Modeling and Managing Service Oriented Business Collaboration

Bart Orriens
Department of Information Systems And Management
Tilburg University, Tilburg, The Netherlands
Email: b.orriens@uvt.nl

Jian Yang
School of Information And Communication Sciences
Macquarie University, Sydney, Australia
Email: jian@ics.mq.edu.au

I. Introduction

Nowadays enterprizes need to be dynamic and adaptive in order to stay competitive. This has led to an increasing demand for providing business services that can adapt to changes in the area such as market conditions, organizational policies, usage scenarios, etc. Recently there has been increasing focus on service oriented computing, the new emerging paradigm for distributed computing and e-business processing, to deliver flexible and adaptable corporate business services by utilizing existing services cross organizational boundaries. *business collaboration* here refers to a cooperation between multiple enterprizes working together to achieve a business goal.

In order to realize the vision of developing business collaborations in an adaptive fashion while adhering to the requirements imposed by the business environment, the specifics of business collaborations must be properly captured and modeled. Business collaboration design needs to apply software development principles and at the same time incorporate the special requirements of modern business collaboration development, i.e, support for high (abstract) level specification and adaptive to market changes. Therefore methodologies, modeling languages, techniques and tools are required to support designers to develop and deliver business collaborations in a effective and manageable way. Furthermore, modifications to existing designs must be verifiable to determine if the resulting collaborations are still in accordance with current market conditions, government regulations, industry guidelines, internal policies and so on. This requires the modeling languages to facilitate the specification of both business and technical demands for a business collaboration, as well as dependencies among them.

However, current composite web service development and management solutions including the defacto standard BPEL4WS [6] are too low-level and not rich enough to capture both the business and technical requirements of relevance to business collaborations. As a result management of service composition in relation to changing requirements based on existing technologies and standards is often difficult. In this paper we introduce a **Business Collaboration Design Framework (BCDF)** for designing business collaborations, which provides a systematic way for analyzing the requirements and modeling the activities involved in business collaboration development. With the development of this framework we set out to achieve (among others) the following goals:

- Provide a holistic approach for business collaboration design with which designers can describe a collaboration regarding its high level business and technical requirements for different contexts; as well as define these requirements from different standpoints.
- Enable the explicit specification of dependencies between different types of requirement in order to enable the traceability required for the verifiability and changeability of collaboration designs.
- Aid designers in the development of business collaboration designs by employing a model driven approach to make possible the rapid development and delivery of business collaborations based on proven and tested models.

In this paper we conceptually introduce the ideas underlying the BCDF, where we emphasize their usage in the context of changeability. The paper is structured as followed: First we introduce a multi-party example in section II, which we will use for illustrative purposes. Then the framework BCDF is introduced in section III. Next in section IV we analyze different types of changes that may occur in the context of business collaboration design based on the BCDF framework. In section V different models for business collaboration design are presented. Subsequently, in section VI we discuss how changeability of collaboration designs is supported in the BCDF framework, and match this against the requirements identified in section IV. Related work is presented in section VII followed by conclusions in section VIII. [1]

II. Example

To exemplify the ideas presented throughout this paper an example inspired by the case study in [14] 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

[1]This work was partially funded with an UNSW Australian ARC Discovery Research Grant

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.

III. Business Collaboration Design Framework

The BCDF framework employs a multi-layered approach for business collaboration design based on three orthogonal concepts *perspectives*, *facets* and *aspects*. The design perspectives represent different levels of abstraction in a business collaboration design, while design facets depict the elements in a business collaboration design which have different contexts when observed from different design perspectives. Design aspects accommodate different design considerations, i.e., collaboration, external and internal views of the participants. An overview of the Business Collaboration Design Framework is provided in Fig.1, which is briefly discussed in the following subsections.

A. Design Perspectives

The first dimension shown in Fig. 1 is *perspective* which capture design requirements at different level of abstraction supporting the concept of 'separation of design concern' [3], [15], [22]. Three design perspectives are identified in BCDF (see Fig.1): 1) *business perspective*: from which the purpose and the requirements of the business collaboration is modeled and specified in terms of **Goal, Schedule, Resource, Enterprize, Stakeholder, Step** (see Fig.2). 2) *conceptual perspective*: from which a computational independent conceptual model is generated that depicts the business activities involved in the collaboration in terms of **Rule, Event, Document, Unit, Actor, Task** (see Fig.4). 3) *logical perspective*: from which a service-oriented business collaboration is modeled in terms of **Constraint, Trigger, Message, Endpoint, Service, Operation** (see Fig.6). Together these three perspectives encompass the business, operational and technical side of business collaborations. They can be used for design in a top-down or bottom-up manner, corresponding to forward and reverse engineering in system development respectively. In addition to the specification provided from these three perspectives, mappings between perspectives are provided to facilitate alignment between business, operational and technical requirements.

