Modeling The Business Collaboration Context

Bart Orriens

January 18, 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 present the second component of our rule driven approach for business collaboration development and management. The approach provides a model driven approach to capture the context in which business collaboration takes place; where the benefit is that enterprizes are enabled to describe their cooperations with others in the form of models. This allows 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.

Infolab Technical Report Series, no. 24

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 second 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 Finally, we present conclusions in 5 and outline future work.

2 Example

To exemplify the ideas presented throughout this paper an example inspired by the case study in [17] 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 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. An overview of the framework is shown in Fig. 1.

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. For more information the reader is referred to [23, 22]. The latest version of the framework can be found in [20].

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. The behavior dimension encompasses three types of behavior captured in three corresponding so-called collaboration aspects (inspired by among others [14,29,34]): 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; and is expressed in the *conversa-*

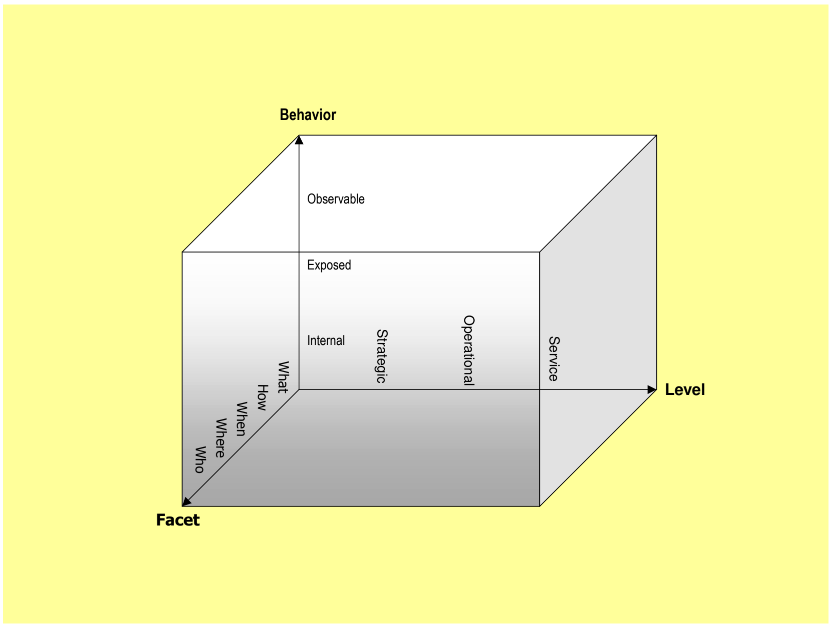


Fig. 1: Business Collaboration Context Framework (BCCF)

tion aspect. Captured in the *participant public behavior aspect* the exposed behavior describes how an individual participant can publicly behave in a business collaboration (i.e. its potential collaboration behavior). In contrast, the internal behavior (specified in the *internal business process aspect*) 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.

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 [19, 36]): 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. 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. The technical realization of operational behavior is done at the service level, describing how the services provided by the IT-infrastructure support the operational activities.

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, 33, 36]): *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.

4 Modeling in the BCDF

TThe 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:

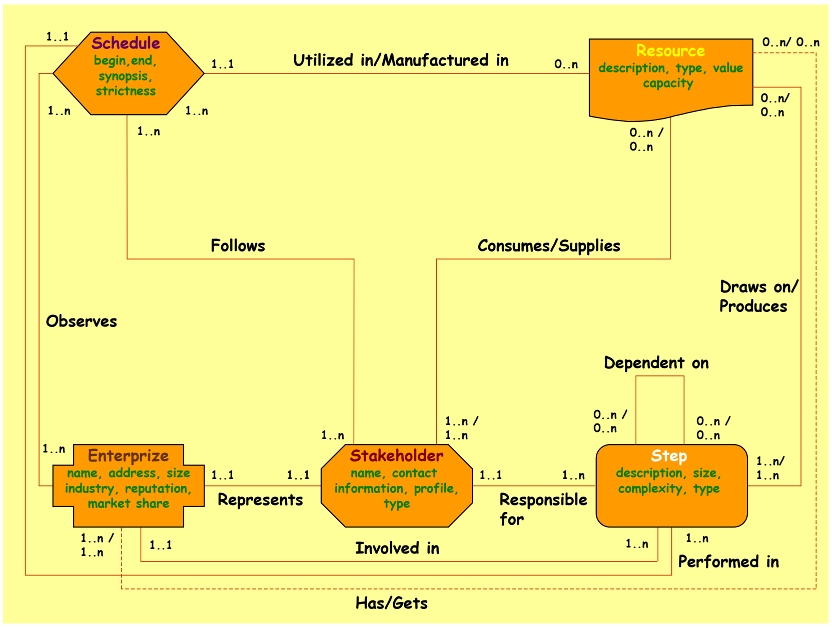
ES: the set of elements defined as $\{e_0 ... e_n\}$.
PS: the set of properties defined as $\{p_0 ... p_n\}$.
LS: the set of links defined as $\{l_0 ... l_n\}$.

