Choreography Conformance Analysis: Asynchronous Communications and Information Alignment[⋆]

Raman Kazhamiakin and Marco Pistore

DIT, University of Trento
via Sommarive 14, 38050, Trento, Italy
{raman, pistore}@dit.unitn.it

Abstract. Web service choreography languages provide a way to describe the collaboration protocol of multiple services that exchange information in order to achieve a common goal. This description may be seen as a specification that should be respected by the joint behavior of the set of services implementing the choreography. Such a conformance requires that (*i*) the observable behavior of the implementation corresponds to the behavior described by the protocol specification, and (*ii*) the business information is properly managed, guaranteeing that the participants have a shared knowledge about it, according to what is specified in the choreography. In this paper we present a choreography conformance analysis approach that addresses both the behavioral correspondence and the business information management. The key features of the approach are the capability to deal with asynchronous interactions and the ability to model and analyse the data managed and exchanged in the protocol, thus providing more accurate verification results. We also present symbolic techniques based on these formalizations that can be used for model checking of the choreography conformance.

1 Introduction

Web service technology enables the development of complex heterogeneous, distributed applications, facilitating the specification, deployment, and enactment of remote software components accessible on the web via standardized protocols. The ability to integrate the existing services owned and managed by distinct stakeholders, obtaining new composite business applications, is one of the fundamental ideas underlying the Web service technology paradigm. Among the various aspects that need to be specified to fully describe a Web service composition, the representation of a stateful and coordinated behavior of the composition plays a prominent role. A wide range of Web service standards and languages has been proposed for these purposes [1,2,3]. The Web Services Choreography Description Language (WS-CDL, [3]) is particularly relevant for the specification

⋆ This work is partially funded by the MIUR-FIRB project RBNE0195K5, "KLASE", by the MIUR-PRIN 2004 project "STRAP", and by the EU-IST project FP6-016004 "SENSORIA".

of the compositions, as it provides a way to describe the observable behavior of the collaboration from the global point of view. One of the main goals of the choreography description is to define a reference model of the composition that the real service implementations should conform to. *Conformance testing* refers to the verification that the joint behavior of the composition of the service implementations corresponds to that described in the choreography.

The conformance analysis, however, does not amount only to check the correspondence between the sequences of externally observable message exchanges generated by the composition of service implementations and the collaboration protocol specification. It is also necessary to verify that the information of the protocol is being managed and distributed accordingly, and that the participants have a common view of the business data described in the choreography. The management of business information in conformance testing is complicated by the fact that WS-CDL allows for specifying in a declarative way that certain pieces of information should be synchronized either as a result of a certain data exchange (interaction alignment) or of the protocol execution as a whole (choreography coordination), without explicitly describing and constraining the mechanisms that should implement them.

In this paper we present a formal analysis framework that allows for the verification of the conformance between the collaboration specification and the composition of service implementations. The presented framework is based on our previous work [4] that provides a formal model for the compositions of *local* participants implementations. The key feature of this framework is the ability to model and analyse compositions, where the interactions are asynchronous, and the messages may be reordered and stored in unbounded queues.

In this work, we extend the approach of [4] in two ways. First, we enrich the model with the capability to represent and manage data-related constructs (e.g., variables, conditions, assignments), thus providing a way to model the data-flow of the compositions. Second, we introduce a formalism for the *global* model that allows for the choreographic description of the compositions. Based on these formalisations, we define the choreography conformance as a kind of bisimulation relation, emphasizing the asynchrony of the message communications. We also present formal definitions for the most common information alignment requirements, such as the interaction coordination alignment rules presented by WS-CDL. Furthermore, we define a symbolic representation of the underlying models, and propose finite-state model checking techniques for verifying the conformance between the implementing composition and the choreography specifications.

The paper is structured as follows. Section 2 introduces the conformance problem using variants of a simple example. Section 3 defines the formal models for the data- and control-flow of the underlying systems from the global perspective and as a composition of interacting local services. In Sect. 4 we present the notions of the asynchronous conformance relation and the information alignment rules, and discuss the symbolic analysis techniques suitable for the conformance verification. Concluding remarks and related works are discussed in Sect. 5.

2 Modelling Web Services Compositions

In order to illustrate the problems related to the conformance between the specification of a Web service composition and its implementation, we consider several variants of the Request For Quotation (RFQ) case study. The goal of the composition is to combine purchasing and delivery functionalities in a single business process, involving several participants. Thus, the composition describes the interactions of three independent services, namely a *buyer*, a *seller*, and a *shipper*.

We model the scenario using a WS-CDL [3] specification that describes the collaborations between the participants from the global perspective. WS-CDL specifications identify the participants of the composition, their variables, the interactions between the partners, and the dependencies between these interactions, such as control-flow and data-flow dependencies, transactional requirements etc. An example of the choreography specification is represented as a UML activity diagram in Fig. 1(a). The elementary actions in the diagram represent message exchanges, like `request` or `offer`; the decisions points, like the choice to accept or reject the offer; the silent internal activities, like the `verify` activity used to check the presence of the product.