B. Design Facets

The second dimension is *facet* which depicts the elements in a business collaboration design that have different contexts when observed from different development perspectives [9], [19], [22]. Facets emphasize the specification of different business description elements: *what, how, where, who, when, why*. Here we refer them as *Design Facets* in the context of business collaboration design. In Fig.1 the *what* facet emphasizes the informational view of the business collaboration. The functional standpoint is taken in the *how* facet, which focuses on how things are done. The geographical facet of the collaboration is expressed in the *where* facet, whereas the *who* facet concerns about collaboration's participants. The temporal aspect is covered in the *when* facet that deals with schedules and events. Finally the *why* facet concentrates on describing the rational behind a collaboration. The facets

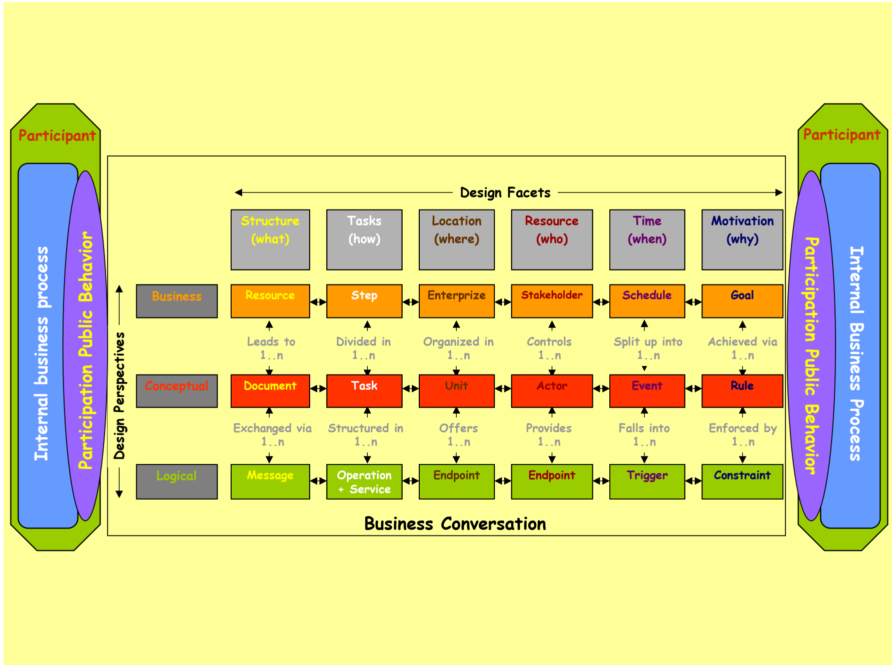

Fig. 1. Business Collaboration Design Framework (BCDF)

and their interactions provide a complete coverage for each individual design perspective.

C. Design Aspects

The third dimension is *aspects* which represents the different standpoints of a business collaboration [11], [20]: *internal business process aspect* describing private behavior of a participant; *participant public behavior aspect* describing how a participant can externally behave in a collaboration; and *business conversation aspect* describing the expected conversation between participants. All design aspects are expressed using the same modeling language for individual design perspectives, so that designers can define different roles of business collaborations in an uniform manner. As such they allow designers to express all standpoints of relevance for a collaboration at any given perspective. They also enable designers to focus on particular positions without losing sight of their relationship with other positions.

IV. Change and Changeability

In order to provide support for maintenance, it is important to get a clear picture of what types of changes may occur in the business collaboration, where they can occur, how they impact the collaboration, and how modifications can be applied. Change in general can be regarded as ”when something becomes different” [7]. In the context of business collaboration change can manifest itself in a wide range of areas (see e.g. [2] and [10]). In the BCDF changes can occur at any of its perspectives, facets and aspects. Within each perspective changes may manifest themselves in any of the design facets. For example, changes at the business perspective can range from an enterprize adopting new business strategies due to shifting customer demands; or changes of resources because of resource scarcity. At the conceptual perspective the adjustment of business policies and the re-sequencing of tasks are exemplary sources of changes. Service unavailability and adoption of new technical protocols are illustrative for changes at the logical perspective. All these changes can occur in any of the design aspects, for example, adoption of new communication protocols by a participant in the overall collaboration can result in loss of communication with other participants.

Key to the facilitation of changes in the BCDF is the understanding that when a change happens at a particular perspective, it has to be propagated up and/or down to the other perspectives to maintain the consistency between the perspectives. We refer to this as the *vertical consistency*, whereas the capacity of self-managing vertical consistency is called *vertical changeability*. Similarly, within a single perspective there must be a way in which a change in one facet is propagated to the other facets, which is required to ensure *horizontal consistency*. Again the capability of self-managing horizontal consistency is called *horizontal changeability*. Finally, the capacity of *aspect changeability* is required to propagate changes from design aspect to another in order to maintain consistency between the different aspects.

V. Modeling in the BCDF Framework

There are two types of models in BCDF: meta models and instance models, both of which are defined from individual perspectives. The meta models are generic which provide design guidelines; while the instance models represent a particular design of an application, which have to conform to the meta-model in terms of the properties and associations. There are three meta-models, each of which describes the relevant elements of (six) design facets as classes and their relationships for a particular design perspective. The relationships connect the classes to indicate the interactions between the design facets. Meta-models for the business, conceptual and logical perspective and their instance models are introduced in subsections V-A through V-C respectively. In this paper all the models are represented based on UML conventions.

A. Business Perspective

The design activities in the business perspective are of a scoping and discovering nature. *Scoping* is reminiscent to early requirements analysis in software development, where the objective is to get an idea of the goals of a collaboration in order to establish its boundaries. *Discovery* is comparable to requirements analysis in software development, in which a description of the collaboration-to-be within its operating environment is established, and its major functions are linked to the goals. The purpose of these activities is to define one of all possible models conceivable in the **Business Meta Model (BMM)**, where the resulting model is called a **Business Model (BM)**. The classes and associations in the BMM are portrayed in Fig.2. In order to distinguish different design facets, we represent them in different shapes in their UML models. In the following we briefly discuss the BMM using examples from the AGFIL-BM in Fig.3.