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 (shown in Fig.2 to Fig.4); exemplified by snippets of exemplary models for the AGFIL application describing an interaction between `Garage Inc` and `Lee C.S` in Fig. 5. The collaboration meta models and models in the figures are loosely based on UML conventions. That is, in order to distinguish different facets, we represent them in different shapes in their models (see also legend in Fig. 5): *what* facet is shown as folded corners, *how* facet as rounded rectangles, *who* facet as octagons, *where* facet as plaques, and *when* facet as hectagons. In the following subsections we discuss the different models and mappings between them.



Schedule
begin, end, synopsis, strictness
Utilized in/Manufactured in
Resource
description, type, value capacity
1..1
1..n
0..n
0..n/ 0..n
Follows
Consumes/Supplies
Draws on/ Produces
Observes
Dependent on
Enterprize
name, address, size industry, reputation, market share
Stakeholder
name, contact information, profile, type
Step
description, size, complexity, type
Represents
Responsible for
Involved in
Performed in
Has/Gets
1..n /
0..n /

Fig. 2: Strategic Meta Model

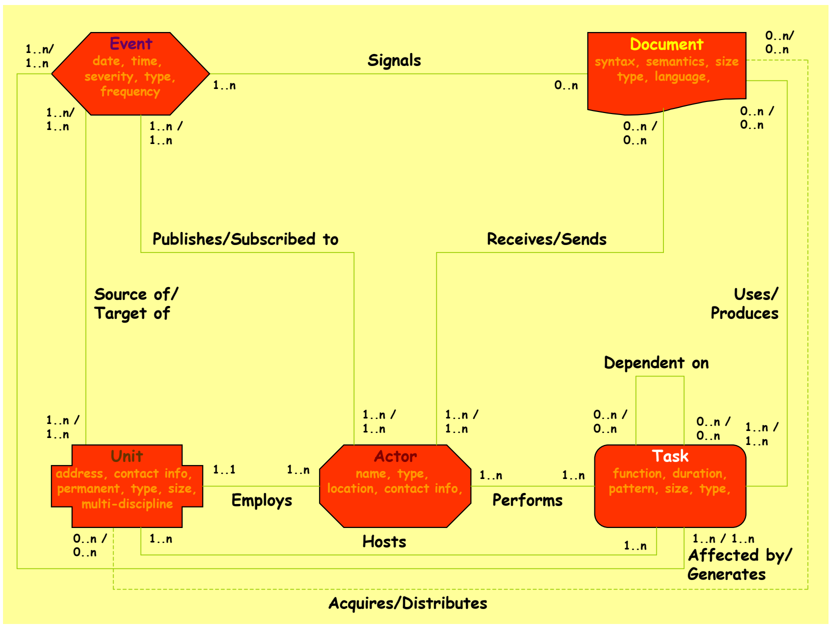


Fig. 3: Operational Meta Model

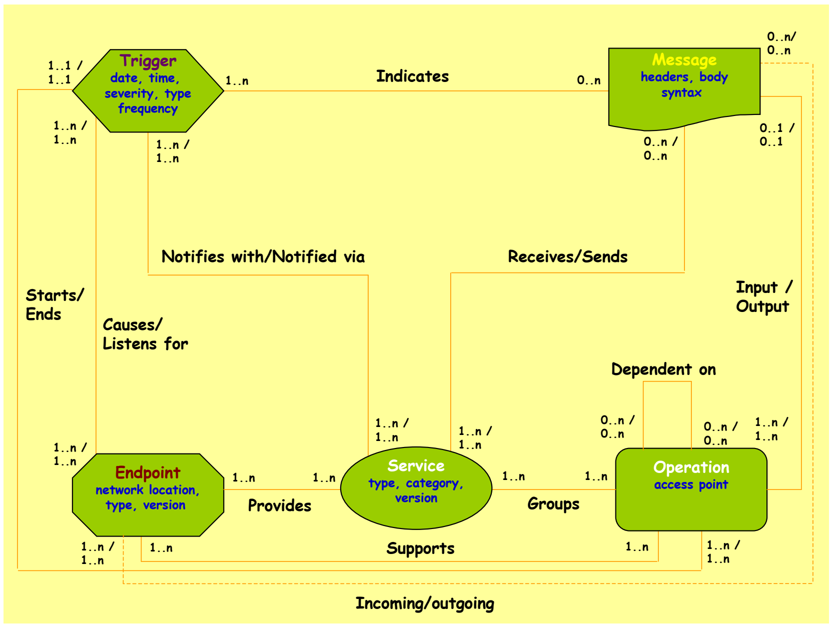


Fig. 4: Service Meta Model

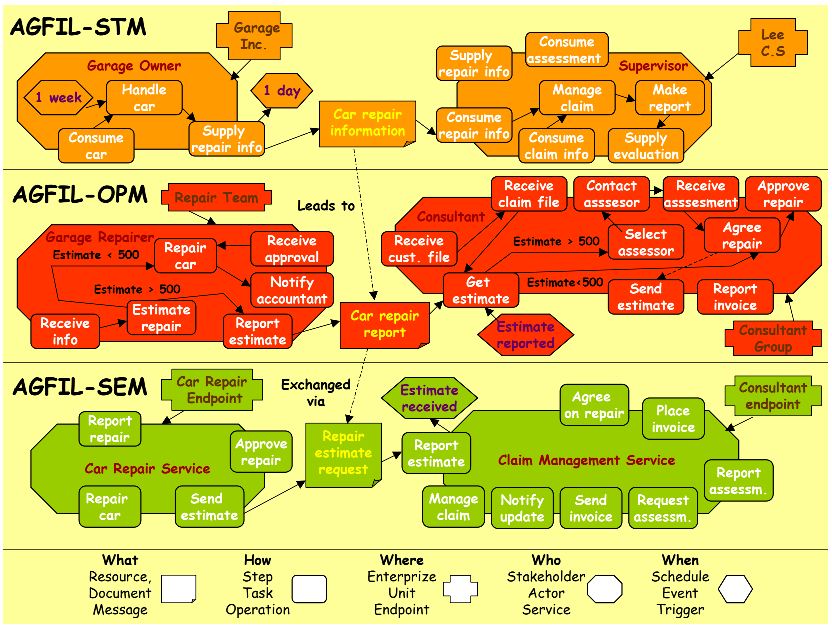


Fig. 5: AGFIL Collaboration Models

4.2 Strategic Models

At strategic level, strategic models like the AGFIL-STM in Fig. 5 capture purpose and high level requirements of business collaborations, akin to requirements analysis [6,34]. 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 (such as `car repair information`) is an instance of the **Resource** class. This class 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 instantiate the **Step** class and can be dependent on one another. Steps are of type 'internal' (like `handle car`), i.e. private to an enterprize (presented inside the stakeholder boundary in Fig.5); 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. Stake