Specification of Payment Requirements for Business Collaborations

Bart Orriens

January 18, 2006

Abstract

Service-oriented computing (SOC) is the computing paradigm that utilizes services as fundamental elements for developing business collaborations. In order to realize this vision payment is a critical issue that must be addressed. Current work in this area is scarce, and usually focused on low level requirements. In this report we present a layered specification of payment properties for business collaboration. The approach supports specification of high level payment criteria, the mechanisms to achieve those, and the measures needed to implement these mechanisms. Moreover, dependencies among criteria, mechanisms and measures are made explicit as such creating a traceable path from criteria to measures.

Infolab Technical Report Series, no. 27

January 2006

1 Introduction

Recently there has been increasing focus on service-oriented computing (SOC), the new emerging paradigm for distributed computing and e-business processing, to deliver flexible and adaptable corporate business services by utilizing existing services across organizational boundaries. *Business collaboration* refers to a cooperation between multiple enterprizes working together to achieve a common business goal. In order to realize the vision of utilizing services as fundamental elements for developing applications [12] for business collaboration, payment is a critical issue that must be addressed. Businesses will be averse to participating in cooperations that can only involve free services; moreover, payment must be facilitated such that problems with regard to for example refutability, refundability and annulability do not occur.

Therefore, for the successful adoption of SOC within the business collaboration domain, the paradigm must provide the means to make cooperation between enterprizes payable. At the moment, the most successful manifestation of SOC can be found in web services technology. A web service is a specific kind of service that can be unambiguously identified (generally by means of a URI) and whose service description and transport utilize open Internet standards, such as XML-based SOAP messages. Unfortunately, most work in the web service payment arena concentrates solely on low-level payment provisions by utilizing existing payment standards; leaving the issue of relating these to higher level business requirements unaddressed.

In this report we briefly introduce our business collaboration context framework, which provides the context required for business collaboration development and management. We also show how this context can be described using collaboration models. Subsequently, we explain how the introduced collaboration models can be augmented to support the specification of payment requirements from high level criteria to low-level payment measures such as provided by current web service payment solutions.

The remainder of this 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 discuss our framework for business collaboration context; after which we explain our model driven approach for the definition of this context in section 4. After that, in section 5 we analyze the role of security in business collaboration, and show how the specification of security requirements can be facilitated. Finally, we present conclusions in 6 and outline future work.

2 Example

To exemplify the ideas presented throughout this paper an example inspired by the case study in [6] 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

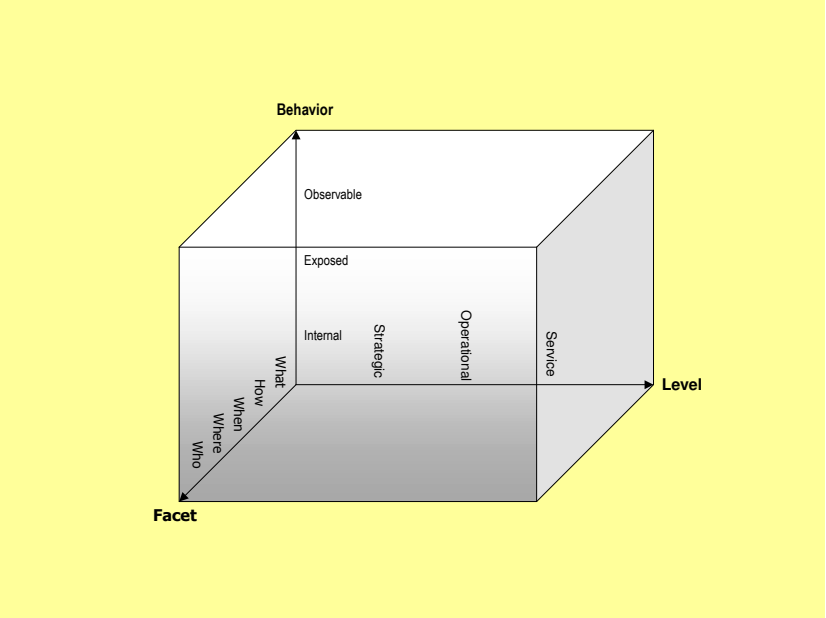

Fig. 1: Business Collaboration Context Framework (BCCF)

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 [9, 10].