Classes need to be instantiated during design. The main class to be instantiated in the business meta model is the *Goal* class, whose instances are concrete goals in an application. Goals represent desirable aims such as `24hr receipt, manage claims`.

Goals are pursued by stake holders (e.g. `insurance director`) who serve for the participating enterprizes. These goals will originate for example from an enterprize's policies. A stakeholder can be interested in achieving multiple goals. The information about the participating enterprize is captured in the *Enterprize* class.

Resource instances such as `customer plus car` provide an abstraction mechanism for means such as financial, human and informational capital. Resources can be scarce, which affects the feasibility of usage scenarios. Compromises may need to be made in which strategic considerations come into play.

Goals are achieved through steps which represents high level functions such as `process claim`. Steps can be dependent on one another or contain other steps to form complex ones. `consume claim` must have been completed before `process claim` can take place. Steps are of type 'internal' or 'exchanged'. Internal steps like `process claim` are specific to an enterprize and are not observable by others, which are presented inside the stakeholder boundary in Fig.3. Exchange steps provide a mechanism for defining the supply or consumption of resources between stake holders (for example `consume claim information`).

Steps can have schedules linked to them, reflecting applicable temporal constraints. For example `process claim` has to be carried out at most 1 day after it has been received, as defined in `1 day > performance`. Schedules have a begin and end date to define a period of time in which a step is to be completed. Schedules for different steps must be synchronized to avoid conflicts, so that, for example, the schedule of `process claim` does not interfere with that of `supply claim information`.

Business models like the AGFIL-BM capture purpose and business requirements of business collaborations, akin to requirements analysis [3], [20]. In business conversation aspect (all modeling elements external to or on boundary of stakeholders like `garage owner`) the exchange of resources like `car repair information` between enterprizes is defined. Participant public behavior aspect (all elements at border of stakeholder like `Lee C.S`) specifies business capabilities of individual enterprizes such as `consume car`, whereas internal business process aspect (inside particular stakeholders) describes private enterprize activities (e.g. `handle car`).

B. Conceptual Perspective

At the conceptual perspective analysis is performed to study the collaboration and investigate its properties. As the result one possible instance model, called **Conceptual Model (CM)**, is produced which is encompassed by the **Conceptual Meta Model (CMM)**. The CMM is portrayed in Fig. 4, whereas an example AGFIL-CM can be found in Fig. 5.

As illustrated in the AGFIL-CM in Fig.5 actors such as `claim office employee` and `consultant` interact with each other to perform tasks. Tasks 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` performed by `claim office employee`. Communication tasks involve receipt or sending of information like `report invoice`. Actors belong to units, which provide organizational grouping constructs (like division or project team). `claim handling` unit is such an example.

Events are used to assess progress, keep logs to ensure non-repudiation, etc. Events describe business occurrences which have properties such as 'date', 'time'. Events can trigger, modify, pause, resume or end a task. Events can be inward oriented affecting internal tasks, or outward directed in which case they influence communication tasks. `claim received` is an example of an internal event, whereas `estimate send` illustrates the usage of external events.

Events, as displayed in Fig.4, are signaled by the receipt or sending of documents, like `estimate send` being signaled by `estimate`. Documents are communicated between actors

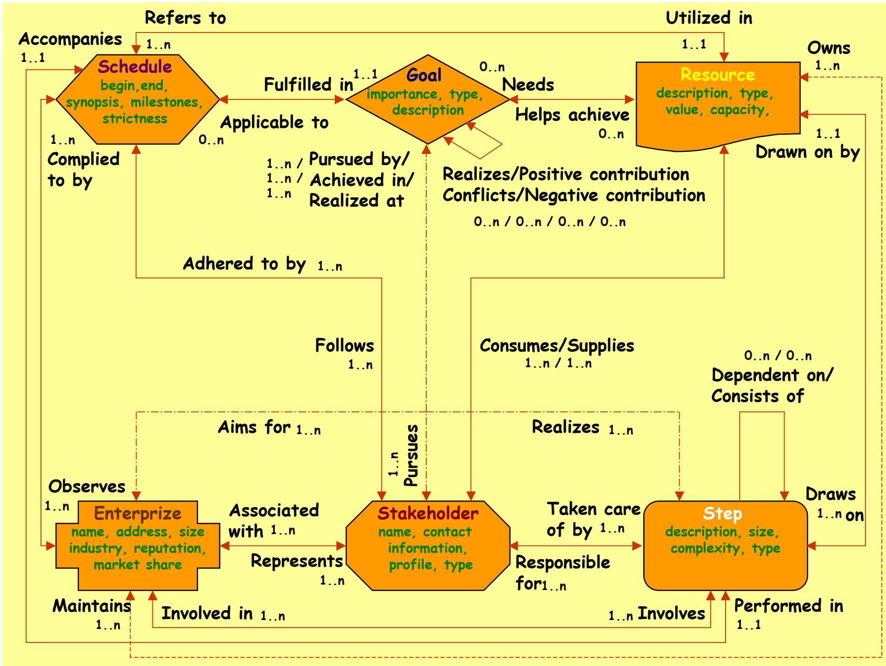

Fig. 2. Business Meta Model (BMM)

