutes a set of elements, properties and links, the set of individual management rules associated with it is the conjunction of the sets of rules of its individual modeling description atoms. Formally we can define this:

$\mathbf{RS}_{MSIM}$: the set of individual management rules for a model MS; defined as $RS_{MIM} \cap RS_{EIM0} \cap ... \cap RS_{EIMn} \cap RS_{PIM0} \cap ... \cap RS_{PIMn} \cap RS_{LIM0} \cap ... \cap RS_{LIMn}$; where these subsets constitutes the sets of individual management rules of the modeling description atoms in model MS, e.g. RS_{EIM0} defined as $\{r_{EIM00} ... r_{EIM0n}\}$

With the above in place we can formalize the notion of conformance of a modeled behavior introduced earlier as follows:

Model Conformance: A model MS is conform to its set of individual management rules RS_{MSIM}, denoted as MS $\models$ RS_{MSIM}, if it is complete and minimal:

1. *complete*: $\forall$r $\in$ RS_{MSIM} with r: $m_1 \wedge e_0 \wedge ... \wedge e_k \wedge p_0 \wedge ... \wedge p_l \wedge l_0 \wedge ... \wedge l_m \rightarrow a$, if ($m_1 \wedge e_0 \wedge ... \wedge e_k \wedge p_0 \wedge ... \wedge p_l \wedge l_0 \wedge ... \wedge l_m$) true in MS, then a $\in$ MS.

2. *minimal*: $\forall$a $\in$ MS there exists a rule r in $\in$ RS_{MSIM} with r: $m_1 \wedge e_0 \wedge ... \wedge e_k \wedge p_0 \wedge ... \wedge p_l \wedge l_0 \wedge ... \wedge l_m \rightarrow a$, and ($m_1 \wedge e_0 \wedge ... \wedge e_k \wedge p_0 \wedge ... \wedge p_l \wedge l_0 \wedge ... \wedge l_m$) true in MS.

which essentially state that 1) for each rule in RS_{MSIM} if its conditions are true in the model, its conclusion must be true as well; and 2) for every atom in the model there must be a rule in RS_{MSIM} that can justify its presence. Note that due to space limitations we do not provide detailed definitions here for when exactly the conditions of a rule r are true in a model MS; these are well established within the area of first order logic (see for example [**?**] for more information).

Now, due to the fact that rules are associated with individual modeling description atoms we can formally verify conformance of any arbitrarily combination of atoms in a similar manner as above. For example, by combining the rule sets of `supply repair information`, `repair information` and `consume repair information` we can verify this specific interaction in the AGFIL-STM in Fig. 2. To check the `leadsTo` mapping between `car repair information` and `car repair report` we can join the rule sets of the modeling description atoms describing them. Also, when the management rules change the conformance of existing models can be verified; and any problems can be identified. In this manner enterprizes are able to develop and manage collaboration models in a highly flexible and adaptive manner.

5.2 Construction Rules

Models and mappings also have sets of **construction** rules associated with them (as shown in Fig. 4). Construction rules are generic in nature, that is, applicable to all business collaborations. Examples are that each step must be decomposed into at least one task, that the specified value of an attribute is within its value domain, that two tasks can not be dependent on each other (in order to prevent deadlock), or that a mapping from a task requiring authentication to an operation not capable of doing so is not feasible.

The purpose of construction rules is to encode knowledge about the business collaboration domain itself. The advantage of making such knowledge explicit is that we can utilize the resulting rules to govern business collaboration model development and management. Construction rules guard the boundaries in which collaboration models and mappings can be developed and managed; that is, ensure that models are valid, aligned and compatible. Their purpose is twofold: 1) verify that every modeling decision required has been correctly made, i.e. the model/mapping is *complete* and *correct*; and 2) verify that modeling decision do not contradict one another, i.e. the model/mapping is *consistent*. Completeness and correctness together constitute the *syntactical soundness* of a model/mapping, whereas consistency reflects their *semantical soundness*.

Similar to management rules, construction rules are grouped into sets associated with individual modeling description atoms. This allows us to gain the same degree of granularity in the verification of syntactical and semantical soundness as for conformance. Each atom has three sets of construction rules governing its usage in an individual model via *individual construction rules*, in a vertical mapping via *alignment construction rules* and in a horizontal mapping via *compatibility construction rules*; which themselves are subdivided into sets of completeness, correctness and consistency rules. These rule sets are defined like management rule sets:

$\mathbf{RS}_{CACN}$: the set of alignment construction rules regarding consistency for a mapping c defined as $\{r_{CACN0} ... r_{CACNn}\}$.

which represents the set of individual management rules for a property p. The other construction rule sets of a mapping as well as those of models, elements, properties and links can be defined in a similar manner.