holders are instances of the **Stakeholder** class, where each stake holder belongs to an enterprize. Enterprizes are manifestations of the **Enterprize** class, and record the information about the participating enterprize where behavior is carried out. Stake holders and their enterprizes are bound by schedules reflecting temporal constraints, like the deadline of `1 week` for `handle car`.

The functional attributes of these five modeling classes have been enriched with several sets of non-functional (NF) characteristics; currently pertaining to assessment, payment, quality of service (QoS) and security. Assessment attributes are associated with schedules, indicating as to whether their time frames should be observed, progress logged and update notifications send out. With regard to payment, steps possess properties describing their cost, the means with which payment is to be done as well as the conditions applying to payment (e.g. that payment must be non-refutable, traceable, refundable, negotiable).

Quality of service parameters represent high level objectives regarding reliability, responsiveness, efficiency, accuracy, availability and accessibility; and are part of the **Step** class as well. In order to support definition and management of security conditions four attributes have been added, addressing the most common security threats: prevention of masquerading and unauthorized access (belonging to the **Step** class), and disclosure and modification (defined in the **Resource** class). For more information on the definition of non-functional attributes the reader is referred to [24, 25, 26, 27].

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. 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. 5 depict how high level strategic behavior is realized in terms of operational activities. These are expressed (as shown in Fig.3) 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, which together constitute abstract information containers as defined in the **Document** class.

