The *must* preorder revisited
– an algebraic theory for web services contracts –

Cosimo Laneve[1] and Luca Padovani[2]

[1] Department of Computer Science, University of Bologna
[2] Information Science and Technology Institute, University of Urbino

Abstract. We define a language for Web services contracts as a parallel-free fragment of CCS and we study a natural notion of compliance between clients and services in terms of their corresponding contracts. The induced contract preorder turns out to be valuable in searching and querying registries of Web services, it shows interesting connections with the must preorder, and it exhibits good precongruence properties when choreographies of Web services are considered. Our contract language may be used as a foundation of Web services technologies, such as WSDL and WSCL.

1 Introduction

Web services contracts are coarse-grained abstract descriptions to be used in workflows and business processes implemented in composite applications or portals. These descriptions are intended to be replaced by fine-grained implementations that provide complex Web services.

Current technologies for describing contracts specify the format of the exchanged messages – the *schema* –, the locations where the interactions are going to occur – the *interface* –, the transfer mechanism to be used (i.e. SOAP-RPC, or others), and the pattern of conversation of the service – the *behavior*. For example, the Web Service Description Language (WSDL) [10, 9, 8] defines simple behaviors: one-way (asynchronous) and request/response (synchronous) ones. The Web Service Conversation Language (WSCL) [2] extends WSDL behaviors by allowing the description of arbitrary, possibly cyclic sequences of exchanged messages between communicating parties. (Other languages, such as the Abstract business processes in the Web Service Business Execution Language (WS-BPEL) [1], provide even more detailed descriptions because they also define the subprocess structure, fault handlers, etc. We think that such descriptions are too much concrete to be used as contracts.)

Documents describing WSDL and WSCL contracts can be published in registries [3, 11] so that Web services can be *searched* and *queried*. These two basic operations assume the existence of some notion of contract equivalence. The lack of a formal characterization of contracts only permits excessively demanding notions of equivalence such as syntactical equality. In fact, it makes perfect sense to further relax the equivalence into a *subcontract preorder* (denoted by $\preceq$ in this paper), so that Web services exposing "larger" contracts can be *safely* returned as results of queries for Web services with "smaller" contracts. The purpose of this paper is to define precisely what "larger" and "smaller" mean, as well as to define which safety property we wish to preserve when substituting a service exposing a contract with a service exposing a larger contract.

In our formalization, contracts are pairs $I : \sigma$, where I is the *interface*, i.e. the set of names used by the service for responses and requests, and σ is the *behavior*, i.e. the conversation pattern (it is intended that names in σ are included into I). Our investigation abstracts away from the syntactical details of schemas as well as from those aspects that are too oriented to the actual implementations, such as the definition of transmission protocols; all of such aspects may be easily integrated on top of the formalism. We do not commit to a particular interpretation of names either: they can represent different typed channels on which interaction occurs or different types of messages. Behaviors are sequences of request or response actions at specific names, possibly combined by means of two choice operators. The *external choice* "+" means that it is the interacting part that decides which one of alternative behaviors to carry on; the *internal choice* "$\oplus$" means that the it is the part exposing the contract that decides how to proceed. Recursive behaviors are also admitted in our contract language. As a matter of facts, contracts are CCS (without τ's) processes that do not manifest internal moves and the parallel structure [17, 19].

To equip our contracts with a subcontract preorder $\preceq$, we commit to a testing approach. We define client satisfaction as the ability of the client to complete successfully the interaction with the service; "successfully" meaning that the client never gets stuck (this notion is purposefully asymmetric as client's satisfaction is our main concern). The preorder arises by comparing the sets of clients satisfied by services. The properties enjoyed by the $\preceq$ preorder are particularly relevant in the context of Web services. For example, a service exposing the contract $\{a, b, c\} : \overline{a}.c \oplus \overline{b}.c$ is one that sends a message on a or on b – the choice is left to the service – and then waits for a response on c. A client compatible with such service must be prepared to accept messages from both a and b. Hence, such a client will also be compatible with services exposing either $\{a, b, c\} : \overline{a}.c$ or $\{a, b, c\} : \overline{b}.c$, which behave more deterministically than the original service, or even $\{a, b, c\} : \overline{a}.c+\overline{b}.c$, which leaves the choice to the client. This is expressed as $\{a, b, c\} : \overline{a}.c \oplus \overline{b}.c \preceq \{a, b, c\} : \overline{a}.c$ and $\{a, b, c\} : \overline{a}.c \oplus \overline{b}.c \preceq \{a, b, c\} : \overline{a}.c + \overline{b}.c$. Notice that $\{a, b, c\} : \overline{a}.c \preceq \{a, b, c\} : \overline{a}.c + \overline{b}.c$ should not hold. For example a client with contract $\{a, b, c\} : a.\overline{c} + b.a.\overline{c}$ successfully completes when interacting with $\{a, b, c\} : \overline{a}.c$, but it may fail when interacting with $\{a, b, c\} : \overline{a}.c + \overline{b}.c$. However, a client interacting with $\{a, c\} : \overline{a}.c$, that is never interacting on b, cannot fail with $\{a, b, c\} : \overline{a}.c + \overline{b}.c$. Namely, if a service is extended *in width* with new functionalities, the old clients will still complete successfully with the new service. Similarly, extensions *in depth* of a service should allow old clients to complete: $\{a, c\} : \overline{a}.c \preceq \{a, b, c\} : \overline{a}.c.\overline{b}.c$. To the best of our knowledge, there is no process semantics corresponding to $\preceq$ in the literature. However, the restriction of $\preceq$ to contracts with the same interface is a well-known semantics: the *must-testing preorder* [18, 19, 14].

Our investigation about a semantics for contracts also addresses the problem of determining, given a client exposing a certain behavior, the smallest (according to $\preceq$) service contract that satisfies the client – the *principal dual contract.* This contract, acting like a *principal type* in type systems, guarantees that a query to a Web services registry is answered with the largest possible set of compatible services in the registry's databases. Technically, computing the dual contract is not trivial because it cannot be reduced to a swapping of requests and responses, and external and internal choices. For example, applying this swapping to the contracts $\{a, b, c\} : a.\overline{b}+a.\overline{c}$ and $\{a, b, c\} : a.(\overline{b}\oplus\overline{c})$, which happen to be equivalent according to $\preceq$, would produce the contracts $\{a, b, c\} : \overline{a}.b \oplus \overline{a}.c$ and $\{a, b, c\} : \overline{a}.(b + c)$, respectively, but only the latter one does actually satisfy the clients exposing one of the two original contracts. As another example, the dual contract of $\{a\} : a$ cannot simply be $\{a\} : \overline{a}$, because a service with contract $\{a\} : \overline{a} \oplus (\overline{a} + a)$ satisfies the original contract and yet $\{a\} : \overline{a} \not\preceq \{a\} : \overline{a} \oplus (\overline{a} + a)$.

We also study the application of our theory of contracts to choreographies of Web services [15]. A choreography is an abstract specification of several services that run in parallel and communicate with each other by means of private names. (Actually, the behavior of each service may be synthesized out of a global description [5, 4]). We show that $\preceq$ is robust enough so that, replacing an abstract specification with an implementation that is related to the specification by means of $\preceq$, the observable features of the choreography as a whole are preserved.

Related work. This research was inspired by "CCS without τ's" [19] and by Hennessy's model of acceptance trees [13, 14]. Our contracts are an alternative representation of acceptance trees. While the use of formal models to describe communication protocols is not new (see for instance the exchange patterns in SSDL [20], which are based on CSP and the π-calculus), to the best of our knowledge the subcontract relation $\preceq$ is original. Although it resembles the must preorder (and it reduces to the must preorder when the interfaces are large enough), $\preceq$ arises from a notion of compliance that significantly differs from the notion of "passing a test" in the testing framework [18] and that more realistically describes well-behaved clients of Web services. The *width* extension property enjoyed by $\preceq$ is closely related to subtyping in object-oriented programming languages. The works that are more closely related to ours are by Carpineti *et al.* [6], by Castagna *et al.* [7] and the ones on *session types*, especially [12] by Gay and Hole. In [6] the subcontract relation (over finite contracts) exhibits all of the desirable properties illustrated in the introduction. Unfortunately, such relation turns out to be non transitive as the preorder arises as a syntactic notion. Transitivity, while not being strictly necessary as far as querying and searching are concerned (in fact, the coinductive notion of compliance defined in [6] would suffice) has two main advantages: on the practical side it allows databases of Web services contracts to be organized in accordance with the subcontract relation, so as to reduce the run time spent for executing queries. The transitive closure of a query can be precomputed when a new service is registered.