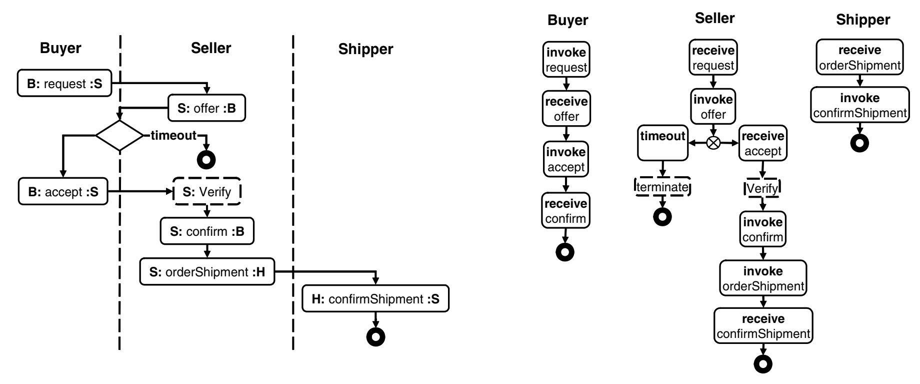
The composition implementation is represented as a set of local specifications, one for each participant of the collaboration, defined in an appropriate language, e.g. BPEL [1]. These local models may represent either the real services, or rather the behavioral interfaces of the participants, to which the real implementations should conform [5]. Each local specification describes the (stateful) behavior of a particular service. It defines the operations that are triggered upon the invocation of the service. These operations include variables assignments, invoking other services and receiving responses, and structured activities like sequences, loops, conditional choices, etc. Examples of the local protocols, as those of the buyer and the seller, are represented as UML diagrams in Fig. 1(b).

It is important to note that the implementation description may include significantly more activities and even participants than is specified in the choreography description. These auxiliary elements are used to ensure the protocol, coordinating and aligning the main parts of the system.

2.1 Behavioral Correctness

The choreography model represented in Fig. 1(a) describes the following business scenario. First, the buyer asks the seller for a particular good, sending a request for quote. The offer is prepared and sent back to the buyer. In this moment two situations are possible: either the buyer accepts the message and the process continues with the confirmation and a shipment engagement; or the acceptance does not happen within a certain time limit, the offer is considered invalid, and the whole procedure terminates.

This choreography specification defines the requirements to the implementing compositions. That is, an implementation should satisfy all the control-flow and data-flow requirements of the model. Consider the BPEL processes that are supposed to implement the participants of the above scenario (Fig. 1(b)). It is easy



(a) (b)

Fig. 1. RFQ case study

to see that these processes, when composed together, satisfy the specified choreography only under the assumption that the interactions are *synchronous*, i.e., a message emission is possible only if it is immediately followed by a reception. This assumption, often used in modelling of Web service compositions, may be violated in real settings due to the *asynchronous* nature of Web service interactions. Indeed, since the buyer and the seller are independent, it is possible that the former emits the acceptance message simultaneously with the timeout of the seller. This leads to a state, where the seller has terminated the execution, while the buyer waits for the offer confirmation. This scenario may not be detected if the assumption on synchronous interactions is applied.

In order to satisfy the choreography specification, some auxiliary activities should be performed. In particular, the `accept` message should follow some availability checking interaction, where the the buyer asks for the possibility to accept the order. In case of positive response, the acceptance is invoked, otherwise the buyer terminates. On the other side, the seller waits for this availability checking message, and responds negatively only if the timeout has expired.

2.2 Information Alignment

Figure 2(a) represents a modified choreography specification of the RFQ case study. Here, instead of termination on timeout, the seller iteratively provides the buyer with the updated information about the requested product (interaction `refresh`), until the latter does not accept the offer.

The process implementations of the participants are presented in Fig. 2(b). In the buyer process the decision to accept the offer is performed in parallel with the loop, where the offer information is continuously updated on the reception of `refresh` message. Analogously, the seller repeatedly waits for either an acceptance message or for a timeout expiration. The boolean variables (`done` and `accepted`) are used to control termination of the loops.

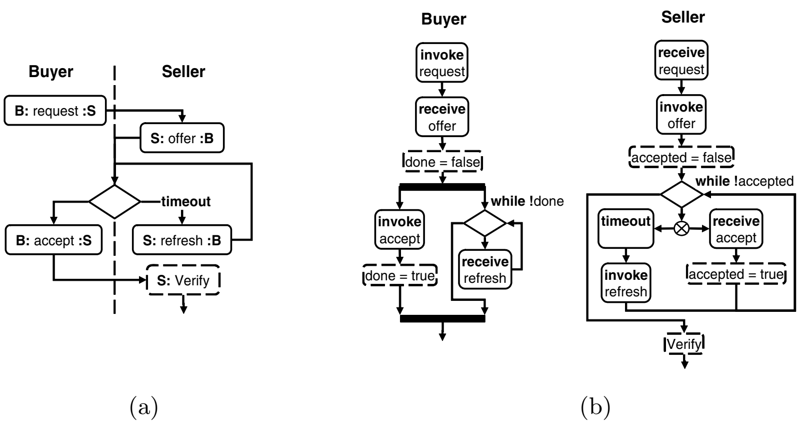


Fig. 2. RFQ case study: iterative quoting

The most relevant property that this choreography should satisfy is the *information alignment* between the participants. In particular, when the offer is accepted after several updates, both the buyer and the seller should have a common knowledge on the current offer instance. Such a requirement is modelled in a declarative way in WS-CDL, by marking certain interactions (e.g. `refresh` and `accept`), as *aligned* interactions.

It is easy to see that the given implementation may violate this requirement. Since the partners are independent, the timeout and acceptance invocations may happen simultaneously. As a result, the local values of the accepted offer may be different. Another negative scenario happens when the acceptance is performed after several updates. If the message queue of the buyer service is not ordered, there is no guarantee that the accepted offer is the last emitted by the seller. The necessity to guarantee the correctness on the information alignment requires the analysis techniques that go beyond the verification of the behavioral correctness.