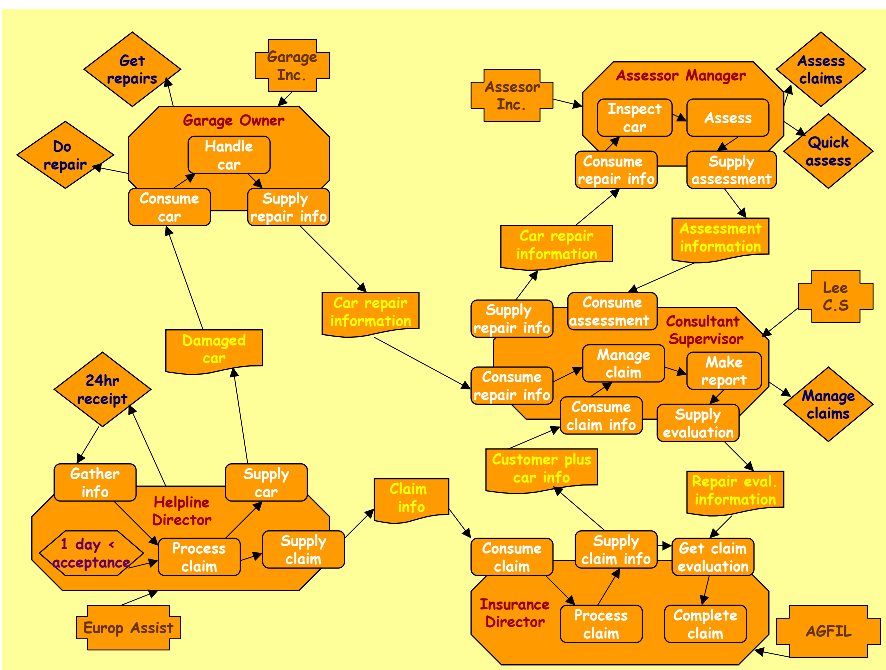

Fig. 3. AGFIL Business Model (AGFIL-BM)

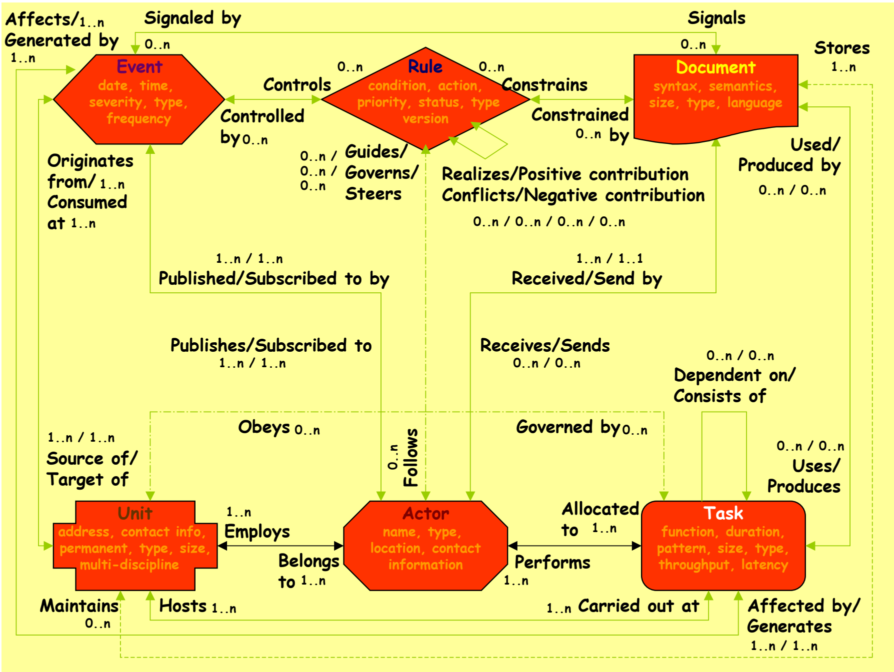

Fig. 4. Conceptual Meta Model (CMM)