On the theoretical side, it is a fundamental prerequisite when contracts are considered as (behavioral) types for typing processes: the subcontract relation is just the subsumption rule. The transitivity problem has been also addressed in [7] in a more general way. However, the authors of [7] must introduce a rather powerful construct (the *filter*) which prevents potentially dangerous interactions. Roughly speaking, a filter actively mediates the client/service interaction at run time, by dynamically changing the interface of the service as it is seen from the client. With respect to [12] our contract language is much simpler and it can express more general forms of interaction. While the language defined in [12] supports first-class sessions and name passing, it is purposefully tailored so that the transitivity problems mentioned above are directly avoided at the language level. This restricts the subcontract relation in such a way that internal and external choices can never be related (hence, $\{a,b\} : a \oplus b \preceq \{a,b\} : a + b$ does *not* hold).

Structure of the paper. In Section 2 we formally define our language for contracts, the corresponding transition relation and the compliance of a client with a service. In Section 3 we study the subcontract preorder and its relationship with the must preorder. In Section 4 we analyze the problem of determining the principal dual contract. In Section 5 we discuss the application of the subcontract preorder to choreographies. Section 6 discusses the expressivity of our contracts by showing the encoding of a WSCL conversation into our language, it draws our conclusions, and hints at future work.

Extended proofs of the relevant results are provided for referee convenience only. They will reduced in case of acceptance to fit the page limit.

2 The contract language

The syntax of contracts uses an infinite set of *names* $\mathcal{N}$ ranged over by a, b, c, ..., and a disjoint set of *co-names* $\overline{\mathcal{N}}$ ranged over by $\overline{a}$, $\overline{b}$, $\overline{c}$, ... We use the term *action* for referring to names and co-names without distinction. We let $\overline{\overline{a}} = a$, we use $\alpha, \beta, \dots$ to range over $\mathcal{N} \cup \overline{\mathcal{N}}$, and we use $\varphi, \psi, \dots$ to range over $(\mathcal{N} \cup \overline{\mathcal{N}})^*$. *Contracts* are pairs $I : \sigma$ where I, called *interface*, is a finite subset of $\mathcal{N}$ representing the set of names on which interaction occurs, whereas σ, called *behavior*, is defined by the following grammar:

$$
\begin{array}{rll}
\sigma ::= & & \\
| & \mathbf{0} & \text{(null)} \\
| & \alpha.\sigma & \text{(prefix)} \\
| & x & \text{(variable)} \\
| & \sigma + \sigma & \text{(external choice)} \\
| & \sigma \oplus \sigma & \text{(internal choice)} \\
| & \mathsf{rec}\ x.\sigma & \text{(recursion)}
\end{array}
$$

The behavior $\mathbf{0}$ defines the empty conversation; the behavior $a.\sigma$ defines a conversation protocol whose initial activity is to accept a request on a and continuing as σ; the behavior $\overline{a}.\sigma$ defines a conversation protocol whose initial activity is to send a response to a and continuing as σ. Behaviors $\sigma + \sigma'$ and $\sigma \oplus \sigma'$ define conversation protocols that follow either the conversation σ or σ'; in $\sigma + \sigma'$ the choice is left to the remote party, in $\sigma \oplus \sigma'$ the choice is made locally. For example, $\texttt{Login}.(\overline{\texttt{Continue}}+\overline{\texttt{End}})$ describes the conversation protocol of a service that is ready to accept `Login`s and will `Continue` or `End` the conversation according to client's request. This contract is different from $\texttt{Login}.(\overline{\texttt{Continue}}\oplus\overline{\texttt{End}})$ where the decision whether to continue or to end is taken by the service. The behavior $\mathsf{rec}\ x.\sigma$ defines a possibly recursive conversation protocol whose recurrent pattern is σ. A (free) occurrence of the variable x in σ stands for the whole $\mathsf{rec}\ x.\sigma$. In the following we write $\mathbf{\Omega}$ for $\mathsf{rec}\ x.x$.

Let $\texttt{names}(\sigma)$ be the set of names a such that either a or $\overline{a}$ occur in σ. We always assume that $\texttt{names}(\sigma) \subseteq I$ holds for every $I : \sigma$. With an abuse of notation we write $\texttt{names}(\varphi)$ for the set of names occurring in φ.

Behaviors retain a *transition relation* that is inductively defined by the rules

$$\alpha.\sigma \xrightarrow{\alpha} \sigma \qquad \sigma_1 \oplus \sigma_2 \longrightarrow \sigma_1 \qquad \frac{\sigma_1 \xrightarrow{\alpha} \sigma_1'}{\sigma_1 + \sigma_2 \xrightarrow{\alpha} \sigma_1'} \qquad \frac{\sigma_1 \longrightarrow \sigma_1'}{\sigma_1 + \sigma_2 \longrightarrow \sigma_1' + \sigma_2}$$

$$\mathsf{rec}\ x.\sigma \longrightarrow \sigma\{^{\mathsf{rec}\ x.\sigma}/_{x}\}$$

plus the symmetric rules for $\oplus$ and $+$. This operational semantics is exactly the same as CCS without τ's [19]. In particular, the rules for $\oplus$ say that the behavior $\sigma_1 \oplus \sigma_2$ may exhibit σ_1 or σ_2 through an internal, unlabeled transition. The behavior $\sigma_1 + \sigma_2$ may exhibit σ_1 or σ_2 only after performing a visible, labeled transition of σ_1 or σ_2, respectively; internal transitions do not modify the external choice. A recursive behavior $\mathsf{rec}\ x.\sigma$ unfolds to $\sigma\{^{\mathsf{rec}\ x.\sigma}/_{x}\}$ with an internal transition. We write $\Longrightarrow$ for the reflexive and transitive closure of $\longrightarrow$; $\sigma \overset{\alpha}{\Longrightarrow} \sigma'$ for $\sigma \Longrightarrow\xrightarrow{\alpha}\Longrightarrow \sigma'$; $\sigma \overset{\alpha_1\cdots\alpha_n}{\Longrightarrow} \sigma'$ if $\sigma \overset{\alpha_1}{\Longrightarrow} \cdots \overset{\alpha_n}{\Longrightarrow} \sigma'$; $\sigma \overset{\varphi}{\Longrightarrow}$ if there exists σ' such that $\sigma \overset{\varphi}{\Longrightarrow} \sigma'$. We write $\sigma\uparrow$ if σ has an infinite internal computation $\sigma = \sigma_0 \longrightarrow \sigma_1 \longrightarrow \sigma_2 \longrightarrow \cdots$ and $\sigma\downarrow$ if not $\sigma\uparrow$. We write $\sigma \downarrow \varphi$ if $\sigma \overset{\varphi}{\Longrightarrow} \sigma'$ implies $\sigma'\downarrow$. Finally, we write $\sigma \uparrow \varphi$ if $\sigma \overset{\varphi}{\Longrightarrow} \sigma'$ and $\sigma'\uparrow$ for some σ'. For example $\mathbf{\Omega}\uparrow$, $\mathsf{rec}\ x.a + x\uparrow$, and $\mathsf{rec}\ x.(a.x + b.x) \downarrow \varphi$ for every $\varphi \in \{a, b\}^*$. Let $\texttt{init}(\sigma) \overset{\text{def}}{=} \{\alpha \mid \sigma \overset{\alpha}{\Longrightarrow}\}$.

A basic use of contracts is to verify whether a client protocol is consistent with a service protocol. This consistency, called behavioral compliance in the following, requires two preliminary definitions: that of communicating behaviors and that of matching:

– The notion of *communicating behaviors* extends the transition relation $\longrightarrow$ to pairs of behaviors as follows:

$$\frac{\rho \longrightarrow \rho'}{\rho \mid \sigma \longrightarrow \rho' \mid \sigma} \qquad \frac{\sigma \longrightarrow \sigma'}{\rho \mid \sigma \longrightarrow \rho \mid \sigma'} \qquad \frac{\rho \xrightarrow{\alpha} \rho' \quad \sigma \xrightarrow{\overline{\alpha}} \sigma'}{\rho \mid \sigma \longrightarrow \rho' \mid \sigma'}$$

– The notion of *matching* is modeled using a special name $\mathtt{e}$ for denoting the successful termination of a party ("$\mathtt{e}$" stands for $\mathtt{end}$). By "success" we mean the ability to always reach a state in which $\mathtt{e}$ can be emitted.