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 [4, 13, 16]): 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 *conversation 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 [8, 17]): 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 [3, 14, 17]): *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 the BCCF

To capture the three dimensions of collaborations aspects, levels and facets of BCCF we employ two types of model: meta models and models, both of which are defined for individual levels. Meta models provide design guidelines in terms of classes and their relationships, where depending on the collaboration aspect being modeled additional constraints are placed on the meta-model. Models represent a particular application design, and are derived by populating a meta model's *classes*.

Every meta model consists of six classes, where each class captures a particular facet; i.e. for *what*, *how*, *where*, *who*, *when* and *why* facet. Every class constitutes a set of logically related *attributes*. *Associations* connect the classes expressing dependencies among facets. *Mappings* define dependencies among levels by providing links between classes that describe the same facet at different perspectives (illustrated by the arrows between facets at different perspectives in Fig. 1).

Snippets of exemplary models for the AGFIL application are illustrated in Fig. 2, showing its strategic, operational and service model respectively; where the models are represented based on UML conventions. In order to distinguish different facets, we represent them in different shapes in their UML models (see also legend in Fig. 2): *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. For more information the reader is referred to [10, 9]; where [11] contains the most recent details.

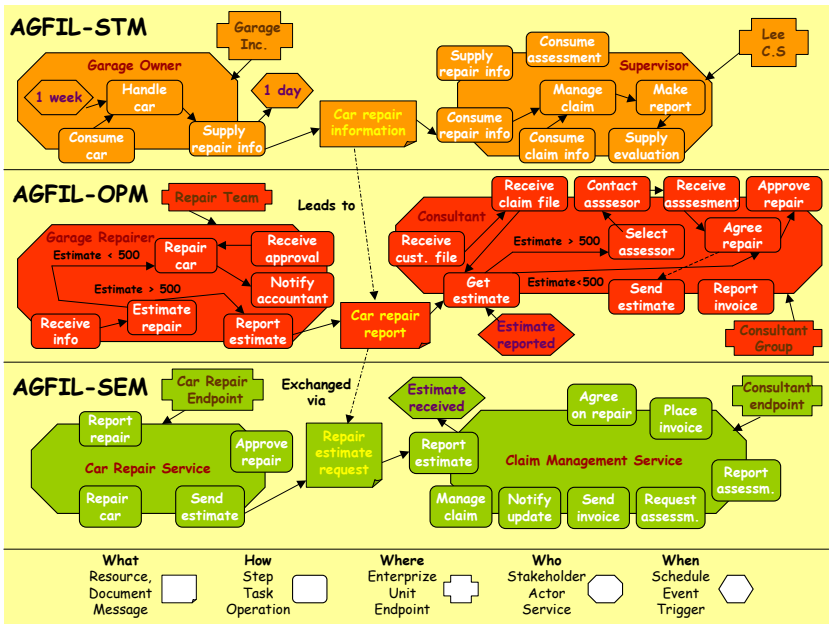

Fig. 2: AGFIL Collaboration Models

4.1 Strategic Models

At strategic level, strategic models like the AGFIL-STM in Fig. 2 capture purpose and high level requirements of business collaborations, akin to requirements analysis [1, 16]. Strategic models are expressed in terms of resources, steps, stake holders, enterprizes, and schedules. Resources such as `car repair information` provide abstractions for means such as financial, human and informational capital. Resources are used and produced by steps which represents high level functions.

Steps are of type 'internal' (like `handle car` presented inside the stakeholder boundary in Fig.2); or type 'communication' representing resource supply and consumption e.g. `consume repair information`. Stake holders like `garage owner` describe the participants involved who are responsible for carrying out defined steps. Stake holders belong to an enterprize, where enterprizes are manifestations 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`.

