Client-oriented Architecture for e-Business

Jose Bernardo Neto and Celso Massaki Hirata
Technological Institute of Aeronautics, Department of Computer Science, Sao Jose dos Campos, SP, Brazil

Keywords: e-Business, Architecture, Web Services, Client, e-Contract.

Abstract: In general e-Business solutions focuses in the service provider, they do not consider companies or large client interest. The current web service business models do not allow consumers either to request for unavailable services nor allow to request their services needs met by a complex arrangement of providers. In this paper we propose a novel approach for making service compositions based on customer demand to meet business needs in an agile way. The business needs are related to the flexibility of the client to express its needs. The agility is related to the capability of the supplier to implement services in timely manner. To achieve the goal, we use as support the business forms and the e-Contract lifecycle methodology to discipline and facilitate the provider and client responsibilities.

1 INTRODUCTION

Most brick-and-mortar stores already have their e-commerce versions, which in some cases offer better prices coupled with incentives such as free shipping and the fact that they are open 24 hours every day. The strategy used for maintaining competitiveness is to store data in the Cloud and use Web Services.Web Services are the agents responsible for exchanging data between the Cloud and the web clients. They support the connection among different types of IT systems, ensuring that the diverse requirements from the clients are properly met. Also, new services can be created by combining existing Web Service components.

(Mitra et al., 2013) detail bureaucratic steps that the customers need to follow in order to obtain the best orders in the websites. The acquisition occurs via intuitive features and options offered by web locals.

Regarding large clients, there are few channels for this type of trade, and the typical searches do not guarantee that quality of service will be provided. In general, work on service composition deals with services managed by the same server or a SOAP solution.

Some problems for large clients, when they use conventional e-commerce portals, are the following:

(i) Service provider cannot meet the high demand;

(ii) Their requirements for services are too specific or unusual to be met; and

(iii) There are restrictions on quality, time, and quantity for the existing services.

In general, large clients use standard business forms as a tool to obtain information or to invite providers to describe their services. The business forms allow the participants to submit their requests in a standard format. They can be designed, saved, printed, shared, and signed online. They also can be used to start an agreement in the proposal phase through different sites. In many cases, the client fills out the forms manually, prints them and sends them by mail.

Current studies show the need for e-business to focus more on customer interest. There is a growing number of publications focusing on the analysis of the data exchanged by people who participate in social networks. (Gong et al., 2013) analyze data exchanged to check the trend of consumption to direct the availability of services.

Thus, there is a need better identify client interest early to automate standard business forms such as proposal documents and to make use of the information exchanged to develop e-agreements. In this paper we propose an innovative architectural model based on the e-Contract lifecycle, respecting the client's demand and the web services characteristics. Our proposal seeks to increase the dynamics, interactivity, and effectiveness in the e-Contract lifecycle using the standard business forms. The e-Contract makes a bond between the parties and its lifecycle supports the automation of the configuration of web services engines during the validation of the accord.

The rest of the paper is organized as follows. The-

Neto, J. and Hirata, C.
Client-oriented Architecture for e-Business.
DOI: 10.5220/0006285303910399
In *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017) - Volume 3*, pages 391-399
ISBN: 978-989-758-249-3

oretical background is provided in Section 2. Then, in Section 3 the formalization of the proposal, the architectural model structure and the schema to link the two different lifecycles. Section 4 focuses on case study. Section 5 presents the conclusion in Section 6.

2 BACKGROUND

This section presents the basic concepts related to this paper, including: e-Contract lifecyle, e-business and TS2PC4RS.

2.1 e-Contract Lifecycle

A contract must clearly define the requirements and terms that will be negotiated between the parties involved. The definition under analysis aims: (1) identify the relationship between businesses and entities, (2) the events and actions of the various parties in the negotiation process and (3) exceptions and penalties in the event of violations (Chiu et al., 2003). The goal of e-Contract lifecycle enables the automatic configuration of the web services engines in an e-commerce solution for distributed long-duration transactions (Bernardo Neto and Hirata, 2015; Neto and Hirata, 2013).

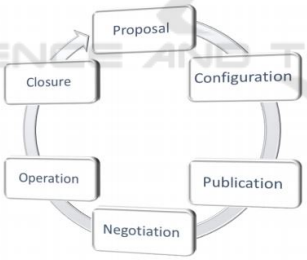

Figure 1: The six phases of the e-Contract Lifecycle.