Definition 1 (Behavioral compliance). *Let* $\mathtt{e} \notin \mathtt{names}(\sigma)$*. The (client) behavior* ρ *is* compliant with *the (service) behavior* σ*, written* $\rho \dashv \sigma$*, if* $\rho \mid \sigma \Longrightarrow \rho' \mid \sigma'$ *implies*

1. *if* $\rho' \mid \sigma' \longarrownot\longrightarrow$*, then* $\{\mathtt{e}\} \subseteq \mathtt{init}(\rho')$*;*
2. *if* $\sigma' \uparrow$*, then* $\{\mathtt{e}\} = \mathtt{init}(\rho')$*.*

According to the notion of behavioral compliance, if the client-service conversation terminates, then the client is in a successful state (it will emit an $\mathtt{e}$-name). For example, $a.\mathtt{e} + b.\mathtt{e} \dashv \overline{a} \oplus \overline{b}$ and $a.\mathtt{e} \oplus b.\mathtt{e} \dashv \overline{a} + \overline{b}$ but $a.\mathtt{e} \oplus b.\mathtt{e} \not\dashv \overline{a} \oplus \overline{b}$ because of the computation $a.\mathtt{e} \oplus b.\mathtt{e} \mid \overline{a} \oplus \overline{b} \Longrightarrow a.\mathtt{e} \mid \overline{b} \longarrownot\longrightarrow$ where the client waits for an interaction on a in vain. Similarly, the client must reach a successful state if the conversation does not terminate but the divergence is due to the service. In this case, however, the compliance relation takes into account the subtleties that divergence imply. In particular, the client is constrained to autonomously reach a state where the only possible action is $\mathtt{e}$. The practical justification of such a notion of compliance derives from the fact that connection-oriented communication protocols (such as TCP/IP) typically provide for an explicit end-of-connection signal. So for example $\mathtt{e} + \overline{a}.\mathtt{e} \dashv \mathbf{0}$ because the client can detect the fact that the service has dropped every connection. A divergent service might keep its connections open, so that a client would have no way to distinguish a service that is taking a long time to offer the next interaction from a service that is perpetually performing internal computations. Hence $\mathtt{e} + \overline{a}.\mathtt{e} \not\dashv \mathbf{\Omega}$ because the service silently diverges and the client might rely on the end-of-connection signal to be able to terminate immediately.

Our notion of behavioral compliance enjoys the following "subject reduction" property, stating that given two compliant behaviors ρ and σ, every residual behaviors ρ' and σ' are also compliant.

Proposition 1. *If* $\rho \dashv \sigma$ *and* $\rho \mid \sigma \longrightarrow \rho' \mid \sigma'$*, then* $\rho' \dashv \sigma'$*.*

3 The subcontract relation

The behavioral compliance is actually a basic test for investigating services. Following De Nicola and Hennessy's approach to process semantics [18], this test induces a preorder on services on the basis of the set of clients that comply with a given service.

Definition 2 (Contract semantics). *Let* $[\![I : \sigma]\!] \stackrel{\text{def}}{=} \{ J : \rho \mid \; J \subseteq I \text{ and } \rho \dashv \sigma \}$*. A contract* $I : \sigma$ *is a* subcontract *of* $I' : \sigma'$*, written* $I : \sigma \preceq I' : \sigma'$*, if and only if* $[\![I : \sigma]\!] \subseteq [\![I' : \sigma']\!]$*. Let* $\simeq$ *be* $\preceq \cap \succeq$*.*

For example, $[\![\emptyset : \mathbf{\Omega}]\!] = \{\emptyset : \mathsf{e}, \emptyset : \mathsf{rec}\ x.\mathsf{e} + x, \ldots\}$ and $[\![\{a,b\} : a \oplus b]\!] = \{\emptyset : \mathsf{e}, \emptyset : \mathsf{rec}\ x.\mathsf{e}+x, \{a\} : \mathsf{e}, \{b\} : \mathsf{e}, \{a,b\} : \mathsf{e}, \{a,b\} : \overline{a}.\mathsf{e}+\overline{b}.\mathsf{e}, \ldots\}$. It turns out that clients in $[\![\emptyset : \mathbf{\Omega}]\!]$ comply with every other service, hence the service $\emptyset : \mathbf{\Omega}$ is the smallest one. As usual it is easier to figure out inequalities: $\{a,b\} : a \not\preceq \{a,b\} : a.b$ because $\overline{a}.(\mathsf{e}+\overline{b}) \dashv a$ but $\overline{a}.(\mathsf{e}+\overline{b}) \not\dashv a.b$; $\{a,b\} : a \not\preceq \{a,b\} : a+b$ because $\mathsf{e}+\overline{b} \dashv a$ but $\mathsf{e} + \overline{b} \not\dashv a + b$.

The next proposition states the desirable properties listed in Section 1 in a formal way. Notice that for most of them it would be possible to weaken the premises, but we prefer this presentation as it highlights more clearly the relevant features of $\preceq$. The proofs are omitted as they are easy (see the alternative characterization of $\preceq$ in Definition 3).

Proposition 2. *Let $I : \sigma$ and $I' : \sigma'$ be contracts such that $I \cap I' = \emptyset$. Then:*

1. *if $J : \rho \preceq I \cup I' : \sigma$ and $J : \rho \preceq I \cup I' : \sigma'$, then $J : \rho \preceq I \cup I' : \sigma \oplus \sigma'$;*
2. *if $\sigma\downarrow$ and $\sigma'\downarrow$, then $I \cup I' : \sigma \preceq I \cup I' : \sigma + \sigma'$;*
3. *if $\sigma'\downarrow$, then $I \cup I' : \sigma\{\mathbf{0}/_x\} \preceq I \cup I' : \sigma\{\sigma'/_x\}$;*
4. *$I : \sigma \preceq I \cup I' : \sigma$.*

Item 1 states that $I \cup I' : \sigma \oplus \sigma'$ is the largest contract that satisfies the clients that are compliant with both $I \cup I' : \sigma$ and $I \cup I' : \sigma'$. Namely, $[\![I \cup I' : \sigma]\!] \cap [\![I \cup I' : \sigma']\!] = [\![I \cup I' : \sigma \oplus \sigma']\!]$. Item 2 gives sufficient conditions for width extensions of Web services: a Web service may be upgraded to offer additional functionalities without affecting the set of clients it satisfies, so long as such new functionalities regard names that were not present in the original service. In fact, it suffices to require $\mathtt{init}(\sigma') \cap I = \emptyset$ to establish the result. Contrary to item 1, $I \cup I' : \sigma + \sigma'$ is *not* the smallest contract that satisfies the clients that are compliant with either $I \cup I' : \sigma$ or $I \cup I' : \sigma'$. For example, $\{a,b\} : a.\mathsf{e} \oplus b.\mathsf{e} \in [\![\{a,b\} : \overline{a} + \overline{b}]\!]$ but $\{a,b\} : a.\mathsf{e} \oplus b.\mathsf{e} \notin [\![\{a,b\} : \overline{a}]\!]$ and $\{a,b\} : a.\mathsf{e} \oplus b.\mathsf{e} \notin [\![\{a,b\} : \overline{b}]\!]$, hence $[\![I \cup I' : \sigma]\!] \cup [\![I \cup I' : \sigma']\!] \subsetneq [\![I \cup I' : \sigma + \sigma']\!]$. Item 3 states a similar result, but for depth extensions, that is the ability to extend the conversation offered by a service, provided that the additional conversation occurs on names that were not present in the original service. The premises can be weakened as for item 2. In fact, item 2 can be seen as a special case of item 3, if we consider the contract $I \cup I' : \sigma + x$. Item 4 shows that merely increasing the names that a Web service can interact on does not affect the clients it satisfies.

As usual with testing semantics, it is hard to establish a relationship between two contracts because the sets $[\![I : \sigma]\!]$ are infinite. A direct definition of the preorder is therefore preferred.