4.2 Operational Models

At operational level, operational models like the AGFIL-OPM in Fig. 2 depict how high level strategic behavior is realized in terms of operational activities. These are expressed in terms of documents, tasks, actors, units, and events. Documents (like `car repair report` represent the flow of information in a collaboration behavior. Documents are used and produced by tasks. Tasks represent specific business functions, and are of type 'internal' or 'communication' (represented inside or on the boundary of the actors respectively), e.g. `collect claim form` and `report invoice` respectively.

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, events are published and subscribed to by actors. Events describe business occurrences which have properties such as 'date', 'time', 'severity'.

4.3 Service Models

At service level, operational models are translated into service models that specify how the described operational behavior is realized using the services offered by the IT-infrastructure. Service models are defined in terms of messages, operations, services, endpoints, and triggers. Messages represents containers of information (e.g., `repair estimate request`), consisting of meta-data and actual data. Messages function as the inputs and outputs of operations such as `place invoice`.

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`, which constitute 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.

4.4 Mappings between Models

For the specification of dependencies between different collaboration behaviors at different levels, we employ vertical mappings. Vertical map-

pings 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. We define the following mappings:

Resources at strategic level are mapped to documents at operational level. Documents themselves are mapped to messages using *exchangedVia* relations. Steps are mapped via *decomposedIn* relations to tasks; whereas tasks are *realizedBy* operations. Stake holders *control* actors, where each actor is *representedBy* a service. Enterprizes are *organizedIn* units, where each unit itself *offers* one or more endpoints. Schedules are *splitInto* events, where each event *causes* multiple triggers.

5 Payment in Business Collaboration

Payment in general constitutes ”to give money to someone for something you want to buy or for services provided” [2]. Interpreted in the context of business collaboration payment deals with providing assurance to enterprizes that they get paid for their role in the cooperation. In other words, payment is concerned with providing peace of mind for the businesses involved, where they can rely on the fact that their collaboration efforts result in some sort of reward, and is safe from risks like repudiation of payment, too limited respendability, and etceteras; caused by the ambiguous specification of payment conditions.

When put into the business collaboration context as presented in section 3, it follows that for business collaboration payment can be perceived at a strategic, operational and service levels, each of which represents a level of abstraction with its own content and meaning regarding payment. In the remainder of this section we shall discuss the role of payment and the specification of payment requirements at the different levels. An overview of the requirements that can be specified is provided in Figure 3.

5.1 Strategic Level Payment

As observed at an abstract strategic level a business collaboration constitutes a cooperation between enterprizes making use of each other's business services to exchange resources to further their business goals. At this level payment specification deals with the definition of the criteria for the payment(s) involved with these resource exchanges; where these are associated with the steps in those exchanges. Criteria analysis here is aimed at