We can define the different kinds of construction rule in a similar manner as management rules; that is, grounded on the different modeling description atoms as first order logic material implications. As such, both management and construction rules are specified in an uniform manner. For example, each rule in RS_{CACN} is formally defined as:

$\mathbf{r}_{CACN}$: $m_1 \wedge e_0 \wedge ... \wedge e_k \wedge p_0 \wedge ... \wedge p_l \wedge l_0 \wedge ... \wedge l_m \wedge c_0 \wedge ... \wedge c_m \rightarrow c$; representing a mapping alignment consistency construction rule with k,l,m >

0.

An example of such a rules is $\text{VMAPR}_{modification}$, which states that for all resources that require modification protection, all documents communicated to realize exchange of these resources must use some form of integrity mechanism:

$\text{VMAPR}_{modificationMapping}$: E(X,resource) $\wedge$ P(modification,true,X) $\wedge$ E(Y,document) $\wedge$ mapping(leadsTo,X,Y) $\rightarrow$ P(integrity,true,Y)

which states that if a resource X is mapped to a document Y, 'modification' is set to true for the resource and 'integrity' to 'false' for the document, a mapping conflict exists. The difference with a management rule definition such as $\text{G}_{modification}$ is that in a construction rule like $\text{VMAPR}_{modificationMapping}$ we find more variables; making such rules more generic in nature.

Complex sets of construction rule can subsequently be defined in an identical manner as explained for management rules. For example: given that a vertical mapping constitutes two models and a set of mappings, if we want to verify its semantical soundness then we have to take the set of alignment consistency rules associated with this mapping; which is the conjunction of the sets of alignment consistency rules associated with the two models, and the set of alignment consistency rules associated with the mapping. Formally we can define this set as:

$\mathbf{RS}_{VMAPACN}$: the set of alignment consistency rules for a mapping VMAP between model MS_1 and MS_2; defined as $\text{RS}_{MSACN1} \cap \text{RS}_{MSAC2} \cap \text{RS}_{CACN0}$ $\cap ... \cap$ RS_{CACNn};

where RS_{MSACN1} and RS_{MSACN2} constitute the set of alignment consistency rules for MS_1 and MS_2 respectively, defined as $\text{RS}_{M1ACN} \cap \text{RS}_{E1ACN0}$ $\cap ... \cap$ $\text{RS}_{E1ACNn} \cap \text{RS}_{P1ACN0} \cap ... \cap \text{RS}_{P1ACNn} \cap \text{RS}_{L1ACN0} \cap ... \cap \text{RS}_{L1ACNn}$ and $\text{RS}_{M2ACN} \cap \text{RS}_{E2ACN0} \cap ... \cap \text{RS}_{E2ACNn} \cap \text{RS}_{P2ACN0} \cap ... \cap \text{RS}_{P2ACNn} \cap \text{RS}_{L2ACN0}$ $\cap ... \cap$ RS_{L2ACNn}.

Once we have the set $\text{RS}_{VMAPACN}$ for the vertical mapping VMAP, then verification of its semantical soundness becomes a matter of checking whether VMAP is conform the the rules in this set, denoted as VMAP $\models \text{RS}_{VMAPACN}$. For reasons of brevity we do not provide the accompanying formalization here; we feel confident to leave it to the reader to work out these definitions, as they are highly similar to the ones presented in subsection 5.1 for conformance of a model to its management rules. We also forego here on formally defining model syntactical soundness, model semantical soundness, mapping conformance and mapping syntactical soundness.