Definition 3. *Let $\sigma \Downarrow \mathrm{R}$ if and only if $\sigma \Longrightarrow \sigma'$ and $\mathrm{R} = \mathtt{init}(\sigma')$. A* coinductive subcontract *is a relation $\mathcal{R}$ such that if $(I : \rho, J : \sigma) \in \mathcal{R}$, then $I \subseteq J$ and whenever $\rho\downarrow$ then*

1. *$\sigma\downarrow$, and*
2. *$\sigma \Downarrow \mathrm{R}$ implies $\rho \Downarrow \mathrm{R}'$ and $\mathrm{R}' \subseteq \mathrm{R}$, and*
3. *$\alpha \in I$ and $\sigma \overset{\alpha}{\Longrightarrow} \sigma'$ imply $\rho \overset{\alpha}{\Longrightarrow} \rho_1, \ldots, \rho \overset{\alpha}{\Longrightarrow} \rho_n$ and $(I : \bigoplus_{1 \le i \le n} \rho_i, J : \sigma_2') \in \mathcal{R}$.*

By this definition, a contract $I : \rho$ such that $\rho\uparrow$ is the smallest one with interface I. When $\rho\downarrow$, condition 1 constrains the larger contract $J : \sigma$ to converge as well, since clients might rely on the convergence of ρ to complete successfully. Condition 2 states that $J : \sigma$ must exhibit a more deterministic behavior: the lesser the number of ready sets is, the more deterministic the contract is. Furthermore, $J : \sigma$ should expose *at least* the same capabilities as the smaller one ($\mathrm{R}' \subseteq \mathrm{R}$). Condition 3 is perhaps the most subtle one, as it deals with all the possible derivatives of the smaller contract. The point is that $\{a, b, c\} : a.b + a.c \simeq \{a, b, c\} : a.(b \oplus c)$ since, after interacting on a, a client of the service on the left side of $\simeq$ is not aware of which state the service is in (it can be either b or c). Hence, we have to consider all of the possible derivatives after a, thus reducing to verifying $(\{a, b, c\} : (b + c) \oplus b \oplus c, \{a, b, c\} : b \oplus c) \in \mathcal{R}$ which trivially holds.

The largest coinductive subcontract and $\preceq$ do coincide (Theorem 1). The following lemma is preparatory to this result.

Lemma 1. *Let $(I : \rho, J : \sigma) \in \mathcal{R}$, with $\mathcal{R}$ a coinductive subcontract, and $\sigma \overset{\alpha_1 \cdots \alpha_n}{\Longrightarrow} \sigma'$, with $\mathtt{names}(\alpha_1.\cdots.\alpha_n) \subseteq I$. If, for every i, $\rho \downarrow \alpha_1 \cdots \alpha_i$, then there exist $\rho \overset{\alpha_1 \cdots \alpha_n}{\Longrightarrow} \rho_1, \dots, \rho \overset{\alpha_1 \cdots \alpha_n}{\Longrightarrow} \rho_m$, $m \geq 1$, such that $(I : \bigoplus_{1 \leq j \leq m} \rho_j, J : \sigma') \in \mathcal{R}$.*

In the proofs that follow we often mention the "unzipping" of a derivation $\rho \mid \sigma \overset{\varphi}{\Longrightarrow} \rho' \mid \sigma'$, which results into two sequences ψ and ψ' of actions such that $\rho \overset{\psi}{\Longrightarrow} \rho'$ and $\sigma \overset{\psi'}{\Longrightarrow} \sigma'$. Intuitively, ψ and ψ' contain (a subset of) the actions in φ interspersed with the actions on which (the derivatives of) ρ and σ have synchronized. When φ is empty, then $\overline{\psi} = \psi'$, where $\overline{\psi}$ is the co-sequence obtained from ψ by swapping names and co-names. By "zipping" we mean the inverse process whereby two or more derivations such as $\rho \overset{\psi}{\Longrightarrow} \rho'$ and $\sigma \overset{\psi'}{\Longrightarrow} \sigma'$ are combined to produce $\rho \mid \sigma \overset{\varphi}{\Longrightarrow} \rho' \mid \sigma'$. See [14] for a more detailed discussion.

Theorem 1. *$\preceq$ is the largest coinductive subcontract relation.*

Proof. We prove that $I_1 : \sigma_1 \preceq I_2 : \sigma_2$ if and only if there exists a coinductive subcontract $\mathcal{R}$ such that $(I_1 : \sigma_1, I_2 : \sigma_2) \in \mathcal{R}$.

Let $\mathcal{R}$ be a coinductive subcontract and $(I_1 : \sigma_1, I_2 : \sigma_2) \in \mathcal{R}$. In order to demonstrate $I_1 : \sigma_1 \preceq I_2 : \sigma_2$, let $\rho \mid \sigma_2 \Longrightarrow \rho' \mid \sigma_2' \not\longrightarrow$. By unzipping this derivation we obtain $\rho \overset{\varphi}{\Longrightarrow} \rho'$ and $\sigma_2 \overset{\overline{\varphi}}{\Longrightarrow} \sigma_2'$, for some sequence φ of actions. At any intermediate step of $\sigma_2 \overset{\overline{\varphi}}{\Longrightarrow} \sigma_2'$, the behavior converges because $\overline{\mathtt{e}}$ actions cannot occur in σ_2 and, by definition of behavior compliance, if the (service) behavior diverges then the (client) behavior cannot propose actions other than $\mathtt{e}$.

By Lemma 1 there exist $\sigma_1 \overset{\overline{\varphi}}{\Longrightarrow} \tau_1, \dots, \sigma_1 \overset{\overline{\varphi}}{\Longrightarrow} \tau_n$, $n \geq 1$, such that $(I_1 : \bigoplus_{1 \leq i \leq n} \tau_i, I_2 : \sigma_2') \in \mathcal{R}$. Let $\sigma_2 \Downarrow \mathrm{R}'$ and let τ be such that $\tau_i \Longrightarrow \tau \not\longrightarrow$ and $\mathtt{init}(\tau) \subseteq \mathrm{R}'$. We know that such τ exists since $\tau_i\downarrow$ for every $1 \leq i \leq n$ and because of the coinductive subcontract we know that $\bigoplus_{1 \leq i \leq n} \tau_i \Downarrow \mathrm{R}$ and $\mathrm{R} \subseteq \mathrm{R}'$. Hence, by zipping the derivations from ρ and σ_1 we obtain

$$\rho \mid \sigma_1 \Longrightarrow \rho \mid \tau_i \Longrightarrow \rho \mid \tau \not\longrightarrow$$

and from $\rho \dashv \sigma_1$ we conclude $\{\mathtt{e}\} \subseteq \mathtt{init}(\rho)$. Next, consider a derivation $\rho \mid \sigma_2 \Longrightarrow \rho' \mid \sigma_2'$ and $\sigma_2'\!\uparrow$. By unzipping the derivation as before, we conclude that there exists a sequence φ of actions such that $\sigma_1 \overset{\overline{\varphi}}{\Longrightarrow} \sigma_1'$ and $\sigma_1'\!\uparrow$. Hence $\rho \mid \sigma_1 \Longrightarrow \rho' \mid \sigma_1'$ and from $\rho \dashv \sigma_1$ we conclude $\{\mathtt{e}\} = \mathtt{init}(\rho')$.

As regards the opposite direction, let

$$\mathcal{R} \overset{\text{def}}{=} \{(I_1 : \sigma_1, I_2 : \sigma_2) \mid I_1 : \sigma_1 \preceq I_2 : \sigma_2\} \ .$$

We prove that $\mathcal{R}$ is a coinductive subcontract. Let $(I_1 : \sigma_1, I_2 : \sigma_2) \in \mathcal{R}$. The client $I_1 : \mathtt{e}$ belongs to $[\![I_1 : \sigma_1]\!]$; therefore it belongs to $[\![I_2 : \sigma_2]\!]$ as well. Hence $I_1 \subseteq I_2$. If $\sigma_1\!\uparrow$ there is nothing to prove. So assume $\sigma_1\!\downarrow$. Proofs of conditions 1, 2, and 3 in the definition of coinductive subcontract are in order:

1. Since $\mathbf{\Omega} \dashv \sigma_1$ we have $\mathbf{\Omega} \dashv \sigma_2$. Hence $\sigma_2\!\downarrow$ for $\mathtt{init}(\mathbf{\Omega}) = \emptyset$.
2. Let $\mathrm{R}_1, \dots, \mathrm{R}_n$ be the ready sets of σ_1 and assume by contradiction that there exists R' such that $\sigma_2 \Downarrow \mathrm{R}'$ and for every $1 \le i \le n$ there exists $\alpha_i \in \mathrm{R}_i$ and $\alpha_i \notin \mathrm{R}'$. By definition of ready set we have $\sigma_2 \Longrightarrow \sigma_2' \not\longrightarrow$ and $\mathtt{init}(\sigma_2') \subseteq \mathrm{R}'$. Consider $\rho \overset{\text{def}}{=} \sum_{1 \le i \le n} \overline{\alpha_i}.\mathtt{e}$. Then, $\rho \dashv \sigma_1$ but $\rho \not\dashv \sigma_2$ because $\rho \mid \sigma_2 \Longrightarrow \rho \mid \sigma_2' \not\longrightarrow$ and $\mathtt{e} \notin \mathtt{init}(\rho)$, which is absurd.
3. Let $\sigma_2 \overset{\alpha}{\Longrightarrow} \sigma_2'$ and $\alpha \in I_1$. It must be the case that $\sigma_1 \overset{\alpha}{\Longrightarrow}$, otherwise $\mathtt{e} + \overline{\alpha} \dashv \sigma_1$ but $\mathtt{e} + \overline{\alpha} \not\dashv \sigma_2$. Consider the derivatives $\rho_1, \dots, \rho_n$ such that $\sigma_1 \Longrightarrow\xrightarrow{\alpha} \rho_i$. Since $\sigma_1\!\downarrow$ there is a finite number of such derivatives. We must prove that $(I_1 : \bigoplus_{1 \le i \le n} \rho_i, I_2 : \sigma_2') \in \mathcal{R}$, namely that $I_1 : \bigoplus_{1 \le i \le n} \rho_i \preceq I_2 : \sigma_2'$. Consider ρ' such that $\mathtt{names}(\rho') \subseteq I_1$ and $\rho' \dashv \bigoplus_{1 \le i \le n} \rho_i$ and take $\rho \overset{\text{def}}{=} \mathtt{e} + \overline{\alpha}.\rho'$. From $\rho \dashv \sigma_1$ we obtain $\rho \dashv \sigma_2$. Now $\rho \mid \sigma_2 \longrightarrow \rho' \mid \sigma_2'$ and from Proposition 1 we conclude $\rho' \dashv \sigma_2'$. □

We are not aware of any process semantics corresponding to $\preceq$ in the literature. However, if we restrict $\preceq$ to contracts with the same interface, we retrieve a well-known semantics: the *must-testing preorder* [14]. We recall the definition of the must preorder for the behaviors in Section 2.

Definition 4 (Must preorder [19]). *A sequence of transitions $\sigma_0 \mid \rho_0 \longrightarrow \sigma_1 \mid \rho_1 \longrightarrow \cdots$ is a* maximal computation *if either it is infinite or the last term $\sigma_n \mid \rho_n$ is such that $\sigma_n \mid \rho_n \not\longrightarrow$.*

Let $\mathtt{e} \notin \mathtt{names}(\sigma)$. Let σ $\mathtt{must}$ ρ if, for every maximal computation $\sigma \mid \rho = \sigma_0 \mid \rho_0 \longrightarrow \sigma_1 \mid \rho_1 \longrightarrow \cdots$, there exists $n \ge 0$ such that $\rho_n \xrightarrow{\mathtt{e}}$. We write $\sigma \sqsubseteq_{\mathtt{must}} \sigma'$ if and only if, for every ρ, σ $\mathtt{must}$ ρ implies σ' $\mathtt{must}$ ρ.

Before showing the precise relationship between $\preceq$ and $\sqsubseteq_{\mathtt{must}}$, let us comment on the differences between $\rho \dashv \sigma$ and σ $\mathtt{must}$ ρ. The $\mathtt{must}$ relation is such that σ $\mathtt{must}$ $\mathtt{e} + \rho$ holds for every σ, so that the observers of the form $\mathtt{e} + \rho$ are useless for discriminating between different (service) behaviors in $\sqsubseteq_{\mathtt{must}}$. However this is not the case for $\preceq$. For example $\mathtt{e} + a \not\dashv \overline{a}$ (whilst $\overline{a}$ $\mathtt{must}$ $\mathtt{e} + a$). In our setting it makes no sense to declare that $\mathtt{e} + a$ is compliant with $\overline{a}$ with the justification that, at some point in a computation starting from $\mathtt{e} + a \mid \overline{a}$, the client can emit $\mathtt{e}$. When a client and a service interact, actions cannot be undone. On the other hand we have $\mathtt{e} \oplus \mathtt{e} \dashv \mathbf{\Omega}$ and $\mathbf{\Omega}$ $\mathtt{m\cancel{u}st}$ $\mathtt{e} \oplus \mathtt{e}$. That is a (client) behavior compliant with a divergent (service) behavior is such that it is compliant with every (service) behavior, hence it is useless for discriminating between different (service) behaviors in $\preceq$. Historically, $\mathbf{\Omega}$ $\mathtt{m\cancel{u}st}$ $\mathtt{e} \oplus \mathtt{e}$ has been motivated by the fact that the divergent process may prevent the observer from performing the one internal reduction that leads to success. In a distributed setting this motivation is no longer sustainable, since client and service will usually run independently on different processors. Finally, consider a divergent (client) behavior ρ. In the $\mathtt{must}$ relation such observer never succeeds unless $\rho \xrightarrow{\mathtt{e}}$. In the $\dashv$ relation such observer is compliant so long as all of its finite computations lead to a successful state. So, for example, the client behaviors $\mathtt{rec}\ x.a.\mathtt{e} + x$ and $a.\mathtt{e}$ have the same discriminating power as far as $\preceq$ is concerned.

Theorem 2. *Let* $I = \mathtt{names}(\sigma)$. $I : \sigma \preceq I : \tau$ *if and only if* $\sigma \sqsubseteq_{\mathtt{must}} \tau$.

Proof ("Only if" part). Let $\mathcal{R}$ be a coinductive subcontract and $(I : \sigma, I : \tau) \in \mathcal{R}$. By contradiction, let $\sigma \sqsubseteq_{\mathtt{must}} \rho$ and τ $\mathtt{m\cancel{u}st}$ ρ. Therefore there exists a maximal computation $\tau \mid \rho = \tau_0 \mid \rho_0 \longrightarrow \tau_1 \mid \rho_1 \longrightarrow \cdots$ such that, for every i, $\rho_i \overset{\mathtt{e}}{\not\longrightarrow}$. We distinguish two cases: (a) the computation is finite, (b) the computation is infinite.

In case (a) there exists n such that $\tau_n \mid \rho_n \not\longrightarrow$ and let $\tau \overset{\varphi}{\Longrightarrow} \tau_n$ and $\rho \overset{\overline{\varphi}}{\Longrightarrow} \rho_n$. Since $(I : \sigma, I : \tau) \in \mathcal{R}$, from Proposition 1 we obtain $\sigma \overset{\varphi}{\Longrightarrow} \sigma_1, \ldots, \sigma \overset{\varphi}{\Longrightarrow} \sigma_m$, $m \geq 1$, such that $(I : \bigoplus_{1 \leq i \leq m} \sigma_i, I : \tau_n) \in \mathcal{R}$. If $\sigma_i\uparrow$, for some $1 \leq i \leq m$, then the derivations $\sigma \overset{\varphi}{\Longrightarrow} \sigma_i$ and $\rho \overset{\overline{\varphi}}{\Longrightarrow} \rho_n$ may be zipped, thus obtaining an infinite computation $\sigma \mid \rho \Longrightarrow \sigma_i \mid \rho_n \longrightarrow \cdots$ that contradicts $\sigma \sqsubseteq_{\mathtt{must}} \rho$. Otherwise, for every $1 \leq i \leq m$, $\sigma_i\downarrow$. From $(I : \bigoplus_{1 \leq i \leq m} \sigma_i, I : \tau_n) \in \mathcal{R}$ we obtain that for every R such that $\tau_n \Downarrow \mathrm{R}$ there exists $\mathrm{R}' \subseteq \mathrm{R}$ such that $\bigoplus_{1 \leq i \leq m} \sigma_i \Downarrow \mathrm{R}'$. In other words, there exists i such that $\sigma_i \Longrightarrow \sigma_i' \not\longrightarrow$ and $\mathtt{init}(\sigma_i') \subseteq \mathrm{R}$. Hence, $\sigma \mid \rho \Longrightarrow \sigma_i' \mid \tau_n \not\longrightarrow$, which again contradicts $\sigma\ \mathtt{must}\ \rho$.