Documents are used and produced by tasks, which represent specific business functions. Tasks instantiate the **Task** class and are of type 'internal' or 'communication' (represented inside or on the boundary of the actors respectively). Internal tasks constitute private activities, e.g. `collect claim form` done by `claim office employee`. Communication tasks involve receipt or sending of information like `report invoice`.

Actors such as `garage repairer` and `consultant` are responsible for carrying out tasks. Actors instantiate the **Actor** class and belong to units such as `repair team` unit, whose abstract definition is provided by the **Unit** class. In order to assess progress, keep logs to ensure non-repudiation, and etceteras, instances of the **Event** class are used. Events describe business occurrences which have properties such as 'date', 'time', 'severity'.

In addition to operationalizing the elements (and their functional properties) in strategic models, documents, tasks, actors, units, and events also operationalize the non-functional parameters defined at strategic level. In the context of assessment these properties reflect how monitoring, storage and publishing of events is to be done. When it comes to payment, they capture for each task the amount to be paid as well as how payment conditions are to be met (e.g. whether account or token based payment model is used, what type of refunding is done, and so on).

With regard to quality of service a wide range of NF operational parameters is associated with the **Task** class. These include among others failure rate, recovery rate, throughput, processing semantics and latency indicators, which facilitate evaluation of the high level QoS objectives. In a similar fashion security parameters can be set for tasks and documents to define the mechanisms to be used for authentication, authorization, confidentiality and integrity; which together address the security threats identified at the strategic level (see [24,25,26,27] 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 [30] or ebBPSS [15] 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 [15]. 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 based on the meta-model shown in Fig.4, and are consequently 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 (like parameters concerning reliable messaging, encryption styles, characters used, etc). The second type of information in messages are payloads, which 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`. Based on the abstract **Operation** class operations represent specific technical functions, and are described in terms of their access details as well as relevant non-functional properties. 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`, where the **Service** class describes these containers for collections of logically related operations. Services themselves are provided by endpoints (like `claim handling endpoint`) and have properties 'network location' and 'type'. To express technical occurrences triggers like `claim request acknowledged` can be defined on the basis of the **Trigger** class.

Besides the above the discussed classes encompass the sets of non-functional attributes required to translate NF operational parameters into a computational context. Assessment attributes at the service level are part of the **Trigger** class and define exactly how monitoring, storage and publishing is to be done (e.g. how long a trigger is to be stored, or what kind of push protocol is to be used for publishing). Similarly, payment attributes specify pricing information for an operation (like minimum and maximum price), what payment instrument is used (such as cash, credit card), payment granularity (payment per request, kg, hour), and etceteras.

In order to be able to measure the quality of service indicators established for tasks at operational level, operations at service level possess properties facilitating their measurement. Task latency e.g. is defined as the sum of the setup, queuing, execution and transport time of the operation(s) that support this task. In order to realize the security mechanisms of a task or document, at the service level operations and messages have properties that depict how this is to be done (like what proof of knowledge is required for proper authentication, e.g. a username/password combination). More information can be found in [24,25,26,27].

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 [29] 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 [29] (not shown in Fig. 5).

4.5 Horizontal Mappings Between Models

In subsection 3.1 we observed that the different collaboration aspects are interrelated. In order to enable enterprizes to manage these interrelationships, we utilize so-called *horizontal mappings*. 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 mappings between an internal and exposed behavior are based on the following notion: an internal behavior partly comprises private activities, and partly communication activities (that is, points where communication with the outside is required to further the execution of the private activities). This communication behavior must thus be defined in some manner in a corresponding exposed behavior. Each internal behavior can only have one corresponding exposed behavior.

Vice versa, all activities as defined in an exposed behavior must in some way be linked to activities in a corresponding internal behavior; i.e. communication with the outside must somehow originate from or have an impact on the private activities. Since an exposed behavior only reflects an internal behavior's communication part, an exposed behavior can correspond to many internal behaviors; where the private part of each internal behavior varies.

Between an exposed and observable behavior the mappings are grounded on a different rationale. An exposed behavior describes how a participant can potentially behave; whereas an observable behavior depicts how it will behave. This means that how the participant will behave, must be equal to or part of the way it can behave (i.e. a subset of the exposed behavior). This implies that an exposed behavior can be part of many observable behaviors; where in each observable behavior a participant's activities must be linked to those defined in its exposed behavior.