The lifecycle is structured into six sequential phases from proposal to closure as shown in Figure 1. The e-Contract, final product of the lifecycle process, goes through a series of activities that are organized in phases. The activities have inputs and at least one output. The phases are (1) Proposal, (2) Configuration, (3) Publication, (4) Negotiation, (5) Operation, and (6) Closure. For each phase, there is at least one product to finish the phase before starting the next phase. The order must be kept due to the dependencies between the activities of the phases in terms of products. The activities were defined based on the standard in transaction rules for communication between Web Services: publishing of services, search and message exchange. There is a product, output, that identifies the end of phase (Bernardo Neto and Hirata, 2015). The e-Contract lifecycle employs the template as the origin of the e-Contract with a static structure. The template structure is a repository of data during the lifecycle of e-Contract. The template characterization is changed when a resource is added on its structure.

2.2 B2B and B2C e-Contracts

In e-business relations, electronic contracts are used to increase the level of trust among the participants. Businesses have demanded more agility from service providers, which creates a competition among them that benefits customers. Not surprisingly, the number of service contracts has grown significantly (Angelov and Grefen, 2004), and companies usually have to switch between different partners to leverage the best trading opportunities at any given moment.

Business-to-business (B2B) negotiations are usually associated with chain-contracts and services, as well as transactions involving sales of large quantities. In Business-to-consumer (B2C) cases, the focus is on negotiation with the customer (Andrew Goodchild and Milosevic, 2004).

2.3 A Timestamp-based Two Phase Commit Protocol for RESTful

Considering TS2PC4RS[1], the client is allowed to read, prewrite, write and update operations; and it initially records to the List of buffered PreWrites using timestamp order (LPW). Clients can update their prewrites in the course of transactions; and it is not required to start a new negotiation in order to implement the desired updates. If one client aborts the transaction or decreases his contracted amount, he loses his priority and other clients will be allowed to have their prewrites accepted. The prewrite update sends messages to all involved in the transaction based on the existing requests. It is necessary to submit a write message (in case of committing) to complete the transaction. Some advantages of the approach are: better support for long time transactions, relaxing of the isolation and atomicity properties, and bringing support to concurrency control (da Silva Maciel and Hirata, 2010).

[1]Timestamp-based Two Phase Commit Protocol for RESTful Services

3 COAeB

Considering the ease coordination of decoupled components that interact by service requests over the network, it is decided to choose the broker architecture pattern to design our solution. The proposed architectural model is based on three entities: Client (C), Broker (B) and Provider (P). The client expresses its needs; the broker adds value to the services; and the provider offers the services.

3.1 Agents Interactions

The interactions are based on standard operations to accomplish the interoperability by web service architecture: publish, find and bind. An operation is a set of request and response messages exchanged between two parties to complete an activity. Besides decomposing the find operation into rfp (request for proposal) and search operations; COAeB changes the original web service operations order, as shown in Figure 2.

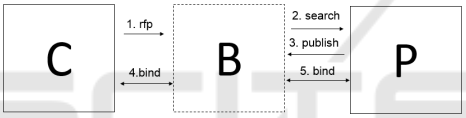

Figure 2: Entities perform the operations: rfp, search, publish, and bind.

To isolate and define the broker responsibilities, we divide the interactions into two groups: (1) client and broker, and (2) broker and providers. Figure 3 presents the sequence of messages exchanged in group 1:

1. The client sends the request for proposal (rfp) message to Broker to trigger the interactions.
2. The broker stores the rfp. The Broker is responsible to finding the resources.
3. The client confirms its interest in the requested resources or tries to change (negotiate) quality, cost, or delivery date of the resources.
4. Therefore, the broker can decide not to agree on the service if the time expires. The broker sends an ok message when it is still able to meet the rfp (or the negotiated rfp).
5. The client sends the complete message to order and use the service.

Each rfp can require one or more services. Services are offered by providers and published on the Broker. If the Broker finds the service required, it

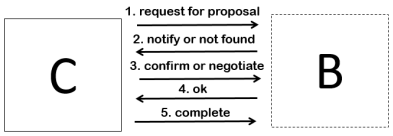

Figure 3: Sequence of messages exchanged in group (2).

sends the confirm message as shown in step 3-5 below. Otherwise, it invites the provider to offer the service, step 1. The Figure 4 illustrates the messages exchanged in group 2.