2.3 Composition Coordination

Apart from the alignment of a particular interaction, it is often required that the participants of the choreography agree on the final state of the collaboration activity. This requirement, referred in WS-CDL as choreography coordination, states that either all the participants suffered an exception, or all of them completed successfully (and, consequently, their finalization is also agreed). In WS-CDL notation it is allowed to declare a coordination requirement without explicitly modelling the corresponding coordination interactions. A choreography implementation, however, should satisfy this requirement by providing special coordination message exchanges.

Consider the choreography model represented in Fig. 3(a). After the confirmation of the availability of the product, the seller interacts with a new actor, namely Credit Card Agency (CCA), in order to verify the payment information

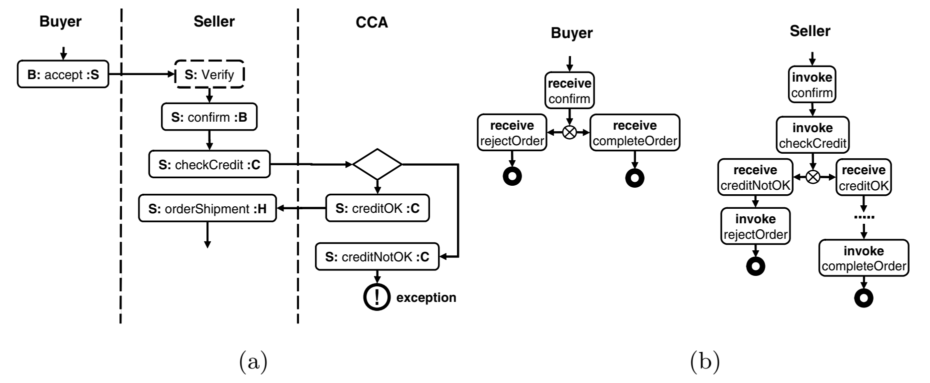


Fig. 3. RFQ case study: coordination management

of the buyer. This verification may result in a fault message, and the seller enters its exception state. In order to guarantee that the exception is propagated to the buyer, an auxiliary communication should be instantiated between these actors. The corresponding implementation is presented in Fig. 3(b). After the reception of the order confirmation the buyer waits for the additional messages that allow to distinguish the resulting state of the protocol. If the credit check failed, the seller sends the `rejectOrder` message to the buyer, and the latter knows that the exception occurred and the composition terminated abnormally.

These examples illustrate two important problems that should be addressed by a formal framework for validation of choreography specifications against compositions of service implementations. First, it is necessary to check the conformance of the behavior of the composition of services to the behavior, described in the choreography model. Doing this, it is important to take into account the asynchronous nature of the Web service communications, i.e., the possibility of message intersections and reorderings, variety of the implementations of the queueing mechanisms. Second, it is necessary to validate that the implementation satisfies the alignment requirements, declared in the choreography model, as those reflecting the interaction alignment and the choreography coordination.

3 Formalization

The formal model we use as a basis for the required analysis techniques consists of three parts, namely the *data model*, the *choreography model*, and the *implementation model*. The data model provides a formalisation of the data manipulated by the services and is used to reason on the data flow of the compositions. The control flow on the other hand, is defined by the choreography model, used to represent a behavior of the WS-CDL specification, and by the implementation model, used to represent a behavior of the composition of several existing services, specified, e.g., in BPEL [1].

3.1 Data Model

We model the data manipulation in Web service compositions using the following notations. Given a set of typed variables $\mathcal{V}$ and a set of typed functions $\mathcal{F}$, the expressions and terms over the variables and functions are defined as follows:

- $E \equiv (t_1 = t_2) \mid \neg e \mid (e_1 \wedge e_2)$ that is equality between terms, negation or disjunction of expressions;
- $T \equiv v \mid f(t_1, \ldots, t_n)$, with $v \in \mathcal{V}$ and $f \in \mathcal{F}$, that is a variable or function call on terms.

We assume a fixed interpretation of typed functions. For the interpretation of variables, instead, we use *valuation* functions g that map variables $v \in \mathcal{V}$ to their values. We write $g \models e$ to denote that the expression e evaluates to true under the valuation g. A *condition* $\phi \in \Phi$ is an expression of the form presented above. An *assignment* $\omega \in \Omega$ has the form $(v := t)$. We denote an *update* of the valuation g with the assignment ω as $upd(g, \omega)$.

3.2 Choreography Model

The formal model of choreography is based on the notion of *roles* and *actions*. A role represents the behavior of a particular participant of the composed system. During the protocol execution, the role can be in one of its possible *states* and can evolve to new states as a result of performing some actions. Moreover, each role is possibly equipped with a set of typed variables.

We model message communications actions as *interactions* defined on a set of service operations (or message types) M. The signature of the interaction has the form $(r_s, r_d, \mu, \bar{v}_s, \bar{v}_d)$, where r_s and r_d are the roles of the sender and receiver respectively, μ is the service operation, and variables $\bar{v}_d$ of the receiver are populated with the values of the corresponding variables $\bar{v}_s$ of the sender. Set of interactions is denoted as $\mathcal{A}_O$.

We also define *internal actions* $\mathcal{A}_\tau$, which are used to represent evolutions of the system that do note involve interactions between services. An internal action a_τ has the form $(\mathcal{R}_\tau, \tau)$, where $\mathcal{R}_\tau \subseteq \mathcal{R}$ denotes a subset of roles that perform an action, and τ is used to denote the internal action itself[1]. The set of all actions is denoted as $\mathcal{A}$.