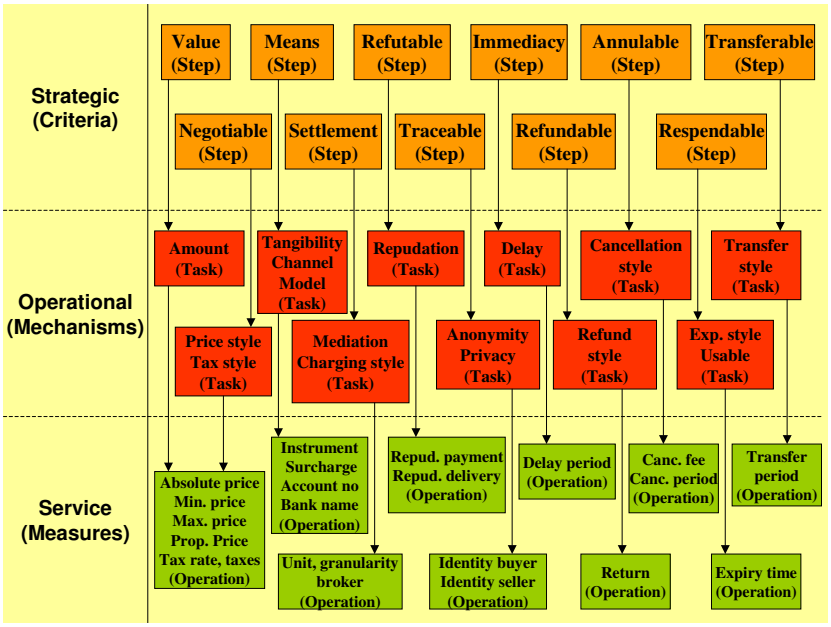

Fig. 3: Payment in Business Collaboration

1) identification of the criteria, 2) measuring the magnitude of the potential loss if this criteria is not met, and 3) the probability that the loss will occur. For business collaboration we identify the following types of payment criteria; where this overview is based on an analysis of current literature such as the works in [5, 7, 15] on the characteristics of payment protocols:

- *Value*

  A first criteria is the *value* of the business services being provided by a stake holder; that is, what is actually to be paid. at strategic level this is vague in nature indicating merely for each individual step whether payment is 'low', 'normal' or 'high'; where this is of course relative to the particular collaboration.

- *Negotiable*

  Related to the value of the performance of a step is whether this value is open for negotiation or not, i.e. whether it is *negotiable*. Standard services may have default prices, but in customized situations price might not necessarily be fixed.

- *Means*

  The things with which a stake holder wishes to be paid for its activities is expressed in the *means* characteristic; where we identify three generic types of means that can be used for payment, being financial

(like money), informational and natural (like land, products) means.

A side issue of importance with the selection of a particular payment means is whether it should be easily *exchangeable* or not? Obviously financial means can be exchanged for other means rather straightforwardly using well established mechanisms. In contrast, natural means like land are not so easily transferred into for example money.

- *Settlement*

  The *settlement* criteria deals with the manner in which payment is to occur. Is payment done on a 'rental', 'subscription', 'metered', 'facilitated', 'escrow' or 'swap' basis?

- *Annulable*

  Another aspect of payment is what occurs when the buyer or seller tries to undo the transaction. Whether payment is *annulable* at all is the first criteria in this regard. If payment is annulable, then even after a stake holder has paid for a business service he/she can still undo the transaction.

- *Refundable*

  Closely related to annulability is whether payment is refundable. That is, upon undoing the transaction can the stake holder get some sort of refund?

- *Refutable*

  Also related to the previous two criteria is the question of whether payment can be *refutable*. That is, is it possible for a stake holder to deny payment has occurred when in fact this is not the case? Alternatively, can the buying stake holder deny that the promised step has not been performed yet to avoid payment?

- *Immediacy*

  An additional interesting question for a payment transaction relates to how quickly the paid amount is received? *Immediacy* is one dimension of this issue, expressing how long it takes before payment is received after the corresponding step has been performed. For example, a bank transfer will take several days to complete, whereas cash knows no such delay. Possible values here are 'immediate' and 'delayed'.

- *Respendable*

An issue related to immediacy is whether a payment is *respendable* after receipt. After payment has been done, does the seller have to take intermediate steps before it can be used in other transactions? Is this possible at all?

- *Transferable*

  Is the payment specific to the stake holder receiving it or can it be used by others as well? That is, is the payment *transferable* to other parties? Named bonds for example are not transferable, but cash obviously is.

- *Traceable*

  A last criteria concerns the traceability of the payment? Is it necessary for example that the identity of both buyer and seller are known? Or the exact amount paid, at what time, and etceteras? This is captured in the *traceable* requirement.

In addition to the above criteria various other payment criteria are sometimes mentioned in literature, such as low transaction costs, scalability, and so on. However, these mostly relate to the characteristics of payment protocols; whereas in this report we are interested in the criteria of payment that precede selection of protocols; and on which this selection is based.