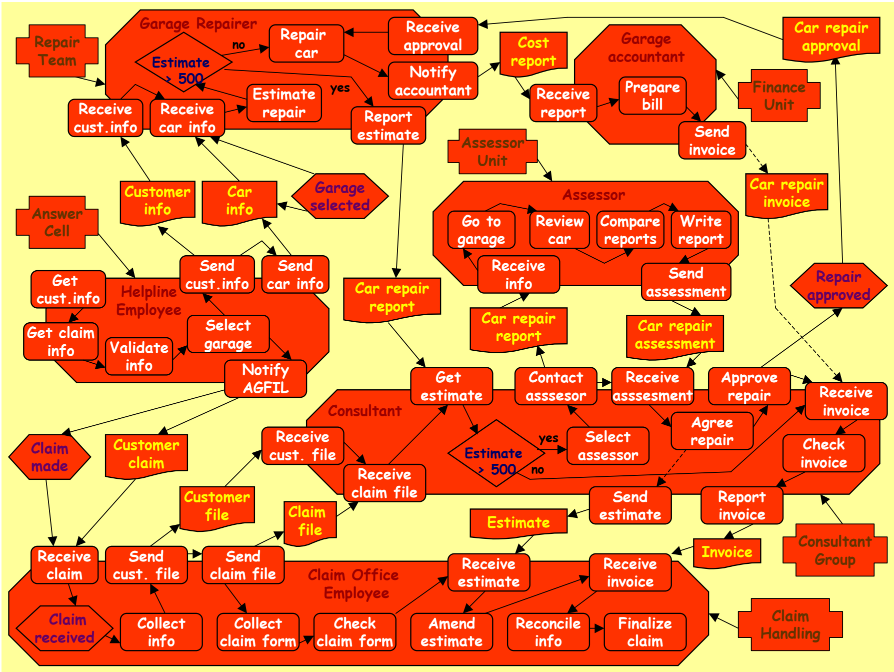

Fig. 5. AGFIL Conceptual Model (AGFIL-CM)

to further their own state as well as that of the overall collaboration, e.g. communication of `invoice` between `claim office employee` and `consultant`. Documents convey information that is stored for example as business objects.

All classes, properties and associations in the CMM can be constrained by rules. Rules comprise statements that define or constrain some aspect of the business, which are intended to assert business structure or to control or influence the behavior of the business [5]. An example of a rule regulating the performance of tasks is `if estimate > 500, then select assessor`, which is followed by `consultant`. Other rules may constrain event occurrences, actor selection, etc. Rules form operational policies guiding the everyday business of enterprizes.

The conceptual model like AGFIL-CM describes a collaboration's operational properties. In terms of aspects, in business conversation aspect (all elements on or outside actor borders e.g. `consultant`), communication of documents between actors is defined; like specified by RosettaNet [18] or BPSS [13]. In participant public behavior aspect (e.g. elements on border of `garage repairer`) the tasks an actor can perform are depicted e.g. `get estimate`; whereas internal business process aspect (elements within actor) is similar to e.g. BPML [4] or workflow [1], specifying how and when activities such as `estimate repair` are conducted.

C. Logical Perspective

A conceptual model is realized in the logical model where the collaboration collaboration is put into a computational context, which is referred as service oriented computing (SOC) in this paper. The **Logical Meta Model (LMM)** is illustrated in Fig.6. A snippet for the AGFIL example is depicted in Fig.7 as one possible logical model (LM) of LMM.

Actors involved in the AGFIL-CM are represented as 'endpoints' in the context of service oriented computing. Endpoints such as `claim handling endpoint` have properties 'network location' and 'type'. Endpoints can be specified in the abstract or in the concrete, and are the access points through which services are requested and/or provided. For example `claim handling endpoint` provides `claim handling service`, whereas requiring `claim management service`.

Services expose their functionality through interfaces, prescribing the conditions under which other endpoints can access them. An example is `claim management service` in Fig.7 provided by `consultant endpoint` to allow clients to request claim management. Services provide containers for collections of logically related operations. Operations such as `place invoice` are specific business functions, which process access information as well as non-functional properties.

Inputs and outputs of operations are represented as messages, which represents containers of information (e.g., `claim management request`), consisting of meta-data and actual data. Meta-data comprises the information that is required to deliver the message and enable its processing. Examples include parameters concerning reliable messaging, encryption styles, digital signatures, the character scheme 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).

The receipt and sending of messages result in triggers, which express a relevant system occurrence e.g. `claim request acknowledged`. They are similar to events in the CMM (see Fig.4), however here the emphasis is on monitoring the progress from a computational point of view. Triggers can be controlled by constraints. Constraints have properties such as 'condition', 'action' and 'status' (as shown in Fig.6) and they can constrain any element of a LM (e.g. on an operation's performance parameters). Once a complete LM is constructed, the design can be mapped to a BPEL representation.

Logical models depict how activities in conceptual models are realized by services and their operations. In the business conversation aspect (the elements on or outside the border of services) is akin to the notion of choreography [17] defining message exchanges among services. Resembling interface behavior in [11], the public participant behavior aspect is captured in models formed by elements placed on the border of individual services like `car repair service` depicting offered operations. Within a service the modeling elements depict internal business process aspect akin to orchestration; where a service internally engages other services to realize its functionality (not shown in Fig. 7).

D. Mappings

Mappings between perspectives are achieved through linkage of classes representing the same facet. To start with, the work a *stakeholder* promises to fulfill in a BM will be delegated to its *actors*, such as `garage owner` delegating car receipt and repair to `garage repairer`. *enterprizes* associated with stakeholders are organized in `units` to which actors belong, like `Garage Inc` having a unit `repair team` of which `garage repairer` is part. *Actors* are portrayed as *endpoints* in a LM, where an actor can offer multiple endpoints (offering different services) like `assessor` offering `assessment endpoint`. An endpoint also encompasses the *where* facet in the logical perspective (as services are location transparent), e.g. `consultant group` being represented in `assessment endpoint` as well.