We model a choreography behavior as a *Global Transition System* (GTS). Informally, we represent a *global state* of the choreography as a vector $\bar{s} = \langle s_1, \ldots, s_n \rangle$, where s_i is a local state of the role r_i. We denote a vector with component s_i updated to s'_i as $\bar{s}[s'_i/s_i]$. The behavior of the choreography is defined by the *global transition relation* $\mathcal{T}$. The relation defines *conditions*, under which the action can be performed, and *effects* of these executions, which specify the modification of the states and variables of the participants.

[1] The possibility of a group of participants to participate to an internal action is used in WS-CDL to model that the branching condition may be evaluated simultaneously by a group of roles.

Definition 1 (GTS). *A* global transition system *representing the choreography of n roles is a tuple $\Sigma_p = \langle \mathcal{V}, \mathcal{S}, \mathcal{S}_0, \mathcal{A}, \mathcal{T} \rangle$, where*

- *$\mathcal{V} = \bigcup_i \mathcal{V}_i$ is a set of all the role variables;*
- *$\mathcal{S} \subseteq \mathcal{S}_i \times \cdots \times \mathcal{S}_n$ is a finite set of global states and $\mathcal{S}_0 \subseteq \mathcal{S}$ is a set of initial states;*
- *$\mathcal{A} = \mathcal{A}_\tau \cup \mathcal{A}_O$ is a set of actions;*
- *$\mathcal{T} \subseteq \mathcal{S} \times \Phi \times \mathcal{A} \times \Omega^* \times \mathcal{S}$ is a global transition relation. A transition $(\bar{s}, \phi, a, \Omega, \bar{s}') \in \mathcal{T}$ if*
  - *$a = (r_a, r_b, \mu, \bar{v}_s, \bar{v}_d)$ and $\bar{s}' = \bar{s}[s'_a/s_a, s'_b/s_b]$, or*
  - *$a = (\mathcal{R}_\tau, \tau)$ and $\bar{s}' = \bar{s}[s'_i/s_i]$ for each $r_i \in \mathcal{R}_\tau$.*

Let $\gamma = \langle \bar{s}, g \rangle$ be a *configuration* of the choreography. The transition of GTS $(\bar{s}, \phi, a, \Omega, \bar{s}')$ is fireable in γ only if $g \models \phi$. The resulting configuration is defined as $\langle \bar{s}', upd(g, \Omega) \rangle$. We write $\gamma \xrightarrow{\mu} \gamma'$, if the action a has the form $(r_s, r_d, \mu, \bar{v}_s, \bar{v}_d)$, and $\gamma \xrightarrow{\tau} \gamma'$ otherwise. We denote a set of transitions, fireable in γ, as $out(\gamma)$.

3.3 Implementation Model

We model a system that implements a given choreography specification as a composition of *local transition systems* (LTSs). Each LTS represents the behavior of one of the participants. The implementation model may include more participants, interactions and operations than those declared in the choreography specification. In particular, these participants and/or operations may describe the low-level mechanisms that are used to implement the coordination requirements, declared in the choreography specification.

The behavior of the participant is defined using set of local variables and local actions. The local actions of the i^{th} participant are divided into *input actions* $\mathcal{I}^i$, representing the reception of message α, denoted as $\overleftarrow{\alpha}$; *output actions* $\mathcal{O}^i$, representing the emission of message α, denoted as $\overrightarrow{\alpha}$; and internal actions $\mathcal{A}^i_\tau$. A message $\alpha \in \mathrm{M}_\alpha$ has the form $\mu(\bar{x})$, where μ is the service operation, and $\bar{x}$ denotes a message content.

Definition 2 (LTS). *A local transition system representing the i^{th} participant of the implementation model is a tuple $\langle \mathcal{V}^i, \mathcal{S}^i, \mathcal{S}^i_0, \mathcal{A}^i, \mathcal{T}^i \rangle$, where*

- *$\mathcal{V}^i$ is a set of local variables;*
- *$\mathcal{S}^i$ and $\mathcal{S}^i_0$ are the finite sets of local states and initial local states respectively;*
- *$\mathcal{A}^i = \mathcal{I}^i \cup \mathcal{O}^i \cup \mathcal{A}^i_\tau$ is a set of local actions;*
- *$\mathcal{T}^i \subseteq \mathcal{S}^i \times \Phi \times \mathcal{A}^i \times \Omega^* \times \mathcal{S}^i$ is a local transition relation.*

We define a composition of local participants as follows. During the execution, the composition participants evolve independently, exchanging messages with other participants through a certain communication medium, represented as a set of queues. We refer to this medium as *communication model.* The behavior of the composition strongly depends on the structure of the communication model: the number of queues, queue ordering, queue bounds etc. An example of this dependency is illustrated in Sect. 2.1, where the composition of service implementations conforms to the specification behavior under the synchronous communication model, but violates it under a more realistic asynchronous model. Therefore, the implementation model should be correct with respect to the choreography specification regardless to the communication model applied for the composition representation.

In our previous work [4], we present a hierarchy of communication models and introduce the notion of the *most general communication model* (MG-model). We show that under this model any composition of LTSs exhibits more behaviors than the composition under any other communication model. The MG-model is defined as a structure with one unbounded and unordered queue. That is, all the exchanged messages are stored in and consumed from this queue regardless their ordering[2]. We will use this model to represent and analyse the composition of the local transition systems.