1. Through the service request, the broker invites the provider to publish the described service.
2. The publish message means to registry a service available to offer.
3. The broker sends the confirm message to prepare to commit the agreement or try to change (via a negotiate message).
4. The broker can take some time to decide or initiate the negotiation. Therefore, the provider can decide not to agree to the service if the time expires.
5. The broker sends the complete message to close the agreement and imposes the service delivery obligation on the provider side.

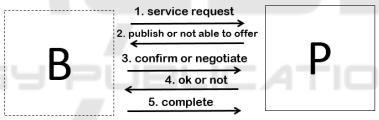

Figure 4: Illustrates the second group of interactions.

We designed a set of agents for the COAeB architecture that cooperate to perform the activities listed above. Figure 5 illustrates the COAeB architecture. The external agents are squares and internal agents are circles: Client (C), Business Broker (BB), Transaction Coordinator (TC), Composer (CP), Provider Broker (PB), and Provider (P).

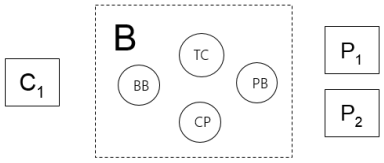

Figure 5: An overview of COAeB schema.

We assume that the client needs a complex service that is composed of several basic services. It sends the rfp. Business Broker (BB) works as Rfp Storage, Agreement Manager, and Service Negotiator. It collects the rfp information to prepare a draft. It sends

the service description to CP. It receives the result of the composition transaction via TC to build the pre-Contract with the client. The pre-Contract (with the client) is the publication of the booked services. Composer (CP) is responsible for the composition of services based on the Try- Cancel/Confirm (TCC) pattern for web service (Pardon and Pautasso, 2014). It receives a complex description of services from BB. It divides a complex service into basic services. It requests TC to book and acquire the resources. Transaction Coordinator (TC) works as a Service Coordinator. It opens the atomic transactions with Try-Cancel/Confirm pattern to build the composition. In the end, it handles commit or abort the transaction. It receives the URL of the services via CP and orchestrates the operations to complete the transaction of the composition. To improve the scalability of the COAeB, we provide two scalable agents to interface with clients and providers, BB and PB. In accordance with the e-Contract lifecycle, they are also responsible for preparing the agreements with client and provider respectively.

3.2 Agents Relationship

Agents use operations to perform the activities. Operations follow request-response pattern in COAeB. An operation is decomposed into sequences of more fundamental send-receive-reply messages. The agents' relationship is characterized by exchanged messages that are grouped as interactions. The interactions are designed to perform an allowed sequence of business activities. The main purpose is to define a workflow and discipline the sequence of business activities to respect the e-Contract lifecycle. Through allowed interactions, the agents perform activities to match the rfp with published services, producing an electronic agreement. An agreement creates a binding between the parties and provides a guarantee of service delivery.

Let us consider an operation as a subset of a transaction. A transaction is an atomic unit of database access that is either completed or not executed at all (Ceri and Pelagatti, 1984). In terms of a transaction's duration, it can be short or long. The short-running transaction can be part of a long-running transaction. The WS-Transaction specification distinguishes an atomic transaction from a Business Activity (BA) (OASIS, 2012). BA is characterized for long-running transactions. The atomic transaction is committed and made visible before a long-running BA can be completed. Our approach considers each e-Contract lifecycle phase as a BA, and the operational request-response pattern as a single Atomic Transaction to perform the activities. As mentioned, the transactions performed into BB and PB change their internal states to build the agreement. Both are an artifact repository, BB supports the agreement structure on the client side and PB supports the agreement on the provider side. The *rfp* is used to build a draft on the client side. In contrast, PB supports the publication of services, i.e., the draft is built from service descriptions. Besides the activities association, PB and BB are scalable agents. The internal states of BB and PB are updated in their replicas to increase scalability and availability within the architecture. It expands services to client and providers avoiding the system failure.

The service acquisition is performed in two phases (2PC). In the first phase a *prewrite* (pw) operation holds the services. COAeB has a mechanism to match the *rfp* with published services offered in a pre-Contract format called matchmaking activity. If the whole demand was matched, then the coordinator receives all confirmations. Thus, the acquisition can be completed in the second phase of the protocol.

The online requests (*rfi* and *rfq*) also are used in the negotiation phase. The *rfi* is used to invite new publications and *rfq* to modify some attributes of a service that as previously published.