In case (b), we distinguish two subcases:

b1. there exists n such that $\tau_n\uparrow$ or $\rho_n\uparrow$. Then using an argument similar to case (a), it is possible to show a contradiction to $\sigma \sqsubseteq_{\mathtt{must}} \rho$.

b2. The behaviors τ and ρ communicate infinitely often, that is the computation may be unzipped into $\tau \overset{\varphi}{\Longrightarrow}$ and $\rho \overset{\overline{\varphi}}{\Longrightarrow}$, where φ is infinite. It is easy to prove that, for every finite prefix φ' of φ there is σ' such that $\sigma \overset{\varphi'}{\Longrightarrow} \sigma'$. Therefore there exists an infinite computation of $\sigma \mid \rho$ that transits in the same terms $\rho_0, \rho_1, \ldots$ as the ones of $\tau \mid \rho$. This contradicts that $\sigma\ \mathtt{must}\ \rho$.

("If" part). We prove that

$$\mathcal{R} = \{(I : \tau_1, I : \tau_2) \mid \mathtt{names}(\tau_1) \subseteq I \text{ and } \tau_1 \sqsubseteq_{\mathtt{must}} \tau_2\} .$$

is a coinductive subcontract. Let $(I : \tau_1, I : \tau_2) \in \mathcal{R}$. If $\tau_1\uparrow$ there is nothing to prove. Let $\tau_1\downarrow$; the proofs of the three conditions of Definition 3 are in order.

1. By contradiction, let $\tau_2\uparrow$. Then τ_1 $\texttt{must}$ $\texttt{e} \oplus \texttt{e}$ whereas τ_2 $\texttt{m}\cancel{\texttt{u}}\texttt{st}$ $\texttt{e} \oplus \texttt{e}$ which is absurd, hence $\tau_2\downarrow$.
2. Let $\mathrm{R}_1, \dots, \mathrm{R}_n$ be the ready sets of τ_1. Assume by contradiction that there exists R such that $\tau_2 \Downarrow \mathrm{R}$ and $\mathrm{R}_i \not\subseteq \mathrm{R}$ for every $1 \leq i \leq n$. That is, every R_i is nonempty and, for every $1 \leq i \leq n$, there exists $\alpha_i \in \mathrm{R}_i$ such that $\alpha_i \notin \mathrm{R}$. From $\tau_2\downarrow$ and the definition of ready set, there exists τ_2' such that $\tau_2 \Longrightarrow \tau_2' \longrightarrow\!\!\!\!\!/$ and $\texttt{init}(\tau_2') \subseteq \mathrm{R}$. Now τ_1 $\texttt{must}$ $\sum_{1\leq i\leq n} \overline{\alpha_i}.\texttt{e}$ but τ_2 $\texttt{m}\cancel{\texttt{u}}\texttt{st}$ $\sum_{1\leq i\leq n} \overline{\alpha_i}.\texttt{e}$, which is absurd.
3. Let $\tau_2 \stackrel{\alpha}{\Longrightarrow} \tau_2'$. Then also $\tau_1 \stackrel{\alpha}{\Longrightarrow}$. In fact, if this were not the case, then τ_1 $\texttt{must}$ $(\texttt{e} \oplus \texttt{e}) + \overline{\alpha}$ but τ_2 $\texttt{m}\cancel{\texttt{u}}\texttt{st}$ $(\texttt{e} \oplus \texttt{e}) + \overline{\alpha}$, which is absurd. Let $\rho_1, \dots, \rho_n$ be all the derivatives of τ_1 after α, namely $\tau_1 \Longrightarrow\xrightarrow{\alpha} \rho_i$ for every $1 \leq i \leq n$. (This set is finite because $\tau_1\downarrow$.) We show that $(I : \bigoplus_{1\leq i\leq n} \rho_i, I : \tau_2') \in \mathcal{R}$, that is $\bigoplus_{1\leq i\leq n} \rho_i \sqsubseteq_{\texttt{must}} \tau_2'$. By contradiction, let $\bigoplus_{1\leq i\leq n} \rho_i \not\sqsubseteq_{\texttt{must}} \tau_2'$, then there exists ρ such that $\bigoplus_{1\leq i\leq n} \rho_i$ $\texttt{must}$ ρ but τ_2' $\texttt{m}\cancel{\texttt{u}}\texttt{st}$ ρ. In particular ρ_i $\texttt{must}$ ρ for every $1 \leq i \leq n$. In this case τ_1 $\texttt{must}$ $(\texttt{e} \oplus \texttt{e}) + \overline{\alpha}.\rho$ but τ_2 $\texttt{m}\cancel{\texttt{u}}\texttt{st}$ $(\texttt{e} \oplus \texttt{e}) + \overline{\alpha}.\rho$, which is absurd. We conclude that $(I : \bigoplus_{1\leq i\leq n} \rho_i, I : \tau_2') \in \mathcal{R}$.

□

4 Dual contracts

We now turn our attention to the problem of querying a database of Web services contracts. The basic idea is that given a client $I : \rho$ we wish to find all the service contracts $J : \sigma$ such that $I \subseteq J$ and $\rho \dashv \sigma$. We can partly simplify the problem by computing *one particular service contract* $J_0 : \sigma_0$ such that $I \subseteq J_0$ and $\rho \dashv \sigma_0$ and then by taking all the services in the registry that are larger than this one. Actually, in order to maximize the number of service contracts returned as answer to the query, the *dual operator* of a (client) contract $I : \rho$ should compute a behavior $\rho^\perp$, so that $I : \rho^\perp$ is *smallest service contract* that satisfies the client contract $I : \rho$. We call such contract the principal dual contract of $I : \rho$.

It is convenient to restrict the definition of dual to those behaviors ρ that never lead to $\mathbf{0}$ without emitting $\texttt{e}$. For example, the behavior $a.\texttt{e}+b.\mathbf{0}$ describes a client that succeeds if the service proposes $\overline{a}$, but that fails if the service proposes $\overline{b}$. As far as querying is concerned, such behavior is completely equivalent to $a.\texttt{e}$. As another example, the degenerate client behavior $\mathbf{0}$ is such that no service will ever satisfy it. In general, if a client is unable to handle a particular action, like b in the first example, it should simply omit that action from its behavior. We say that a (client) contract $I : \rho$ is *canonical* if, whenever $\rho \stackrel{\varphi}{\Longrightarrow} \rho'$ is maximal, then $\varphi = \varphi'\texttt{e}$ and $\texttt{e} \notin \texttt{names}(\varphi')$. For example $\{a\} : a.\texttt{e}$, $\{a\} : \mathsf{rec}\ x.a.x$, and $\emptyset : \mathbf{\Omega}$ are canonical; $\{a, b\} : a.\texttt{e} + b.\mathbf{0}$ and $\{a\} : \mathsf{rec}\ x.a + x$ are not canonical.

To ease the definition of the dual we introduce a countable set of behavior names, called $\texttt{dual}(I, \rho)$, which are defined by equations $\texttt{dual}(I, \rho) \stackrel{\text{def}}{=} \sigma$, where I is finite and σ is a behavior containing behavior names $\texttt{dual}(I', \rho')$ instead of variables. In order for the definition to be well founded, we need to prove a preliminary finiteness result.

Proposition 3. *Continuations of prefixes in behaviors are finite. Namely, for every σ the set $\{\sigma' \mid$ there exist φ, α such that $\sigma \overset{\varphi}{\Longrightarrow}\xrightarrow{\alpha} \sigma'\}$ is finite.*

Proof. Let σ be a behavior. Then there exists n_σ such that for every $\sigma \overset{\varphi}{\Longrightarrow} \sigma'$, the different subterms $\alpha.\rho$ of σ' that are not underneath a prefix, or an internal choice, or a recursion are less than n_σ. To compute this n_σ, count all the prefixes $\alpha.\rho$ (for every α and ρ) in σ. It is possible to show that in σ' every $\alpha.\rho$ not underneath a prefix or an internal choice or a recursion may be traced back to one of those counted in n_σ. A standard technique for demonstrating this statement uses labels [16]. $\square$

Notice that the set $\{\sigma' \mid \sigma \Longrightarrow \sigma'\}$ may be infinite. This is the case when $\sigma = \mathsf{rec}\ x.a + x$. However, the set $\{\sigma' \mid \mathsf{rec}\ x.a + x \Longrightarrow\xrightarrow{a} \sigma'\}$ is finite and it is $\{\mathbf{0}\}$. An immediate consequence of Proposition 3 is that behaviors are image finite. Namely, for every σ and α, the set $\{\sigma' \mid \sigma \Longrightarrow\xrightarrow{\alpha} \sigma'\}$ is finite.