Let $\mathbb{N}^{\mathrm{M}_\alpha}$ be a set of multisets of M_α, i.e. set of mappings from M_α to natural number $\mathbb{N}$. Given two elements w and w', we write $w.w'$ to denote the multiset union, if $w, w' \in \mathbb{N}^{\mathrm{M}_\alpha}$. Thus, the queue content is defined as a multiset w.

Definition 3 (CTS). *A* composition transition system *representing the composition of n participants is a tuple* $\Sigma_c = \langle \mathcal{V}^c, \mathcal{S}^c, \mathcal{S}_0^c, \mathcal{A}^c, \mathcal{T}^c \rangle$*, where*

- $\mathcal{V}^c = \bigcup_i \mathcal{V}_i$ *is a set of all local variables;*
- $\mathcal{S}^c$ *is a set of composition states of the form* $\langle \bar{s}, w \rangle$*;*
- $\mathcal{S}_0^c \subseteq \mathcal{S}^c$ *is a set of initial composition states with empty queue* $w = \epsilon$*;*
- $\mathcal{A}^c = \bigcup_i \mathcal{A}^i$ *is a set of actions;*
- $\mathcal{T}^c \subseteq \mathcal{S}^c \times \Phi \times \mathcal{A}^c \times \Omega^* \times \mathcal{S}^c$ *is a composition transition relation. The transition* $(\langle \bar{s}, w \rangle, \phi, a, \Omega, \langle \bar{s}', w' \rangle) \in \mathcal{T}^c$ *if for some i there exists a transition* $(s_i, \phi, a, \Omega, s_i') \in \mathcal{T}^i$ *such that* $\bar{s}' = \bar{s}[s_i'/s_i]$ *and*
  - *if* $a = \overrightarrow{\alpha}$*, then* $w' = w.\alpha$*;*
  - *if* $a = \overleftarrow{\alpha}$*, then* $w = \alpha.w'$*;*
  - *if* $a = \tau$*, then* $w' = w$*.*

The behavior of the composition is defined analogously. Let us denote the configuration of the composition as a triple $\gamma = \langle \bar{s}, g, w \rangle$. The transition of CTS $(\langle \bar{s}, w \rangle, \phi, a, \Omega, \langle \bar{s}', w' \rangle)$ is fireable in γ only if $g \models \phi$. The resulting configuration is defined as $\langle \bar{s}', upd(g, \Omega), w' \rangle$. We write $\gamma \xrightarrow{\overrightarrow{\mu}} \gamma'$, if the action a has the form $\overrightarrow{\mu}(\bar{x})$, $\gamma \xrightarrow{\overleftarrow{\mu}} \gamma'$, if the action a has the form $\overleftarrow{\mu}(\bar{x})$, and $\gamma \xrightarrow{\tau} \gamma'$ otherwise.

A (possibly infinite) sequence $\pi = \gamma_0, a_0, \gamma_1, a_1, \ldots$ is a *run* of the CTS, if $\gamma_0 \in \mathcal{S}_0^c$, and for any $i \geq 0$ $\gamma_i \xrightarrow{a_i} \gamma_{i+1}$.

4 Choreography Validation

An important issue in the analysis of Web service specification is verifying that the given composition of existing services satisfies the requirements of the spec-

[2] If certain interaction constraints (e.g., synchronizability, message ordering) should be satisfied by the composition, a corresponding communication model may be used instead of MG-model. See [4] for the details on the analysis and implementation.

ified global choreography protocol. This analysis has to address the following problems. First, it has to check that the behavior exhibited by the composition corresponds to those described in the choreography document. This problem is referred to as *conformance checking* [6]. Second, it is often needed that the participants agree on the state of the protocol as a result of its execution. In other words, they expect to have a common knowledge on certain variables that describe the state of the protocol. This problem is referred to as *information alignment.*

4.1 Choreography Conformance

In [7] the notion of conformance between choreography and orchestration (i.e. implementation specification) was introduced as a bisimulation-like relation. However, some crucial aspects are ignored in that framework. The model of the composition, adopted in this framework, relies on the assumption that the message exchanges are synchronous, which is often not realistic in the Web service environments. As a consequence, it is not always possible to reveal the implementation problems like, e.g., the message losses, queue unboundedness, message intersections and disorder.

We extend the presented approach in the following way. Given an implementation specification, we model the composition of participants in the most “liberal” (i.e., with respect to the possible behaviors) settings, that is, under the most general communication model. We require that the following properties hold on the resulting composition:

– the composition specification is *complete*, i.e. all the messages send by any participant should be eventually consumed by the recipient;
– the composition is *bounded*, that is there exists such a constant K that in every reachable configuration of the composition the number of messages in the queue is less than this constant: $|w| \leq K$;
– the (relevant part of) observable behavior of the implementation is *similar* to the behavior of the choreography specification.

More formally, we define the notion of conformance as follows. Let M^p be a set of service operations of the choreography specification. In order to hide irrelevant operations of the implementation, we use the operator $[\cdot]$. That is, given an action $a \in \mathcal{A}^c$, we write $[a] = \mu$, if $a = \overrightarrow{\mu}(\bar{x})$ and $\mu \in \mathrm{M}^p$, and $[a] = \tau$ otherwise. We write $\gamma^c \xrightarrow{\tau}{}^* \gamma^c_1$ to denote that γ^c_1 is reachable from γ^c through (zero or more) irrelevant operations. Analogously, $\gamma^p \xrightarrow{\tau}{}^* \gamma^p_1$ means that γ^p_1 is reachable from γ^p through (zero or more) internal actions.