4 CASE STUDY

In this section, we present three possible flows of messages exchanged by the agents considering the set of services capabilities stored in the service registry. We propose a hypothetical scenario to illustrate the interactions in COAeB.

Supposing a client of the proposed application is an engineer that works for a company located in Virginia, US. He receives the following task: train new employees in a subsidiary located in California for six months. The engineer, a client of our approach, decides to buy several airline tickets to visit his family in Virginia every weekend. For him to visit every weekend, he will need 6x4, or 24 round trips in total. He includes the taxi service to-from airport in (VA) and (CA). He will need 4 taxis per round trip flight. We can also include an economic hotel to stay at weekdays. The RFP demand is a composition of flight ticket, taxi, and hotel service. The client can use his mobile device to access the application and send the rfp to start the proposal.

Based on the characteristics previously listed, we design three possible ways, at different flow of messages to reach the agreement that reflects the result of the matchmaking activity:

i. Published service fully matches the rfp without additional external requests.

ii. Published service can partially match the rfp. The rfi is used to invite new publications.

iii. Published services fully match the rfp, but the client can negotiate some attributes of service exploited in the pre-Contract. The rfq is used to start the negotiation phase.

The next subsection details these three ways. The sequence determines adjacent interactions. Interaction *n* starts only if the interaction *n-1* is completed.

4.0.1 The First Case: Previous Publications of Services Fully Match the rfp

The sequence illustrated in Figure 6 (1 to 6) and Figure 7 (7 to 10) and described in the following steps::

1. The client sends a complex service description via a rfp interaction.
2. BB receives the rfp, builds the draft, and forwards a complex service description to CP.
3. CP splits the service description into basic services. It uses the compose interaction to send resulting set of the services to TC.
4. Prewrite operation starts 2PC. TC requests PB to hold the appropriate resources held via a pre-Contract publication with providers.
5. After receiving the confirmations, TC orders t BB to prepare the pre-Contract with the services booked.
6. BB notifies the client that the pre-Contract is published and is ready to be signed.

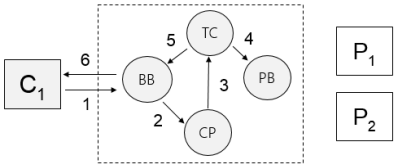

Figure 6: Previous publications fully match the rfp.

7. The client uses sign to confirm the agreement. It waits for the confirmation message from the BB.
8. The e-Contract with the client is stored in BB. Then, BB notifies TC that the agreement is established.
9. Write operation starts the second phase of the 2PC, global commitment.
10. PB forwards the write message to all providers to avoid the agreement's cancellation. After the providers' confirmation, PB builds the e-Contract with providers and notifies the providers that the agreement has been reached.

Step 7 occurs only if all the sequential operations 1-6 are completed.

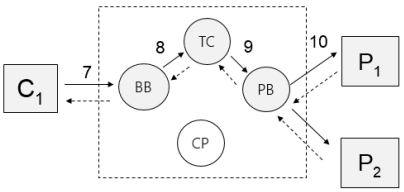

Figure 7: Previous publications fully match the rfp.

In our hypothetical scenario, previous publication of services match the rfp with the hotel, flight, and taxi services. The complex service is divided in basic services by CP.

4.0.2 The Second Case: Existing Previous Publication Partially Match the rfp

1. Client sends the *rfp* to BB.
2. BB receives the client's demands and forwards a complex service description to CP.
3. CP splits the demand into basic services and sends the request to TC via *compose* interaction.
4. TC starts 2PC. The composition result partially contemplates the *rfp*. The result is obtained from all responses in the first phase of 2PC.

At this point, the flow of messages can follow two ways: PB sends the *rfi* to the providers and waits for responses to fully match the *rfp* or BB publishes the pre-Contract on the client side and after the client signature, PB sends the *rfi*.

5. PB uses *rfi* to elicit changes in P1's.
6. P1 publishes new pre-Contract as well as, holds the resources offered.
7. PB forwards the resources held to TC, as the response of the first phase of 2PC. At this moment, all operations of the composition transaction are ready to commit. Now, the *rfp* fully matches published services.
8. TC forwards the result of the transaction to BB (publish interaction) then it prepares the pre-Contract for the client.
9. BB *notifies* the client to sign the pre-Contract. The approach solution includes a timeout as limitation to confirm the services acquisition.