Resource trades between stakeholders are mapped to exchanges of *documents* between actors, like trading of `car repair information` leading to communication of `car repair report` and `car repair invoice`. *Steps* associated with resource trade are decomposed into specific *tasks*, e.g. `supply repair information` is divided into `report estimate` and `send invoice`. Communication of documents between actors is mapped to sending and receiving of *messages* among endpoints, like `customer claim` resulting in messages `claim processing request` and `claim processing response`. *Tasks* corresponding to document exchanges are decomposed into *operations*, e.g `collect info` is further divided in `get claim`

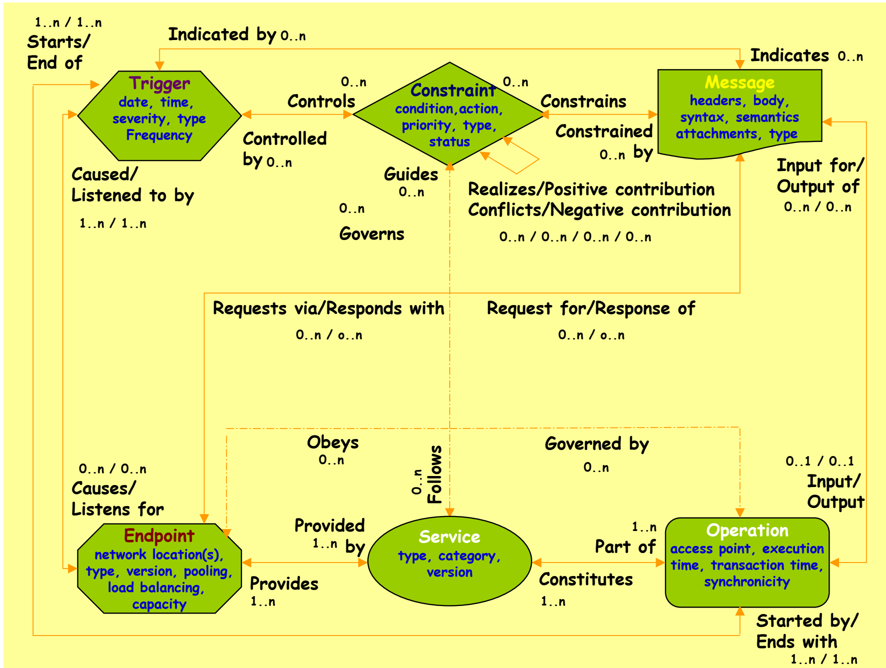

Fig. 6.   Logical Meta Model (LMM)

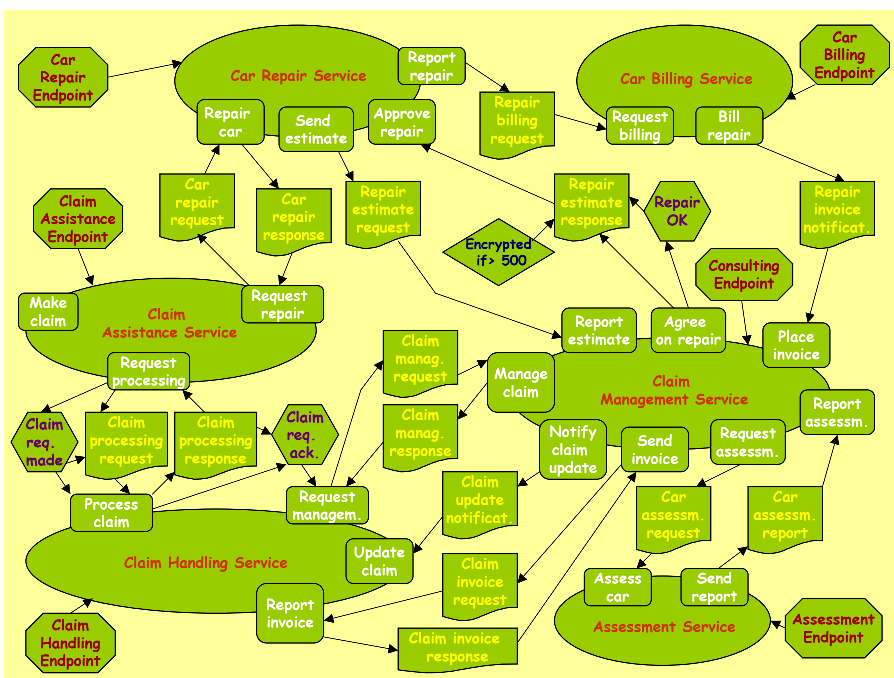

Fig. 7.   AGFIL Logical Model (AGFIL-LM)

`history` and `get customer info`.

Schedules constraining steps with regard to temporal requirements are split into *events* representing concrete business occurrences, e.g. `claim made` indicating a claim has been made. *Events* occurring during business activities are mapped to *triggers* in the computing context. To exemplify, `claim made` is captured `claim request made` and `claim request acknowledged`. Lastly, achievement of *goals* pursued in steps is enforced via *rules*, like `quick assessment` resulting in `if estimate > 500, do assessment`. In turn, *rules* are implemented via *constraints*, such as `estimate > 500, assessment` being realized by (among others) `request assessment constraint`.

VI. Changeability

Changeability (automatic or manual) requires traceability, while traceability requires a mechanism to identify all the affected areas introduced by the change. In this section, we shall explain how change management is supported in the BCDF framework.