The conformance relation requires that conversations of the implementing composition reflects all and only the conversations of the choreography.

Definition 4 (Conformance Relation). *Let γ^p and γ^c be configurations of Σ_p and Σ_c respectively. We say that the relation $R(\gamma^p, \gamma^c)$ is a* conformance relation *if for any transition label a*

– *if $\gamma^p \xrightarrow{a} \gamma^p_1 \ \wedge\ a = \mu$, then $\gamma^c \xrightarrow{\tau}{}^* \gamma^c_2 \ \wedge\ \gamma^c_2 \xrightarrow{\overrightarrow{\mu}} \gamma^c_1 \ \wedge\ R(\gamma^p_1, \gamma^c_1)$;*

– *if* $\gamma^c \xrightarrow{a} \gamma_1^c \ \wedge\ [a] = \mu$*, then* $\gamma^p \xrightarrow{\tau}{}^* \gamma_2^p \ \wedge\ \gamma_2^p \xrightarrow{\mu} \gamma_1^p \ \wedge\ R(\gamma_1^c, \gamma_1^p)$*;*
– *if* $\gamma^p \xrightarrow{a} \gamma_1^p \ \wedge\ a = \tau$*, then* $\gamma^c \xrightarrow{\tau}{}^* \gamma_1^c \ \wedge\ R(\gamma_1^p, \gamma_1^c)$*;*
– *if* $\gamma^c \xrightarrow{a} \gamma_1^c \ \wedge\ [a] = \tau$*, then* $\gamma^p \xrightarrow{\tau}{}^* \gamma_1^p \ \wedge\ R(\gamma_1^c, \gamma_1^p)$*.*

We write $\Sigma_p \approx \Sigma_c$ *if there exists a conformance relation* R*, such that any initial configuration of* Σ_p *conforms to some initial configuration of* Σ_c*, and vice versa.*

Definition 5 (Asynchronous Choreography Conformance). *An implementing composition* Σ_c *is* asynchronously conformant *to the choreography* Σ_p*, if* Σ_c *is complete, bounded, and* $\Sigma_c \approx \Sigma_p$*.*

4.2 Information Alignment

An interesting property being modelled in the choreography specifications is the information alignment, i.e. the ability to control that the participating roles agree on the outcome of the interactions or even of the execution of the whole protocol [3]. In particular, in the scenario in Fig. 2(a) it is required that both the buyer and the seller have a have a common view on the offer value. That is, the partners may need to have a common knowledge on the information they exchange (interaction based alignment). As a result of such an alignment the participants act on the basis of their shared knowledge. In other cases, like those illustrated in Fig. 3(a), this property expresses a requirement that the participant will agree on the way the choreography ended, regardless the alignment of intermediate interactions (choreography coordination). In either case, the implementing system should ensure that the specified requirements are satisfied (i.e., the interaction complete and the partner have the same information understanding, or choreography termination state is agreed).

Following the above patterns, we distinguish two kinds of properties to be modelled and validated on the implementing composition. The properties of the first group are used to check the proper interaction completion and the corresponding data alignment. The property of the second group are used to verify that the participants have a common view on the termination state.

More formally, let $a = (r_s, r_r, \mu, \bar{v}_s, \bar{v}_r) \in \mathcal{A}_O$ be an interaction action whose alignment has to be ensured. Let also ϕ be an expression over the variables of the partners that is expected to evaluate to true on the completion of the interaction. The interaction alignment rule requires that any emitted message should be eventually consumed, a new message can not be emitted until the previous is consumed, and the values of the variables should satisfy the expression on the interaction.

Definition 6 (Interaction Alignment Rule). *An* interaction alignment rule $\langle (r_s, r_r, \mu, \bar{v}_s, \bar{v}_r), \phi \rangle$ *requires that for any run* $\pi = \gamma_0, a_0, \gamma_1, a_1, \ldots$ *of* Σ_c*, if* $\gamma_i \xrightarrow{\overrightarrow{\mu}} \gamma_{i+1}$ *for some* $i \geq 0$*, then*

– *there exists* $j > i$*, such that* $\gamma_j \xrightarrow{\overleftarrow{\mu}} \gamma_{j+1}$*, and*
– *for any* $i < k < j$ $a_k \neq \overrightarrow{\mu}$*, and*
– $\gamma_{j+1} \models \phi$*.*

Consider an example choreography and implementation in Fig. 2(a) and 2(b) respectively. The interaction alignment rule for the `refresh` interaction has the form $\langle (s, b, \mathit{refresh}, \mathit{sOffer}, \mathit{bOffer}), (\mathit{sOffer} = \mathit{bOffer})\rangle$. It is easy to see that the rule is violated by the implementation.

The coordination alignment rule requires that the participants agree on the information in a termination state of the choreography. Given some termination state $\bar{s}$, let $\phi_{\bar{s}} = \phi^1_{\bar{s}} \wedge \cdots \wedge \phi^n_{\bar{s}}$ be an expression over the implementation that evaluates to true if and only if the participants are in the required state. Let E be a set of the all the expressions of the terminating states: $E = \{\phi_{\bar{s}}\}$.