With client signature, the sequence follows step 7 as presented in the first scenario.

In this second scenario, the flight ticket is not available after step 4. PB sends the rfi to invite P1 to publish a pre-Contract for an airline ticket with the attributes required on, step 5. After confirmation, PB sends the response to TC to continue with the primary flow. An alternative configuration is detailed below:

4. TC uses *composition* and receives some ready responses that partially matches the rfp, the first phase of 2PC.
5. Even so, TC sends *publish* to BB.
6. BB prepares the pre-Contract with the client as if it was a full match the rfp and notifies the client.
7. The client *signs* to confirm; BB holds the confirmation to give PB time to obtain resources. This time is determined by the timeout.
8. BB forwards the client confirmation to PB.
9. PB uses *rfi* to invite new publications.

The confirmation of step 7 is the agreement confirmation message. This message is sent only if newly published services fully match the rfp. Otherwise, the e-Contract is discarded.

4.0.3 Third Case: The Client Wants to Negotiate a Service Attribute

The third case starts after the publication of the pre-Contract on the client side, i.e., after notify between BB and client (step 7 of in the first case). The published pre-Contract fully matches rfp, but some services attributes do not satisfy the client. The approach is designed to allow negotiation. The negotiation period is set by a timeout in the pre-Contract.

The sequence described below begins with the publication of the pre-Contract with client in BB:

1. The client uses *negotiate* to request some modification in the published pre-Contract.
2. BB receives the client demands and notifies PB to identify potential providers to forward the request to them.
3. PB uses *rfq* to elicit pre-Contract updates by sending the rfq to appropriate providers.
4. If P1 can meet the client's demands it responds by sending a new publication, otherwise remains silent and the timeout is reached.
5. PB forwards the info to BB via *publish* interaction.
6. BB notifies C1 to sign the agreement.

If the provider agrees to the modification, then it needs to configure the service to be offered and lock the resources negotiated. Two situations can occur at this point. First, the provider updates the pre-Contract according to client. PB notifies the BB that some modification occurred after the negotiation. BB requests the client signature. After that point, it follows the step 7 of the first case. Another possibility can occur because a new priority is given to a client's demand, i.e., its prewrite receives a new timestamp order. Held resources are released and can be used to match other incomplete agreements. In our engineer's case, it corresponds to the hotel, flight and taxi matching rfp services. At this point, client is be able to start the negotiation of claims attributes.

4.0.4 The Operation Phase: Order, Consult or Cancel

In all cases, once the agreement is reached, BB and PB are responsible for receiving the client and provider demands in terms of agreement use. The agents can search the e-Contracts to control the rules agreed upon after receiving any request from the parties. In our approach, the operation phase represents the use of the e-Contract such as a warranty period. In this phase, the client can check the parameters agreed upon. Penalties can be established if rules are not fulfilled by the provider in terms of the service agreed upon. The client uses the order interaction with BB when the service can be delivery on demand.

1. BB verifies the e-Contract clause and sends the request to PB
2. PB stores the individual e-Contracts with providers and sends the request to the respective provider.

It is important to mention that the end of the negotiation phase allows the client to order the services. Internal agents work as mediators between the parties until the agreement's validity ends.

Let us illustrate the operation phase of our practical example, now the client can update its package or can order extra services always considering the agreement terms.

4.0.5 Combining Services via COAeB

Before introducing the composition of service in COAeB, it is necessary to describe the mechanism to produce files, called artifacts during the phases of the e-Contract lifecycle. First, the main file, that defines the agreement structure, is the template. It is the skeleton of the e-Contract. The different specialization of the agents BB and PB imposes the creation of

two types of template. Besides the template, the logical structure to support the building of the agreement with the client is different from the providers.

Let us consider the agreement on the client side. The e-Contract is built based on the rfp. The rfp is encapsulated within a template published in BB. Thus, it becomes a draft. The draft is a service description without an owner or a set of services needed to be matched with published services. Any client can use the draft to publish its demand. To differ from an agreement with providers, let us identify the agreement on the client side with number one,"1x".

To differentiate, the artifacts produced from template number two are identified by the additional information 2y. Figure 8 illustrates the arrangement of the e-Contract lifecycle for the COAeB architecture. The agreements are created concurrently on the client and provider sides to combines services. The e-Contract lifecycle of Agreement 1 is in white and the e-Contract lifecycle of Agreement 2 is in gray. The rfp starts a lifecycle of Agreement 1, and the rfi starts the lifecycle of Agreement 2. There is a mutual dependency between Agreements 1 and 2. Acronyms are used in the e-Contract lifecycle of Agreement 2.