Definition 5 (Dual contract). *Let $I : \rho$ be a canonical contract. Let $\mathtt{co}(I) \overset{\text{def}}{=} \{\overline{a} \mid a \in I\}$. The* dual *of ρ with respect to I, written $\mathtt{dual}(I, \rho)$, is defined as follows:*

$$\mathtt{dual}(I,\rho) \overset{\text{def}}{=} \begin{cases} \mathbf{\Omega}, & \textit{if } \mathtt{init}(\rho) = \{\mathtt{e}\} \\[1ex] \sum_{\rho \Downarrow \mathrm{R}, \mathrm{R}\setminus\{\mathtt{e}\} \neq \emptyset} \Big(\underbrace{\mathbf{0} \oplus}_{\textit{if } \mathtt{e} \in \mathrm{R}} \bigoplus_{\rho \Longrightarrow \xrightarrow{\alpha \in \mathrm{R}\setminus\{\mathtt{e}\}} \rho'} \overline{\alpha}.\mathtt{dual}(I, \rho') \Big) & \\ \underbrace{+ \Big(\mathbf{0} \oplus \bigoplus_{\alpha \in (I \cup \mathtt{co}(I)) \setminus \mathtt{init}(\rho)} \alpha.\mathbf{\Omega} \Big)}_{\textit{if } (I \cup \mathtt{co}(I)) \setminus \mathtt{init}(\rho) \neq \emptyset}, & \textit{otherwise} \end{cases}$$

The behavior $\mathtt{dual}(I, \rho)$ is well defined because the summands are always finite: the external summand is finite because behaviors manifest finitely many different ready sets; the internal summand is finite because of Proposition 3. It follows that from every equation $\mathtt{dual}(I, \rho) \overset{\text{def}}{=} \sigma$ it is possible to reach a finite set of behavior constants. It is folklore to transform such equations into recursive behaviors, thus conforming with the syntax of Section 2.

Few comments about $\mathtt{dual}(I, \rho)$, when $\mathtt{init}(\rho) \neq \{\mathtt{e}\}$, follow. In this case, the behavior ρ may autonomously transit to different states, each one offering a particular ready set. Thus the dual behavior leaves the choice to the client: this is the reason for the external choice in the second line. Once the state has been chosen, the client offers to the service a spectrum of possible actions: this is the reason for the internal choice in the first line (of the "otherwise" clause). The second line covers all the cases of actions that are allowed by the interface and that are not offered by the client. The point is that the dual operator must compute the principal (read, the smallest) service contract that satisfies the client, and the smallest convergent behavior with respect to a (finite) interface I is $\bigoplus_{\alpha \in I \cup \mathtt{co}(I)} \alpha$. The external choice in the second line distributes the proper dual contract over the internal choice of all the actions not allowed by the interface. The $\mathbf{0}$ summand accounts for the possibility that none of the actions not allowed by the interface is present. For example, $\texttt{dual}(\{a\}, a.\mathsf{e}) = \overline{a}.\mathbf{\Omega} + (\mathbf{0} \oplus a.\mathbf{\Omega})$. The dual of a divergent (canonical) client is also well defined: $\texttt{dual}(\{a\}, \mathsf{rec}\ x.a.\mathsf{e} + x) = \overline{a}.\mathbf{\Omega} + (\mathbf{0} \oplus a.\mathbf{\Omega})$. We notice that the definition also covers duals of nonterminating clients: $\texttt{dual}(\{a\}, \mathsf{rec}\ x.a.x) = \overline{a}.\texttt{dual}(\{a\}, \mathsf{rec}\ x.a.x) + (\mathbf{0} \oplus a.\mathbf{\Omega})$, namely $\texttt{dual}(\{a\}, \mathsf{rec}\ x.a.x) \simeq \mathsf{rec}\ x.(\overline{a}.x + (\mathbf{0} \oplus a.\mathbf{\Omega}))$.

Theorem 3. *Let $I : \rho$ be a canonical contract. Then:*

1. $\rho \dashv \texttt{dual}(I, \rho)$*;*
2. *if $\rho \dashv \sigma$ and $\texttt{names}(\sigma) \subseteq I$, then $I : \texttt{dual}(I, \rho) \preceq I : \sigma$.*

Proof. Regarding item 1, we remark that, by definition of dual, every derivation $\rho \mid \texttt{dual}(I, \rho) \Longrightarrow \rho'' \mid \sigma$ may be decomposed into $\rho \mid \texttt{dual}(I, \rho) \Longrightarrow \rho' \mid \texttt{dual}(I, \rho') \Longrightarrow \rho'' \mid \tau$, where $\rho' \Longrightarrow \rho''$ and $\texttt{dual}(I, \rho') \Longrightarrow \tau$.

If $\tau\uparrow$ then $\texttt{dual}(I, \rho') = \mathbf{\Omega}$, which means that $\{\mathsf{e}\} = \texttt{init}(\rho')$. In this case, the conditions in Definition 1 are satisfied. If $\rho'' \mid \tau \longrightarrow\!\!\!\!\!\!/\ \ $ then assume by contradiction that $\mathsf{e} \notin \texttt{init}(\rho'')$. By definition of canonical client, $\texttt{init}(\rho'') \neq \emptyset$. Therefore, by definition of dual, $\texttt{dual}(I, \rho') \Downarrow \mathrm{R}$ implies $\mathrm{R} \neq \emptyset$ because $\texttt{dual}(I, \rho')$ has an empty ready set provided every ready set of ρ' contains e, which is not the case by hypothesis. Hence we conclude $\texttt{init}(\tau) \neq \emptyset$ and $\texttt{co}(\texttt{init}(\rho'')) \cap \texttt{init}(\tau) \neq \emptyset$ by definition of dual, which is absurd.

Regarding item 2, let $\mathcal{R}$ be the least relation such that:

– $(I : \texttt{dual}(I, \rho), I : \sigma) \in \mathcal{R}$;
– if $\sigma \overset{\varphi}{\Longrightarrow}$, $\rho \overset{\overline{\varphi}}{\Longrightarrow}$, $\sigma \downarrow \varphi$, and $\rho \downarrow \overline{\varphi}$, then $(I : \bigoplus_{\rho \overset{\overline{\varphi}}{\Longrightarrow} \rho'} \texttt{dual}(I, \rho'), I : \sigma') \in \mathcal{R}$ for every $\sigma \overset{\varphi}{\Longrightarrow} \sigma'$;
– if $\sigma \overset{\varphi}{\Longrightarrow}$ and either $\rho \overset{\overline{\varphi}}{\not\Longrightarrow}$ or $\sigma \uparrow \varphi$ or $\rho \uparrow \overline{\varphi}$, then $(I : \mathbf{\Omega}, I : \sigma') \in \mathcal{R}$ for every $\sigma \overset{\varphi}{\Longrightarrow} \sigma'$.

We prove that $\mathcal{R}$ is a coinductive subcontract. Let $(I : \tau_1, I : \tau_2) \in \mathcal{R}$. If $\tau_1\uparrow$ there is nothing to prove; therefore let $\tau_1\downarrow$. The conditions of Definition 3 are proved in order:

1. By definition of $\mathcal{R}$, there exists φ such that $\sigma \overset{\varphi}{\Longrightarrow} \tau_2$ and $\sigma \downarrow \varphi$. This allows us to conclude $\tau_2\downarrow$.
2. By definition of $\mathcal{R}$, there exist $\rho_1, \dots, \rho_m$ such that $\tau_1 = \bigoplus_{1 \leq j \leq m} \texttt{dual}(I, \rho_j)$ and $\rho \overset{\overline{\varphi}}{\Longrightarrow} \rho_j$. Let $\mathrm{R}_1, \dots, \mathrm{R}_n$ be the ready sets of τ_1. Assume by contradiction that $\tau_2 \Downarrow \mathrm{R}$ and $\mathrm{R}_i \not\subseteq \mathrm{R}$ for every $1 \leq i \leq n$. In other words, for every $1 \leq i \leq n$ there exists $\alpha_i \in \mathrm{R}_i$ such that $\alpha_i \notin \mathrm{R}$. By definition of $\texttt{dual}(I, \rho_j)$ this means that for every $1 \leq j \leq m$ and for every ρ' such that $\rho \overset{\overline{\varphi}}{\Longrightarrow} \rho_j \Longrightarrow \rho' \longrightarrow\!\!\!\!\!\!/\ \ $ we have $\texttt{co}(\texttt{init}(\rho')) \cap \mathrm{R} = \emptyset$ because $\texttt{dual}(I, \rho_j) \Downarrow \{\overline{\alpha}\}$ for every $\alpha \in \texttt{init}(\rho') \setminus \{\mathsf{e}\}$. Furthermore, it cannot be that $\mathsf{e} \in \texttt{init}(\rho')$ for every ρ', for otherwise we would have $\tau_1 \Downarrow \emptyset$ again by definition of $\texttt{dual}(I, \rho_j)$. Hence, there exists at least one ρ' such that $\rho \overset{\overline{\varphi}}{\Longrightarrow} \rho' \longrightarrow\!\!\!\!\!\!/\ \ $ and $\mathsf{e} \notin \texttt{init}(\rho')$.