Definition 7 (Coordination Alignment Rule). *A* coordination alignment rule $E = \{\phi_{\bar{s}}\}$ *requires that*

- *for each γ^c of Σ_c, with $out(\gamma^c) = \emptyset$, there exists $\phi_{\bar{s}} \in E$, such that $\gamma^c \models \phi_{\bar{s}}$;*
- *for each $\phi_{\bar{s}} \in E$, there exists γ^c of Σ_c, such that $out(\gamma^c) = \emptyset$ and $\gamma^c \models \phi_{\bar{s}}$.*

The coordination requires that each termination state of the implementation should correspond to some termination state of the choreography, and every termination state of the choreography is also a termination state of the implementation.

For the protocol represented in Fig. 3(a) the coordination alignment rule is formulated as follows:

$$E = \left\{ \begin{array}{l} (b.state = done \wedge s.state = done \wedge c.state = ok \wedge h.state = done), \\ (b.state = fail \wedge s.state = fail \wedge c.state = fail \wedge h.state = init) \end{array} \right\}$$

That is, either all the partners are in their successful states, or the buyer the seller and the CCA services fail, and the shipper is not initiated.

4.3 Choreography Analysis

The formal model represented above allows for the definition of systems with potentially infinite number of reachable configurations. This makes the application of formal analysis techniques very complex, if at all possible. In order to be able to perform the choreography conformance validation, the model should be made finite. For these purposes, we recall the approach of [8,9], and for the lack of space we only sketch the formalization here.

Symbolic Representation. We represent the composition models using an abstraction-based approach [8,9]. In this model the variables and their valuations are given in terms of valuations of the set of propositions. These propositions may express certain facts about the composition states, variables, relations between them, function values, etc. More formally, we allow the proposition to have a form of expression: $p \equiv (t_1 = t_2) \mid \neg p \mid p_1 \wedge p_2$. We will refer to the set of propositions as $\mathcal{P}^A$.

We define an abstract model corresponding to the concrete one, based on the set $\mathcal{P}^A$. An abstract valuation g^A is simply a mapping from $\mathcal{P}^A$ to $\{true, false\}$.

Since the set $\mathcal{P}^A$ is finite, a set of concrete valuations corresponds to an abstract valuation g^A: $\{g \mid \text{ for each } p \in \mathcal{P}^A,\ g \models p \text{ iff } g^A(p) = \mathit{true}\}$. We denote such set as a *interpretation* of the abstract valuation, written as $\mathcal{I}(g^A)$.

According to the definition[3], the transition $(\bar{s}, \phi, a, \Omega, \bar{s}')$ of Σ_p is fireable in a (concrete) configuration $\gamma = \langle \bar{s}, g \rangle$, if the valuation satisfies the transition guard. The resulting valuation is defined as $\mathit{upd}(g, \Omega)$. Given an *abstract configuration* $\gamma^A = \langle \bar{s}, g^A \rangle$, the transition is fireable in γ^A, if $g^A \models \phi$. Analogously, the result of the transition is some valuation $\mathit{upd}^A(g^A, \Omega)$, such that there exists $g \in \mathcal{I}(g^A)$, for which $\mathit{upd}(\gamma, \Omega) \in \mathcal{I}(\mathit{upd}^A(\gamma^A, \Omega))$. The run of the abstract model as defined in the same way. It is easy to see that the abstract model is finite.

Symbolic Analysis Techniques. As we discussed above, the analysis of the correspondence between the choreography and the implementation requires that the following three properties are satisfied: the implementation is complete (i.e., all the messages are received), bounded, and the asynchronous conformance relation is satisfied. The algorithm that allows for the boundedness and completeness analysis of the above implementation model is presented in [4]. The verification of the asynchronous conformance relation between Σ_p and Σ_c models may be done symbolically, based on the abstractions for these models. The symbolic algorithm, adopted for the conformance checking analysis is presented in [10]. In particular, it is shown how the equivalence relation may be represented symbolically, and verified using BDD-based model checking algorithm.

Symbolic model checking algorithms may be used also for the verification of the alignment rules. We exploit the Computational Tree Logic (CTL, [11]) for this purposes. Given an alignment rule, a corresponding CTL formula ϕ_R is constructed, which holds when the implementation satisfies the rule.

More formally, let $IR = \langle (r_s, r_r, \mu, \bar{v}_s, \bar{v}_r), e_{IR} \rangle$ be an interaction alignment rule. Let $\phi_{\overrightarrow{\mu}}$ (respectively, $\phi_{\overleftarrow{\mu}}$) be an expression, which is true if and only if the message μ is emitted (resp. received). A CTL formula ϕ_{IR} is defined as follows:

$$\phi_{IR} = \mathrm{AG}(\phi_{\overrightarrow{\mu}} \Rightarrow ((\mathrm{AF}(\phi_{\overleftarrow{\mu}} \wedge \phi_{IR})) \ \wedge \ \mathrm{A}(\neg \phi_{\overrightarrow{\mu}} \mathrm{U} \phi_{\overleftarrow{\mu}}))) \ .$$

In other words, from each state, where the aligned interaction is started, (i) the state, where the interaction is complete, should be always reachable, (ii) the information alignment condition should be satisfied, and (iii) there should not be any intermediate emissions.

Analogously, let $CR = \{\phi_{\bar{s}}\}$ be a coordination alignment rule. The corresponding CTL formula is defined as follows:

$$\phi_{CR} = (\mathrm{AF} \bigvee_{\bar{s}} \mathrm{AG}\phi_{\bar{s}}) \ \wedge \ (\bigwedge_{\bar{s}} \mathrm{EF}\ \mathrm{AG}\phi_{\bar{s}}) \ .$$

The formula states that some of the allowed termination states is always reachable, and each of them may be reached by some execution of the composition.

[3] The abstraction of CTS may be defined analogously.