5.2 Operational Level Payment

Based on the criteria analysis at the strategic level, payment mechanisms are associated with tasks at the operational level to establish support for the identified criteria. At this level a business collaboration constitutes the sending and receiving of appropriate documents by enterprizes to further the state of the business collaboration. As such, the payment mechanisms that will be employed, are to augment this document communication with payment-related specifics. In correspondence with the described criteria in subsection 4.1 we identify the following payment mechanisms:

- *Value, negotiable* $\rightarrow$ *Amount, price style, tax style*

  To operationalize the value of a step at strategic level, essentially the *amount* to be paid for each of its tasks must be specified. Here the qualitative value indicator is concretized in quantitative amounts. Naturally, given the indication provided at strategic level for a step, the total sum of amounts of its tasks at operational level must be in

fair relation to this indication. Note by the way that not all tasks of a step necessarily need to cost money.

Additionally, the manner in which the amount is specified, is captured in the *price style*. If the height of payment is negotiable for a task, then the manner in which the price must be defined differs from when it is not negotiable. Price style captures the mechanism used, where the options are 'absolute', 'proportional' and 'ranged' price. In the first two cases negotiation is not possible. If the price style is 'ranged' then there is room to negotiate.

Lastly, since usually tax must be paid whenever business services are paid for, the *tax style* must be depicted. The style can be equal to 'inclusive' or 'exclusive'. This information is naturally important for buyers to determine the real cost of a task.

- *Means → Tangibility, channel*

  The type of means selected at strategic level falls into two dimensions at operational level: *tangibility* and *channel*. Tangibility reflects whether the payment instrument that will be used at service level must be tangible (i.e. can be touched) or intangible. Natural means will always require a tangible instrument; whereas informational ones use intangible instrument. Financial means can be both tangible or intangible. Related to the previous is the question of what type of payment channel is used: an 'online' or 'offline' one. Tangible payments can only be made offline naturally, intangible ones via an online channel.

- *Settlement → Mediation, model, charging style*

  The type of settlement chosen for a payment is concretized in the form of *mediation*, *model* and *charging style*; where these are associated with individual tasks. Mediation may be 'direct', i.e. no mediation, or 'indirect'; in which a third party is involved in establishment of the payment. Possible payment models include 'account based' and 'token based' payment; whereas charging can be done on a 'per request', per unit', or a 'percentage' basis.

- *Annulable → Cancellation style*

  If payment for a step is annulable, then its related tasks at operational level must depict what *cancellation style* is supported. Annulment may be done for 'free', or be accompanied by a 'penalty'.

- *Refundable → Refund style*

  When a step's payment is refundable, then for each task to which it is mapped the *refund style* must be depicted. Possible styles that currently have been identified are 'return with money', 'return for voucher' and 'trade'.

- *Refutable → Repudiation*

  The refutability of a step, when set to 'false', requires that for each task the type of *repudiation* (that is to be prevented) must be depicted.

- *Immediacy → Delay*

  In case payment for a step is not immediate, i.e. there is a delay, then it must be indicated for each task how 'short' or 'long' the *delay* will be. Naturally if a stake holder immediately has control over the payment, then at operational level there must be no delay.

- *Respendable → Expiration style, Usable*

  If the means with which a payment is made, are respendable then two issues must be addressed at operational level: whether the means will expiry or not; and if the means are immediately usable or not. The former is captured in the *expiration style* of a task; where possible values are 'indefinite' and 'limited'. The latter is represented in the *usable* property; where usage may be 'immediate' or 'postponed'. Observe that it is theoretically possible that if the means will expiry and they can not be immediately used, an actor could receive a payment but by the time he/she can spend it the means used would no longer be valid.

- *Transferable → Transfer style*

  In many cases payment of a step will be transferable from the seller onto others. If so, at operational level the manners in which this can be done must be specified for each related task. This is depicted in the *transfer style*; where transfer can be 'direct' or 'indirect', e.g. money can be given from one person to another, whereas a bank transfer involves a third party.