A. Changeability along perspectives

The design perspectives in the BCDF are different in terms of level of abstract and content. However, for the same business collaboration they are related to one another. These relationships need to be made explicit to determine the impact of changes at one perspective on the other. This is realized by providing mappings between the classes in different meta-models and instance models at different perspectives. Implicit links already exist between classes that describe the same design facet at different design perspectives. For example *stake holder* in the BMM is related to *actor* in the CMM (they are all octagon shape in the figures). The semantic meaning is that the work that a stake holder promises to fulfill, is delegated to actors such as employees, information systems, etc. `insurance director` in the AGFIL-BM is mapped to `claim office employee` in the AGFIL-CM. Continuing the traceability within the who facet, an actor in a CM is represented as one or more endpoints in the corresponding LM. For example, `assessor` in Fig.5 offers its services via `assessment endpoint` in Fig.7.

To illustrate how vertical traceability is supported let us assume that AGFIL decides to adjust its business strategy and will now pursue the goal `accurate assessment` rather than `quick assessment`. Linked to the original goal was the rule `if estimate > $500, select assessor` in the AGFIL-BM. However, this rule has now become invalid as it does not always ensure that assessment is done by an external assessor. In turn, constraints in the AGFIL-LM following from this rule are now no longer applicable. Once we identify all the affected areas, we can decide how necessary changes should be introduced and implemented. Changes can of course also be traced in a bottom-up manner.

B. Changeability along facets

Similar to perspectives, design facets interact with one another through associations among them. Facet relationships enable *horizontal traceability*, which complements the earlier discussed vertical traceability. Horizontal traceability is supported via associations in the meta-models and instance models. To recall from the introduction of section V, the associations between classes and objects capture the interactions amongst different facets. The ’pursue’ link between `insurance director` and `24hr receipt` is an example of the interaction between the ”who” and ”why” facet in the business perspective. Similarly, the ’performs’ link between `claim office employee` and `send claim file` illustrates the interaction between the ”who” and ”how” facet in the logical perspective.

To demonstrate how horizontal traceability for changeable business collaboration design is supported, let us take the goal `24hr receipt` for example in the AGFIL-BM (see Fig3). Suppose that `AGFIL` decides not to pursue this goal in the future, but rather set up an internal claim call center. This change has severe ramifications for the current model. Most notable is the fact that `Europ Assist` will be no longer pursuing any goals in the AGFIL-BM. Since each stake holder must pursue at least one goal, this violates the requirement of consistency. The obvious solution is to remove `Europ Assist` and `helpline director` from the model. This in turn affects the resource exchanges `helpline director` has with `insurance director` and `garage owner`. Both are now dangling as one end of the exchange has disappeared. The first exchange can be removed completely including the steps associated with it. The exchange with `garage owner` needs to be re-defined to now have `helpline director` as the second party involved. This example clearly illustrates how horizontal traceability enables designers to determine the parts of a model that are affected by a change, and decide the possible solutions.

C. Changeability along aspects

Dependencies exist among design aspects as they encompass different parts at the same design perspective for a business collaboration. As observed in subsection III-C all aspects are expressed using the same modeling language for a particular design perspective. Looking for example at the business perspective, **Collaboration-BMs** can be defined from the generic BMM by excluding all internal elements (like internal goals) constituting all description elements on or outside the boundary of participants in Fig. 3 (like `24hr receipt` and `damaged car`). The individual views of participants on the collaboration constitute the elements positioned at their border, such as `supply claim info` and `repair evaluation information` of `insurance director`, and are captured in so-called **ParticipantBehavior-BMs** prohibiting use of internal description elements or more than one external stakeholder and enterprize element. All description elements contained within a participant’s borders together form the model of its local aspect, i.e. **Local-BMs** expressed through internal elements only , e.g. `inspect car` and `assess` for `assessor manager`.

To illustrate how aspect changeability is supported let us assume that AGFIL decides to adjust its business strategy and will now pursue the goal `accurate assessment` rather than `quick assessment`, thus changing the AGFIL Collaboration-BM. As a consequence the expected behavior from `assessor manager` changes, requiring it to assess how the ParticipantBehavior-BM describing its potential behavior needs to be modified. This can lead to the installment of an extra quality check of step `assess`, as such resulting in changes to its Local-BM. Of course changes to internal behavior can also be traced to potential behavior of participants, and consequently to the overall public behavior.

VII. Related Work

Most of the work in service composition and business collaboration has been focusing on the development of compositions without taking into too much considerations in modeling and change management. BPEL [6], the de facto standard in this area, provides ”a model and a grammar for describing the behavior of a business process based on interactions between the process and its partners”. BPEL provides no basis though for high level modeling and change management. Representative work from the scientific community on service composition specification is provided in [12] that provides an ”appropriate conceptual model for developing and describing web services and their composition”. The framework includes the notion of goal specification, but no support is offered for capturing business or operational specifics, or for maintaining consistency among the different roles in business collaboration design.

Some specific work has been done in the service composition arena concerning business-IT alignment. [8] presents a web service management architecture (WSMA) in which business level metrics are mapped to service executions to allow assessment from a business perspective. However, the metrics are fairly low-level, lacking support for capturing high level business requirements. [23] describes a rule inference framework called DYflow, where ”end users declaratively define their business objectives and the system dynamically composes web services”. But there is no clear-cut separation between technical and business oriented rules, compromising the framework’s capability to facilitate change management. In the area of changeability along aspects relevant work can be found in [20] present a development process to incrementally align so-called global and local requirements within business collaboration through negotiation. However, also here technical and business requirements are intermixed diminishing. [11] identifies similar design aspects as the ones in the BCDF, however, their application is limited to a service oriented computing context.