Rather, before we conclude we want to formally define the earlier introduced notions of validity, alignment and compatibility. Informally we stated that a model is valid if it is conform the enterprize's requirements. Model validity can now be formally defined as:

Model Validity: A model MS is valid if it is true that MS $\models$ RS_{MSIM}, MS $\models$ RS_{MSICM}, MS $\models$ RS_{MSICR} and MS $\models$ RS_{MSICN}

stating that MS is valid if it is conform its individual management rules, completeness rules, correctness rules and consistency rules, i.e. it meets the conformance, and syntactical and semantical soundness requirements. Model alignment can be formally depicted as:

Model Alignment: Two models MS_1 and MS_2 are aligned via a mapping VMAP if it is true that VMAP $\models$ RS_{VMAPAM}, MS $\models$ $\mathrm{RS}_{VMAPACM}$, MS $\models$ $\mathrm{RS}_{VMAPACR}$ and MS $\models$ $\mathrm{RS}_{VMAPACN}$

stating that models MS_1 and MS_2 are aligned via a mapping VMAP if the mapping is conform the associated alignment management rules, completeness rules, correctness rules and consistency rules. Lastly, model compatibility is formally defined as:

Model Compatibility: Two models MS_1 and MS_2 are compatible via a mapping HMAP if it is true that HMAP $\models$ RS_{HMAPAM}, MS $\models$ $\mathrm{RS}_{HMAPACM}$, MS $\models$ $\mathrm{RS}_{HMAPACR}$ and MS $\models$ $\mathrm{RS}_{HMAPACN}$

stating that models MS_1 and MS_2 are compatible via a mapping HMAP if the mapping is conform the associated compatibility management rules, completeness rules, correctness rules and consistency rules.

6 Related Work

When it comes to service composition and business collaboration in general, most work has focused on development without taking adaptability into too much consideration. Current solutions like BPEL [11] and ebXML BPSS [16] are pre-determined and pre-specified, have narrow applicability and are almost impossible to reuse and manage. The same applies to works from academia like from workflow [1, 5], system development [6, 35] and enterprize modeling [37].

Relevant work in [3] and [19] describe a generic mechanism for defining WS-Policy based policies (e.g. in [14]), but only web service based rule specification is supported. Also, only rules in participant public behavior aspect are considered. [2] describes a way to establish WS-Agreements between service providers and requesters, but business and technical details are mixed. [9] presents a web service management architecture, however, its metrics cannot capture high level business requirements. [38] describes the rule inference framework DYflow, but there is no clear separation between technical and business rules.

In comparison our work provides a systematic way of specifying management rules for business collaboration in the BCCF context. Moreover, the business collaboration development process is driven by these management rules to model the different behaviors of enterprizes; where the process is constrained by construction rules in terms of 1) conformance and consistency of models, 2) alignment of strategic, operational and service models, and 3) compatibility among different models describing private, exposed and agreed upon behavior; and where it is partially automated by employing construction rules in a derivative fashion.

7 Conclusions

Current standards in business collaboration design, due to their pre-defined and inflexible nature, are precluded from accommodating business dynamics. The challenge is thus to provide a solution in which business collaboration development can be done in an flexible and adaptive manner.

In this paper we presented a rule driven approach for business collaboration development. We briefly introduced the Business Collaboration Context Framework (BCCF), which provides a highly modularized context for business collaboration development and management; and described how this context can be described using modeling. Subsequently we explained how rules drive, control and further the development and management process to facilitate flexible and adaptive business collaborations.

Work for future research will foremost be focused on completion of the reported prototype Icarus. Concurrently, extension of support for non-functional properties is a near-future goal.

References

[1] W. van der Aalst et al, Business Process Management: A Survey, *Proceedings of the International Conference on Business Process Management, 2003*.

[2] A. Andrieux et al, Web Services Agreement Specification (WS-Agreement), *http://www.gridforum.org/Meetings/GGF11/Documents/draft-ggf-graap-agreement.pdf, June 2004*

[3] S. Bajaj et al, Web Services Policy Framework (WS-Policy), *http://www-106.ibm.com/developerworks/library/specification/ws-polfram/, September 2004*

[4] Harold Boley, Integrating Positional and Slotted Knowledge on the Semantic Web?, *http://www.cs.unb.ca/ bspencer/cs6795swt/poslintweb-talk-pp4.pdf, September 2004*

[5] J. Bowers et al, Workflow from within and without, *Proceedings of the 4th European Conference on CSCW, 1995*

[6] P. Bresciani et al, Tropos: An Agent-Oriented Software Development Methodology, *Autonomous Agents and Multi-Agent Sytems, Vol. 8, No. 3, pp. 203-236, 2004*

[7] D. Burdett et al, Web Service Conversation Language *http://www.w3.org/TR/ws-chor-model/, March 24, 2004*

[8] Business Process Modeling Initiative, Business Process Modeling Language, *http://www.bpmi.org, June 24, 2002*

[9] F. Casati et al, Business-Oriented Management of Web Services, *Communications of the ACM, Vol. 46, No. 10, pp. 55-60, 2003*