5 Conclusions and Related Work

In this paper we presented a formal framework for the verification of the conformance between the choreography specification and the composition of service implementations. The formalism allows for modelling the data- and control-flow of the Web service compositions, defined as a global protocol and as a set of interacting local services. The key feature of the framework is the asynchronous message exchange, where the messages may be reordered and stored in unbounded queues. We exploit this feature for the definition of asynchronous choreography conformance, thus allowing for more accurate analysis of a wider class of compositions. We also formalize advanced declarative synchronization requirements exploited by WS-CDL, such as the interaction alignment and the coordination alignment rules. Finally, we presented symbolic reasoning techniques for model checking choreography specifications against the implementing compositions.

The work close to ours is presented in [7]. The choreography and the orchestration languages are formalized, and the notion of conformance between the specifications is presented. Here we extends the model of [7] in several directions. First, our approach allows for representation and management of data. Second, we adopt asynchronous communication model, while the interactions are defined in [7] as synchronous. Third, we also aim at addressing the information alignment problem, thus covering more essential choreography properties.

The problem of verification of the global protocol specification against the implementing composition is also discussed in [12,13,14]. In [12] the notion of conformance is defined by means of automata and is restricted only to compositions of two services. In [13] the choreography specifications are used to represent the service obligations rules, and then are verified against the implementations defined as compositions of interacting BPEL processes. Again, the analysis does not consider the data-flow of the composition, and relies on the synchronous communication model, which is not realistic for a wide class of composition scenarios. The work of [14] concentrates on checking that the choreography specification is respected by the implementing services at run time. The formalisation is given in terms of Petri Nets.

The formalization of the Web service choreography models are also presented in [15,16,17]. In particular, in [17] the global and the local (end-point) calculi are introduces to describe the choreography and the behavior of compositions of local implementations. The work discusses the relation between the two paradigms, and presents the potential problems related to the asynchronous exchanges and message reorderings. The problem of synchronous versus asynchronous interactions in global models is also discussed in [18], where the notion of the protocol synchronizability is presented together with the sufficient conditions. The results of [4] extend this approach and provide a way to determine an appropriate level of asynchronism and a suitable communication model for the given composition.

References

1. Andrews, T., Curbera, F., Dolakia, H., Goland, J., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weeravarana, S.: Business Process Execution Language for Web Services (version 1.1) (2003)
2. OMG: Business Process Modeling Language (BPML). (2005) [http://www.bpmi.org].
3. W3C: (Web Services Choreography Description Language Version 1.0. W3C Candidate Recommendation 9 November 2005) [http://www.w3.org/TR/ws-cdl-10/].
4. Kazhamiakin, R., Pistore, M., Santuari, L.: Analysis of Communication Models in Web Service Compositions. In: Proc. WWW'06. (2006)
5. Dijkman, R.M., Dumas, M.: Service-Oriented Design: A Multi-Viewpoint Approach. Int. J. Cooperative Inf. Syst. **13**(4) (2004) 337–368
6. Guerin, F., Pitt, J.: Verification and compliance testing. In: Communication in Multiagent Systems. (2003) 98–112
7. Busi, N., Gorrieri, R., Guidi, C., Lucchi, R., Zavattaro, G.: Choreography and Orchestration: A Synergic Approach for System Design. In: Proc. ICSOC'05. (2005)
8. Graf, S., Saidi, H.: Construction of abstract state graph with PVS. In: Proc. CAV'97. (1997)
9. Chaki, S., Clarke, E., Groce, A., Ouaknine, J., Strichman, O., Yorav, K.: Efficient verification of sequential and concurrent C programs. In: Proc. FMSD'04. (2004)
10. Burch, J., Clarke, E., McMillan, K., Dill, D., Hwang, L.: Symbolic Model Checking: 10^{20} States and Beyond. In: Proceedings of the Fifth Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society Press (1990)
11. Clarke, E., Emerson, E., Sistla, A.: Automatic Verification of Finite-state Concurrent Systems Using Temporal Logic Specifications. ACM Transactions on Programming Languages and Systems **8**(2) (1986)
12. Baldoni, M., Baroglio, C., Martelli, A., Patti, V., Schifanella, C.: Verifying the Conformance of Web Services to Global Interaction Protocols: a First Step. In: Proc. EPEW/WS-FM. (2005)
13. Foster, H., Uchitel, S., Magee, J., Kramer, J.: Model-Based Analysis of Obligations in Web Service Choreography. In: Proc. AICT-ICIW'06. (2006)
14. van der Aalst, W.M., Dumas, M., Ouyang, C., Rozinat, A., Verbeek, H.: Choreography Conformance Checking: An Approach based on BPEL and Petri Nets. Technical report, BPM Center Report BPM-05-25 (2005)
15. Brogi, A., Canal, C., Pimentel, E., Vallecillo, A.: Formalizing Web Services Choreographies. In: Proc. WS-FM'04. (2004) ENTCS.
16. Bravetti, M., Guidi, C., Lucchi, R., Zavattaro, G.: Supporting e-commerce systems formalization with choreography languages. In: Proc. SAC '05. (2005)
17. Carbone, M., Honda, K., Yoshida, N.: A theoretical basis of communication-centred concurrent programming (2005) [http://lists.w3.org/Archives/Public/public-ws-chor/2005Nov/att-0015/part1_Nov25.pdf].
18. Fu, X., Bultan, T., Su, J.: Synchronizability of Conversations among Web Services. IEEE Transactions on Software Engineering **31**(12) (2005) 1042–1055