Looking at it from the perspective of the observable behavior, an observable behavior will correspond to exactly two exposed behaviors (since there are two participants involved in every binary agreement, as explained in subsection 3.1). For each participant it must be true that the behavior it promises to perform, is supported and thus present in its exposed behavior. In addition, the two exposed behaviors must mirror one other in terms of the displayed communication behavior in order to realize actual communication between the two participants. For example, if one participant sends something to the other, then the other must be able to receive it (and vice versa).

Based on the above we define the following horizontal mappings (categorized along facet):

- **How facet**

The communication steps, tasks and operations in an internal behavior (depending on the level) must be related to steps, tasks and operations in the corresponding exposed behavior. Such dependencies are made explicit using a *offeredAs* relationship. Each internal communication step, task and operation corresponds to exactly one exposed communication step, task and operation. Vice versa, each exposed communication step, task and operation is linked to at least one internal communication step, task and operation.

In a similar fashion *performedAs* links connect steps, tasks, and operations in an exposed behavior to those in an observable behavior. Steps, tasks, and operations in an exposed behavior can be linked to zero or more steps, tasks and operations in an observable behavior. Vice versa, steps, tasks and operations in an observable behavior must be linked to exactly one step, task and operation in the exposed behavior.

- **What facet**

  Resources, documents and messages originating from the internal behavior are linked to those in the corresponding exposed behavior via *sendAs* relations. This relation is one-to-one from the point of view of the internal behavior; and one-to-many from the perspective of the exposed behavior (identical to those discussed for the how facet).

  The connection from an exposed to an observable behavior is made in the what facet by utilizing *exchangedAs* links. The cardinality of these links is zero-to-many from the point of view of the exposed behavior; and one-to-one from the point of view of the observable behavior.

- **Who facet**

  The exponents of the who facet in the different collaboration behaviors, i.e. stake holders, actors and services, are linked via *exposedAs* and *participatesAs* relationships. The former connect stake holders, actors and services in an internal and exposed behavior; whereas the latter link those in an exposed and observable behavior. The multiplicities are as discussed for the other facets.

- **Where facet**

  In the where facet enterprizes, units and endpoints are linked from one collaboration behavior to the other by defining *knownAs* and *involvedAs* relations between an internal and exposed, and an exposed and observable behavior respectively. These relations exhibit the same cardinality as the ones discussed so far.

- **When facet**

  publishedAs and *definedAs* links are employed to relate modeling elements representing the when facet in the different collaboration behaviors. Schedules, events and triggers in an internal behavior are published in the exposed behavior in a one-to-one fashion (the reverse cardinality is one-to-many); whereas these themselves are defined in observable behaviors in a zero-to-many manner. Vice versa, the multiplicity of this relation is one-to-one.

Before we conclude a number of final remarks: firstly, it must be noted that modeling elements in one collaboration behavior can be very different from those in another. For example, a document that is send from an internal behavior to the outside may be completely different in e.g. syntax, semantics, content and language from the corresponding document in the exposed behavior. Another illustration of this phenomenon is that an exposed operation internally supports several authentication protocols, however, in the corresponding observable behavior only one of them is offered to other parties.

Secondly and related to the previous, in the above mappings we have not taken individual properties into consideration. Naturally dependencies among them exist as well, e.g. that the price of an operation internally must not be higher than the price offered to outside parties. Similarly, the schedules mandated in a strategic agreement should not be more strict than what can be supported by the participants involved. These types of relationship affect the validity of horizontal mappings; and are informally discussed for non-functional properties in [24,25,26,27].

4.6 Vertical Mappings Between Models

For the specification of dependencies between different collaboration behaviors at different levels, we employ vertical mappings. Vertical mappings are realized by providing links between the classes in different metamodels 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. Similarly, the mappings connecting an operational and service behavior reflect how operational activities are supported by the IT-infrastructure. Here as well the mapping relation is many-to-many, that is, an operational behavior can be realized in different service behaviors; vice versa a service behavior may realize multiple operational behaviors.

Based on the above we define the following mappings:

- **What facet**

  Resources at strategic level are mapped to documents at operational level via a *leadsTo* relation (like `car repair information` leading to `car repair report` in Fig.5). Each resource is mapped to at least one document, whereas each document can be mapped to multiple resources. The idea is that when a resource is exchanged this is accompanied by an information flow. It is this information flow that is defined in terms of document(s) at operational level.

  Documents themselves are mapped to messages using *exchangedVia* relations, which reflect how the information flow is supported in the IT-infrastructure (`car repair report` is exchanged via `car repair request` for example). A document is exchanged via at least one message, where each message can facilitate exchange of multiple documents.