- *Traceable → Anonymity, privacy*

  When a payment should not be traceable, it must be defined what this means in operational terms. For this purpose the properties of *anonymity* and *privacy* are used. Anonymity pertains to the issue of whether the actors involved are to be anonymous or not. Privacy is concerned keeping the details of the payment a secret, e.g. regarding the height of the payment, time of payment, and so on.

Together these mechanisms provide the building blocks required to meet the criteria identified at the strategic level on the operational level. In the following subsection we will explain how the mechanisms can be realized at service level.

5.3 Service Level Payment

The payment options selected at the operational level must subsequently be realized at the service level via payment measures. This level is the domain of the service oriented computing paradigm. In this paradigm a business collaboration is viewed as a set of interacting technical services, where these interactions are message based and facilitate the communication of information among services. In order to meet the business driven security demands at this level, the message based interactions, i.e. message exchanges, must have concrete payment details.

For this purpose we have augmented the definition of operations with the following payment measures; where these are discussed grouped in accordance with the payment mechanism they realize:

- *Amount, price style → Absolute price, proportional price, minimum price, maximum price*

  The height of the amount to be paid and the style in which this price is depicted, are reflected in the *absolute*, *proportional*, *minimum* and *maximum price*. If the pricing style is 'ranged', then a minimum and maximum price are specified to define the offered price range. Otherwise, an absolute or proportional price suffices.

  When it comes to the price of an operation with regard to the total cost of a task, for a proper alignment the sum of operation prices must be equal to this total cost; that is, the total cost of all operations realizing a task must equal the amount to be paid for this task.

- *Tax style → Tax rate, taxes*

  Depending on whether tax is specified exclusive or inclusive with regard to the price, at service level the *tax rate* or *taxes* amount is defined for each operation. In case tax is exclusive, then the prices plus taxes of all operations summed must equal the total amount of the corresponding task at operational level.

- *Tangibility, channel, model → Instrument, surcharge, account number*

At service level tangibility, type of channel, and payment model determine the payment instrument used per operation. Supported instruments are 'cash', 'cheque', 'directFundTransfer', 'creditcard', 'travellersCheque', 'wireTransfer', 'moneyOrder', 'bankBill', 'voucher', 'storedValueCard', 'digitalCash' and 'anonymousCash'; where depending on whether the instrument must be tangible or intangible, exchanged online or offline, and account or token based.

In addition, in case a credit card is used, then additionally the *surcharge* must be specified. If a money order or bank bill is accepted, then bank details like *account number bank name* of the service provider must be provided. If the payment model used for a task is 'account based', then for each operation the requester must provide its *account number*, *name*, and etceteras.

- *Mediation → Broker*

  If there is a third party mediating in the payment transaction, then at service level each operation must depict what *broker* is to be used.

- *Charging style → Unit, granularity*

  To recall from subsection 5.2, several different charging styles can be employed at operational level, being 'per request', 'per unit' and 'percentage based'. In case 'per unit' payment is opted for, then each operation at service level must define its *unit* and *granularity*. Possible values for unit are 'time', 'weight', 'volume', 'area', 'length', 'watt', 'byte', 'person', 'event', and 'permit'. Depending on the chosen unit granularity is then one of the values 'hour', 'minute', 'second', 'day', 'month', 'year', 'night', 'week', 'fortnight', 'gram', 'kilogram', 'tonne', 'cubicMetre', 'squareMetres', 'millimetre', 'centimetre', 'metre', 'kilometre', 'kilowatt', 'megawatt', 'kilobyte', 'megabyte', 'gigabyte', 'adult', child', 'infant', 'pensioner', 'senior', 'mouseClick' and 'ticket'.

- *Cancellation style → Cancellation fee*

  If cancellation can be done for free, then no additional information is required at service level. If, however, cancellation comes with a penalty, then for each task a *cancellation fee* must be defined.