[10] E. Christensen et al, Web Service Description Language, *http://www.w3.org/TR/wsdl, March 15, 2001*

[11] F. Curbera et al, Business Process Execution Language for Web Services, *http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/, July 31, 2002*

[12] B. Curtis et al, Process Modeling, *Communications of the ACM, Vol. 35, No. 9, pp. 75-90, 1992*

[13] W. Deiters et al, Flexibility in Workflow Management: Dimensions and Solutions, *International Journal of Computer Systems Science and Engineering, Vol. 15, No. 5, pp. 303-313, September 2000*

[14] G. Della-Libera et al, Web Services Security Policy, *http://www-106.ibm.com/developers/library/ws-secpol/, 2002*

[15] R. Dijkman et al, Service-oriented Design: A Multi-viewpoint Approach, *International Journal of Cooperative Information Systems, Vol. 13, No. 4, pp. 337-368, 2004*

[16] ebXML, *http://www.ebxml.org*

[17] D. Fensel et al, The Web Service Modeling Framework WSMF, *Electronic Commerce Research and Applications, Vol. 1, No. 2, pp. 113-137, 2002*

[18] P. Grefen et al, CrossFlow: Cross-Organizational Workflow Management in Dynamic Virtual Enterprises, *International Journal of Computer Systems Science & Engineering, Vol. 15, No. 5, pp. 277-290, 2000*

[19] P. Nolan, Understand WS-Policy processing, *http://www-106.ibm.com/developerworks/webservices/library/ws-policy.html, 2004*

[20] Object Management Group, Model Driven Architecture, *http://www.omg.org/docs/ormsc/01-07-01.pdf, July 2001*

[21] B. Orriens, Business Collaboration Context Framework, *INFOLAB Technical Report Series (23), Tilburg University, January 2006*

[22] B. Orriens, Modeling the Business Collaboration context, *INFOLAB Technical Report Series (24), Tilburg University, January 2006*

[23] B. Orriens, Specification of Assessment Requirements for Business Collaborations, *INFOLAB Technical Report Series (28), Tilburg University, January 2006*

[24] B. Orriens, Specification of Payment Requirements for Business Collaborations, *INFOLAB Technical Report Series (27), Tilburg University, January 2006*

[25] B. Orriens, Specification of Quality Requirements for Business Collaborations, *INFOLAB Technical Report Series (26), Tilburg University, January 2006*

[26] B. Orriens, Specification of Security Requirements for Business Collaborations, *INFOLAB Technical Report Series (25), Tilburg University, January 2006*

[27] B. Orriens et al, A Rule Driven Approach for Developing Adaptive Service Oriented Business Collaboration, *Proceedings of the 3th International Conference on Service Oriented Computing, Amsterdam, The Netherlands, December 2005*

[28] B. Orriens et al, Establishing and Maintaining Compatibility in Service Oriented Business Collaboration, *Proceedings of the 7th International Conference on Electronic Commerce, Xi'an, China, August 2005*

[29] M. Papazoglou et al, Service-Oriented Computing, *Communications of the ACM, Vol. 46, No. 10, pp. 25-28, October 2003*

[30] C. Peltz, Web services orchestration: a review of emerging technologies, tools, and standards, *Hewlett Packard White Paper, January 2003*

[31] RosettaNet, *http://www.rosettanet.org*

[32] R. Ross, Principles of the Business Rule Approach, *Addison-Wesley, 2003*

[33] RuleML, *http://www.ruleml.org*

[34] A. Scheer, Architecture for Integrated Information Systems - Foundations of Enterprise Modeling, *Springer-Verlag New York, Secaucus, NJ, USA, 1992*

[35] P. Traverso et al, Supporting the Negotiation between Global and Local Business Requirements in Service Oriented Development, *Proceedings of the 2d International Conference on Service Oriented Computing, New York, USA, 2004*

[36] J. Yang, Web Service Componentization: Towards Service Reuse and Specialization, *Communications of ACM, Vol. 46, No. 10, pp. 35-40, October 2003*

[37] J.A. Zachman, A framework for information systems architecture, *IBM Systems Journal, Vol. 26, no. 3, pp. 276-292, 1987*

[38] L. Zeng et al, Flexible Composition of Enterprise Web Services, *Electronic Markets - The International Journal of Electronic Commerce and Business Media, Vol. 13, No. 2, pp. 141-152, 2003*