- **How facet**

  Steps represent high level, vaguely defined strategic activities. The underlying notion is that by mapping them to tasks these steps are operationalized, e.g. `supply repair information` is mapped via a *decomposedInto* relation to `report estimate`. Each step is decomposed into at least one task; and one task can belong to multiple steps.

  Tasks are themselves mapped to operations using *realizedBy* relationships, e.g connecting `report estimate` with `send estimate`, indicating how they are supported via a technical service provided by the IT-infrastructure. A task is realized by one or more operations; vice versa an operation can help realize multiple tasks.

- **Who facet**

  The stake holders at strategic level are mapped to actors at operational level via *controls* links. The purpose is to make explicit how a stake holder delegates the realization of the high level steps it is responsible for to the actors it controls (such as `garage owner` delegating car receipt and repair to `garage repairer`). A stake holder will typically control multiple actors; whereas each actor may be controlled by multiple stake holders.

Actors themselves are portrayed as services at service level, like `garage repairer` connected to `car repair service` via a *representedBy* mapping. Each actor can be represented in terms of one or more services, where each service can represents exactly one actor.

- **Where facet**

  Enterprizes are organized in units at operational level using *organizedIn* links, like `Garage Inc` having a unit `repair team`. This mapping mimics the manner in which enterprizes have set up their the organizational structure. Following the latter an enterprize usually constitutes many units; with each unit belonging to one enterprize.

  At the service level `repair team` is mapped to `car repair endpoint`, expressing that this unit is responsible for providing this endpoint in the IT-infrastructure. To capture such a relationship *offers* mappings are utilized, where each unit can offer many endpoints; but each endpoint is offered by exactly one unit.

- **When facet**

  Schedules expressing temporal requirements at strategic level are *splitInto* events representing concrete business occurrences, e.g. `estimate reported` indicating a claim has been made. One schedule is typically split into multiple events; where an event can be part of multiple schedules.

  Events are themselves mapped to triggers like `car estimate reported` *causes* `car estimate received`. Usually an event is mapped to many triggers, i.e. before a business event occurs multiple lower level things must have taken place. A trigger itself can be mapped by multiple events.

Before we conclude the above discussion, two remarks need to be made: firstly, we wish to stress that the mapping of actors to services encompasses both human and non-human actors. Taking `garage repairer` as an example, this is clearly an example of a human actor. However, due to the self-contained nature of services we can represent this actor as a `car repair service`. Internally this service will be performed by a human being, however, from the point of view from the IT-infrastructure it is just another service being offered (be it by a computer or a human). This allows enterprizes to develop and manage both automated and non-automated behavior in the same manner in our approach.

Secondly, in the above we have not discussed the matter of mapping individual properties at one level to those on another level. Obviously such dependencies do exist, for example the begin date of any event belonging to a particular schedule must not be prior to the start of this schedule. Another example is that the choice for certain payment parameters at strategic level, like that payment is to be negotiable, means that at operational level the pricing style must be interval based in order to allow negotiation; which in turn implies that at service level both a minimum and maximum price are specified. These types of relationship affect the validity of a mapping; and are informally discussed for non-functional properties in [24,25,26,27].

5 Conclusions

In this technical report we introduced a model driven approach for capturing the context in which business collaboration takes place. This work is motivated by the lack of cohesive approach thereof in the current research with regard to modeling different parts of business collaborations in a comprehensive and consistent manner; as most work (like [11,15,6]) focuses on parts of the whole context without considering their role in the overall picture.

To remedy this situation we introduced our model driven approach, which is based on the Business Collaboration Context Framework (BCCF). We explained how we use meta models and models to capture the different dimensions identified in the BCCF; and also showed how dependencies between aspects, levels and facets can be made explicit. The modeling approach has been published in several papers among others [21,22,23].

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] 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*

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

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

[17] 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*

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

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

[20] B. Orriens, Business Collaboration Context Framework, *INFOLAB Technical Report Series, No. 27, Tilburg, The Netherlands, January 2006*

[21] B. Orriens et al, Specification and Management of Policies in Service Oriented Business Collaboration, *Proceedings of the International Conference on Business Process Management, Lille, France, September 2005*

[22] 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*

[23] B. Orriens et al, Bridging the Gap between Business and IT in Service Oriented Business Collaboration, *Proceedings of the IEEE International Conference on Services Computing, Orlando, Florida, USA, July 2005*

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

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

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

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

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

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

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

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

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

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

[34] 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*

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

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

[37] 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*