Prominent work in the area of business process modeling is the ebXML framework. Its specification on business process modeling, ebBPSS [13] defines a comprehensive meta-model for describing business collaborations. The implementation of instances of this meta-model is facilitated through other components of the ebXML suite, e.g. ebXML’s messaging service. Although ebBPSS offers a very complete set of concepts to define business collaborations, it too does not provide a change assessment mechanism with which the affect of business and/or technical changes can be easily determined. Moreover, although ebBPSS in conjunction with the components ebCPP and ebCPA offers a conceptual approach for merging business profiles into agreements, this mechanism is not grounded on an explicit traceability mechanism with which the affect of a change in one design aspect on another can be easily determined.

The system development community has focused much of its efforts in developing comprehensive methodologies for developing enterprize information systems. An exemplary proponent of this effort is Tropos [3], which is an agent-oriented software engineering methodology focusing particularly ”on activities that help to gain understanding of how and why the intended system would meet the set out goals”. The main difference with our work lies in the fact that in [3] the notion of traceability is not made explicit in relation to changeability as in our business collaboration design framework, and its potential is thus not harnessed.

The contributions of our work in comparison to the above mentioned work on adaptive business collaboration design can be summarized as followed:

- The BCDF provides a complete approach to business collaboration design by specifying the modeling languages required for business collaboration design ranging from high-level requirements to actual implementation in a consistent and comprehensive manner.
- Due to its support for tracing the BCDF enables the assessment of the impact of changes in the existing collaboration design, and solutions can be provided accordingly.
- The support for traceability enables detection of conflicts arising due to the incorporation of a change in a design perspective or facet. Especially when combined support for the three forms of changeability provides a powerful set of tools for ensuring consistency throughout business collaboration designs.
- Its model-driven approach makes possible the rapid development and delivery of business collaborations based on proven and tested models, which is a critical benefit in a highly dynamic business environment.

VIII. Conclusions

Current standards in business collaboration design, such as BPEL and ebXML, are not suitable for dealing with the complex and dynamic nature of developing and managing business collaborations in accordance with specified business requirements. Without the support for high level modeling facilities and traceability such standards can not facilitate the verification and changeability required in business processes crossing organizational boundaries. The challenge is thus to provide a solution in which business collaboration design can be done in an effective and traceable manner.

In this paper we have presented the Business Collaboration Design Framework (BCDF), a c framework that utilizes a multi-perspective and multi-facet approach to collaboration design. Business collaborations are modeled in three different perspectives, intersected by six facets. The work presented herein describes the relevance of the BCDF for modeling at high level and enabling changeability in business collaboration design in a conceptual manner.

Work for future research will be focused on the formalization of the BCDF to allow formal assessment of the introduced notion of verifiability and changeability. Another issue that needs to be addressed is the development of a change management sub-system to control the evolution of business collaboration designs. A prototype for presented approach is currently under development; where an early, partial implementation has been reported in [16].

REFERENCES

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

[2] J. Bowers, G. Button, and W. Sharrock, Workflow from within and without, *Proceedings of the 4th European Conference on CSCW, 1995*

[3] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, J. Mylopoulos, Tropos: An Agent-Oriented Software Development Methodology, *Autonomous Agents and Multi-Agent Sytems, Vol. 8, No.3, pp. 203236, 2004*

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

[5] Business Rules Group, Defining business rules - What are they really?, *http://www.brcommunity.com, July 2000*

[6] F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, S. Weerawarana, Business Process Execution Language for Web Services, *http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/, July 31, 2002*

[7] Cambridge Learner's Dictionary, *http://dictionary.cambridge.org*

[8] Business-Oriented Management of Web Services, F. Casati, E. Shan, U. Dayal, M. Shan, *Communications of the ACM, Vol. 46, No. 10, pp. 55-60, 2003*

[9] B. Curtis, M. Kellner, J. Over, Process Modeling, *Communications of the ACM, Vol. 35, No. 9, pp. 75-90, 1992*

[10] W. Deiters, T. Goesmann, T. Lffeler, Flexibility in Workflow Management: Dimensions and Solutions, *International Journal of Computer Systems Science and Engineering, Vol. 15, No. 5, pp. 303-313, 2000*

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

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

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

[14] P. Grefen, K. Aberer, Y. Hoffner, H. Ludwig, CrossFlow: Cross-Organizational Workflow Management in Dynamic Virtual Enterprises, *International Journal of Computer Systems Science & Engineering, Vol. 15, No. 5, pp. 277-290, 2000*

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

[16] B. Orriens et all, ServiceCom: A Tool for Service Composition Reuse and Specialization, *Proceedings of the 4th International Conference on Web Information Systems Engineering, Rome, Italy, 2003*

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

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

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

[20] P. Traverso, M. Pistore, M. Roveri, A. Marconi, R. Kazhamiakin, P. Lucchese, P. Busetta, P. Bertoli, Supporting the Negotiation between Global and Local Business Requirements in Service Oriented Development, *Proceedings of the 2th Int. Conference on Service Oriented Computing, New York, USA, 2004*

[21] F. Vernadat, CIMOSA - A European Development for Enterprise Integration Part 2, Enterprise Modeling, *Pergamon Press Inc. Tarrytown, NY, USA, 1992*

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

[23] L. Zeng, B. Benatallah, H. Lei, A. Ngu, D. Flaxer, H. Chang, 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*