- *Refund style → Refund amount*

  Similar as for cancellation, if refund is offered in the form of money back, then the *refund amount* must be specified. This can be either absolute, or proportional to the paid price. Naturally the total refund can never exceed the total price.

- *Repudiation → Delivery repudiation, payment repudiation*

  When the payment transaction should not be refutable, then it must be indicated for each operation of a task whether *delivery repudiation*, *payment repudiation*, or both, must be prevented.

- *Delay → Delay period*

  Delay at operational level of a task must be quantified at service level for each operation by depicting the exact *delay period*. This of course must be in relation to the qualitative height of the task.

- *Expiration style → Expiry period*

  If the means with which the payment is conducted can expire, the *expiry period* of each corresponding payment instrument must be depicted at service level. Note that some instruments used may be non-expirable, however, at least one of them should have an expiry period.

- *Transfer style → Transfer period*

  The transfer style available for a task translates itself in the *transfer period* of the payment for each operation. If transfer is 'direct', then this period will be zero. Otherwise, some sort of transfer period will need to be specified.

- *Anonymity → Anonymous buyer, seller*

  The anonymity required for a task at operational level is refined at service level for its operations in whether the identity of the buyer and/or seller must be protected; defined in *anonymous buyer* and *anonymous seller*.

- *Privacy → Private price, time, place*

  Lastly, the privacy guarantees defined for a task, are refined to depict whether the various details about the payment transaction should be kept private. These are captured in *private price*, *private time*, and *private place*, which are all boolean indicators.

Observe that the above described measures at service level are not intended to be exhaustive in nature. The authors are aware that many other measures exist; the above is therefore intended to be of illustrative nature to show how operational payment mechanisms may be refined at a technical service level.

6 Conclusions

In this technical report we addressed the issue of specification of security requirements for business collaborations. This work is motivated by the lack of support thereof in the current research with regard to relating high level payment objectives to concrete payment measures; as most work (like [5, 7, 15]) focuses on establishing comparison criteria for payment protocols without; or on provision of low level payment measures without taking higher level, business driven security requirements into consideration.

To remedy this situation we introduced our generic framework for capturing the business collaboration context; and explained how this context can be described via the usage of various meta models and models. After that we explained how these meta models and models can be augmented to facilitate payment requirement specification at strategic, operational and service level. Furthermore, we established relations between the requirements at these different levels; as such enabling traceability of strategic payment criteria to operational payment mechanisms to service level payment measures (and vice versa).

A caveat concerns the defined payment properties: these are not intended to be exhaustive in nature nor do the authors expect them (and the relations between them) to be final. As the authors are not themselves experts in the field of payment, more work to further develop the (currently basic) support for payment requirement specification is required. However, we believe that the presented approach provides a first step on the road to comprehensive payment requirement specification for business collaboration.

References

[1] 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*

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

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

[4] 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*

[5] M. Fischer, H. Gall, M. Hauswirth, Towards a Generalized Payment Model for Internet Services, *Technical Report TUV-1841-2002-53, Distributed Systems Group, Technical University of Vienna, Austria, 2002*

[6] 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*

[7] J. Mackie-Mason, K. White, Evaluating and Selecting Digital Payment Mechanisms, *Interconnection and the Internet, pp. 113-134, 1997*

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

[9] 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*

[10] 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*

[11] B. Orriens, Modeling The Business Collaboration Context, *INFOLAB Technical Report Series, No. 28, Tilburg, The Netherlands, January 2006*

[12] M. Papazoglou, G. Georgakapoulos, Introduction to the Special Issue about Service-Oriented Computing, *Communications of the ACM, Vol. 46, No. 10, pp. 24-29*

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

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

[15] J. O'Sullivan, D. Edmond, A. ter Hofstede, What's in a Service? Towards Accurate Description of Non-Functional Service Properties, *Distributed and Parallel Databases, no. 12, pp. 117-133, 2002*

[16] 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*

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