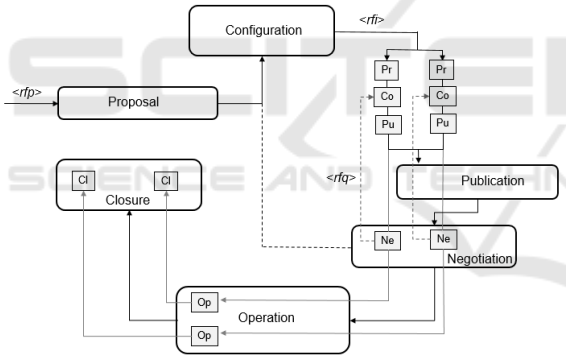

Figure 8: An arrangement of the e-Contract lifecycle that produces concurrent agreements with clients and providers.

The artifacts are built concurrently until the first dependence appears in the publication phase of agreement 1. Typically, draft 1x must be associated to two or more pre-Contracts 2y to reach its publication phase. If the requirements are met, draft 1x becomes pre-Contract 1x, as shown in Figure 8. The configuration phase of Agreement 1 is linked to its publication phase through the first phases of agreement 2.

The dependency creation is a part of the composition technique. Before completing the composition of the service process, the negotiation phase of both agreements occurs. The rfq (dotted lines from Ne to Co) can be used for conclude Agreement 2y during its negotiation phase, as shown in Figure 8.

Considering the first dependency, the condition

Table 1: Combine services by arrangement of Agreements on the client and provider side.

Lifecycle 1	Lifecycle 2
Proposal	
1: template 1	template 2
2: draft 1x	draft 2y,2t,2u,2v
Configuration	
3:	pre-Contract 2y,2t,2u,2v
Publication	
4:	PSL={2y,2t,2u,2v}
5: pre-Contract 1x (p2y,p2u)	
Negotiation	
6: pre-Contract 1x (2y,2u)={e_1x}	
7:	e-Contract 2y(e_1x)
8:	e-Contract 2u(e_1x)
Operation	
9: PAL={e_1x (e2y,e2u)}	
Closure	
10: PAL={ }	PSL={2t,2v}

to start the publication phase of Agreement 1 is that there is one or more type 2 agreements in publication phase as the reference. The second dependency concludes the negotiation phase of Agreement 2. The condition to start the operation phase of Agreement 2 is the signature of e-Contract 1x by the client. Arrows link Ne to Op of the lifecycle to support the production of Agreement 2 are inside the operation phase of the lifecycle to produce Agreement 1.

Let us formalize a relationship of dependency between Agreements (1) and (2). The first dependency appears from Agreement (1) to Agreement (2). We can illustrate P1x (P21, P22, P23,...,P2n) as the relationship of dependency to reach the publication of a Pre-Contract (1x). The second dependency can be represented by E2y (E1x). To reach an e-Contract (2y) it is necessary to match it to an e-Contract (1x). The composition of service process starts at the first dependency and is completed when the e-Contract (2y) is published. Each e-Contract (2y) matches just one e-Contract (1x); on the other hand, a Pre-Contract (1x) is matched to several Pre-Contract (2y). The dependency creates a bond between two agreements and, consequently links the client to providers. The period in which there is a mutual dependency is used to explore the negotiation of the attributes of the service.

The Public Source List (PSL) is stored in PB. It is a public list available for all thrust suppliers. It stores pre-Contracts type 2. PSL is verified in the matchmaking activity, and it is updated in two cases. First, when a pre-Contract publication occurs, and second, once the composition process is completed. In terms of scalability, every time an update in PSL is performed, all active PBs must be replicated. In terms of scalability, every time an update in PSL is performed, all active PBs must be replicated. There-

fore, PAL is a private list available only to the interested parties involved in the agreement. It saves (1) e-Contracts. BB stores the Private Agreement List (PAL). All (1) e-Contracts save (2) e-Contracts information. Thus, PAL is a repository of e-Contracts while PSL is a repository of pre-Contracts. Both lists, PSL and PAL, are verified during the composition process. Tuples saved in the PSL are moved to PAL after the matchmaking activity has been completed, and an (1) Agreement has been reached.

The composition process consists of three activities listed below:

i. Service identification breaks a complex service described in rfp into basic services. It works when the first dependency appears; a draft 1 is produced to be matched by published services stored in PB.

ii. Matchmaking receives the client demand, searches for (2) pre-Contracts in PSL with related requirements and creates the first bond between (1) and (2) agreements.

iii. Service composition is divided into two phases. The composition uses 2PC protocol to complete the (1) and (2) agreements efficiently. The services held via pre-Contract publication need to be booked during the first phase of the composition. The second phase completes the composition process and stores the (1) e-Contracts in PAL.

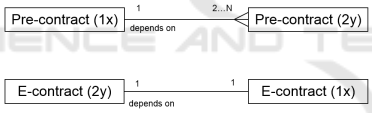

Figure 9: Mutual dependency between agreement 1 and agreement 2.

By following the steps 1-10 presented in Table 1, it is possible to summarize the arrangement of lifecycles from agreements 1 and 2, shown in Figure 8. Starting in the proposal phase, steps 1-2, the drafts type 1 and 2 are instantiated concurrently from their templates. Step 3 illustrates the production a of a set from the pre-Contracts type 2. Step 4 shows the pre-Contracts as the elements of the PSL set. Note that these pre-Contracts can be composed to attend the requirement of the pre-Contract 1x. The negotiation starts during the step 6, i.e., the client can propose changes in the pre-Contracts 2y and 2u, and after that the e-Contract e1x is produced. The steps 7-8 illustrate the dependency to reach the e-Contracts 2y and 2u. The step 9 presents the PAL list updated and step 10 the pre-Contracts available in the PSL, pre-Contracts 2t and 2v.

5 CONCLUSIONS AND FUTURE WORK

Based on e-Contract lifecycle, the Client-Oriented Architecture uses broker mechanism to combine services in the distributed environment. The goal of the Client-Oriented approach is to combine services to match client needs following e-Contract lifecycle methodology. The architecture and logical design of the approach propose a simple way to standardize and meet the business rules focused on client needs. The great advantage of the Client-Oriented architecture is its capacity to adapt the usage patterns by extending the capabilities of protocols and architectures.

The great limitation is the use of the Template. The Template's objective is include the semantic necessary to express the agreement terms. The Template structure limits the scope of the e-Contract and needs to be produced and updated manually.

Future work includes server failures, network partitions or disconnected operations. About the evolution of the approach, enable the customers can propose a composition based in the historical cache.

REFERENCES

Andrew Goodchild, C. H. and Milosevic, Z. (2004). Business contracts for b2b.

Angelov, S. and Grefen, P. (2004). The business case for b2b e-contracting. In *Proceedings of the 6th international conference on Electronic commerce*, ICEC '04, pages 31–40, New York, NY, USA. ACM.

Bernardo Neto, J. and Hirata, C. (2015). e-business architecture for web service composition based on e-contract lifecycle. In *Proceedings of the 17th International Conference on Enterprise Information Systems - Volume 3*, ICEIS 2015, pages 276–283, Portugal. SCITEPRESS - Science and Technology Publications, Lda.

Ceri, S. and Pelagatti, G. (1984). *Distributed Databases Principles and Systems*. McGraw-Hill, Inc., New York, NY, USA.

Chiu, D., Cheung, S., and Till, S. (2003). A three-layer architecture for e-contract enforcement in an e-service environment. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, pages 10 pp.–.

da Silva Maciel, L. A. H. and Hirata, C. M. (2010). A timestamp-based two phase commit protocol for web services using rest architectural style. *J. Web Eng.*, 9(3):266–282.

Gong, M. et al. (2013). Recommending web service based on user relationships and preferences. In *Web Services (ICWS), 2013 IEEE 20th International Conference on*, pages 380–386.

Mitra, G. et al. (2013). A request oriented model for web services. In *Proceedings of the First Australasian Web Conference - Volume 144*, AWC '13, pages 13–20, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Neto, J. B. and Hirata, C. M. (2013). Lifecycle for management of e-contracts based on web service. In *Proceedings of the World Congress on Engineering and Computer Science*, volume 1.

OASIS (2012). Oasis web services transaction (ws-tx) tc.

Pardon, G. and Pautasso, C. (2014). Atomic distributed transactions: A restful design. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 943–948, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.