Since $\tau_2\!\downarrow$ we can find τ'_2 such that $\tau_2 \Longrightarrow \tau'_2 \longrightarrow\!\!\!\!\!\not\;\;$ and $\mathtt{init}(\tau'_2) \subseteq \mathrm{R}$. By zipping the derivations $\rho \overset{\overline{\varphi}}{\Longrightarrow} \rho'$ and $\sigma \overset{\varphi}{\Longrightarrow} \tau'_2$ we obtain $\rho \mid \sigma \Longrightarrow \rho' \mid \tau'_2$. We derive $\rho' \mid \tau'_2 \not\longrightarrow$ because $\rho' \not\longrightarrow$ and $\tau'_2 \not\longrightarrow$ and $\mathtt{co}(\mathtt{init}(\rho')) \cap \mathtt{init}(\tau'_2) \subseteq \mathtt{co}(\mathtt{init}(\rho')) \cap \mathrm{R} = \emptyset$. Furthermore, $\mathtt{e} \notin \mathtt{init}(\rho')$, but this is absurd from the hypothesis $\rho \dashv \sigma$.

3. Let $\tau_2 \overset{\alpha}{\Longrightarrow} \tau'_2$. We must prove that there exist $\tau''_1, \dots, \tau''_n$ such that $\tau_1 \overset{\alpha}{\Longrightarrow} \tau''_i$ and $(I : \bigoplus_{1 \leq i \leq n} \tau''_i, I : \tau'_2) \in \mathcal{R}$. If $\rho \overset{\overline{\varphi\alpha}}{\not\Longrightarrow}$ or $\rho \overset{\overline{\varphi\alpha}}{\Longrightarrow}$ and $\rho \!\uparrow\! \overline{\varphi\alpha}$, then $\tau_1 \overset{\alpha}{\Longrightarrow} \mathbf{\Omega}$ by definition of $\mathtt{dual}(I,\rho)$ and $(I : \mathbf{\Omega}, I : \tau'_2) \in \mathcal{R}$ by definition of $\mathcal{R}$. If $\rho \overset{\overline{\varphi\alpha}}{\Longrightarrow}$ and $\sigma \!\uparrow\! \varphi\alpha$, then from $\rho \dashv \sigma$ we obtain that $\rho \overset{\overline{\varphi\alpha}}{\Longrightarrow} \rho'$ implies $\{\mathtt{e}\} = \mathtt{init}(\rho')$, so again $\tau_1 \overset{\alpha}{\Longrightarrow} \mathbf{\Omega}$ by definition of $\mathtt{dual}(I,\rho)$ and $(I : \mathbf{\Omega}, I : \tau'_2) \in \mathcal{R}$ by definition of $\mathcal{R}$. The last case is when $\sigma \!\downarrow\! \varphi\alpha$ and $\rho \overset{\overline{\varphi\alpha}}{\Longrightarrow}$ and $\rho \!\downarrow\! \overline{\varphi\alpha}$. From the definition of $\mathcal{R}$ we obtain that $(I : \bigoplus_{\rho \overset{\overline{\varphi\alpha}}{\Longrightarrow} \rho''} \mathtt{dual}(I,\rho''), I : \tau'_2) \in \mathcal{R}$ and we conclude by observing that $\{\mathtt{dual}(I,\rho'') \mid \rho \overset{\overline{\varphi\alpha}}{\Longrightarrow} \rho''\} = \{\tau'' \mid \tau_1 \overset{\alpha}{\Longrightarrow} \tau''\}$. □

5 Choreographies

A *choreography* is meant to describe the parallel composition of n services (called participants) that communicate with each other by means of private names and with the external world by means of public names. Standard languages for describing choreographies, such as the Web Service Choreography Description Language (WS-CDL [15]), describe choreography activities, communications, and their mutual dependencies from a global perspective. From such descriptions it is possible to synthesize the so-called *end-point projections*, namely the behavioral specifications of the single participants, provided that the global description respects some fundamental constraints (see for example [5] and [4]).

In this work, we identify a choreography with the composition of its end-point projections, which are represented as contracts. Formally, a choreography is a term

$$\Sigma ::= (I_1 : \sigma_1 \mid \cdots \mid I_n : \sigma_n) \setminus L$$

where L is a finite subset of names representing the private names of the choreography. We write $\Sigma[i \mapsto J : \rho]$ for the choreography that is the same as Σ except that (the contract of) the i-th participant has been replaced by $J : \rho$. We also write Σ_L whenever we want to recall the private ports of the choreography.

The transition relation of choreographies is defined using that of behaviors by the following rules:

$$\frac{\sigma \xrightarrow{\alpha} \sigma' \qquad \mathtt{names}(\alpha) \notin L}{\Sigma_L[i \mapsto I : \sigma] \xrightarrow{\alpha} \Sigma_L[i \mapsto I : \sigma']} \qquad \frac{\sigma \longrightarrow \sigma'}{\Sigma_L[i \mapsto I : \sigma] \longrightarrow \Sigma_L[i \mapsto I : \sigma']}$$

$$\frac{i \neq j \quad \sigma \xrightarrow{\alpha} \sigma' \quad \rho \xrightarrow{\overline{\alpha}} \rho' \quad \mathtt{names}(\alpha) \in L}{\Sigma_L[i \mapsto I : \sigma][j \mapsto J : \rho] \longrightarrow \Sigma_L[i \mapsto I : \sigma'][j \mapsto J : \rho']}$$

Having provided choreographies with a transition relation, the notions of *convergence*, *divergence*, and *ready set* can be immediately extended to choreographies from their previous definitions. Similarly, the notion of behavioral compliance may be extended in order to relate the behavior of a client with (the behavior of) a choreography, which we denote by $\rho \dashv \Sigma$. That is, a choreography $\Sigma = (I_1 : \sigma_1 \mid \cdots \mid I_n : \sigma_n) \setminus L$ is a (complex) service whose interface is $\mathtt{int}(\Sigma) \overset{\text{def}}{=} \bigcup_{1 \leq i \leq n} I_i \setminus L$ and whose behavior is the combination of the behaviors of the participants running in parallel. In this setting, a (client) contract $J : \rho$ *is compliant with* the choreography Σ if $J \subseteq \mathtt{int}(\Sigma)$ and $\rho \dashv \Sigma$.

Definition 6 (Choreography refinement). *Let $\Sigma = (I_1 : \sigma_1 \mid \cdots \mid I_n : \sigma_n) \setminus L$ and $\Sigma' = (I'_1 : \sigma_1 \mid \cdots \mid I'_n : \sigma_n) \setminus L'$. The choreography Σ' is a* refinement *of Σ if and only if the following conditions hold:*

1. $\mathtt{int}(\Sigma) = \mathtt{int}(\Sigma')$*;*
2. *for every $1 \leq i \leq n$ we have $I_i : \sigma_i \preceq I'_i : \sigma'_i$;*
3. *for every $1 \leq i, j \leq n$ with $i \neq j$ we have that $L' \cap (\mathtt{names}(\sigma'_i) \setminus \mathtt{names}(\sigma_i)) \cap (\mathtt{names}(\sigma'_j) \setminus \mathtt{names}(\sigma_j)) = \emptyset$.*

Refinement defines a "safe" replacement of activities in a choreography with more detailed ones (such as their implementations) still preserving the overall interface of the choreography (condition 1). The replacing activities may have more capabilities than those offered by the replaced ones (condition 2) and it must not be the case that the new activities communicate with each other by means of names that do not occur in the original choreography (condition 3) for otherwise the original choreography specification would be violated.

We now prove a soundness result for the notion of refinement which represents a restricted form of precongruence of $\preceq$ with respect to the parallel composition. The result is not based on any particular property (e.g. deadlock freedom) of the choreography itself. We merely show that, from the point of view of a client interacting with a choreography as a whole, the refinement of the choreography does not jeopardize the completion of the client.

Theorem 4. *Let $I : \rho$ be compliant with a choreography Σ_1 and Σ_2 be a refinement of Σ_1. Then $I : \rho$ is also compliant with Σ_2.*

Proof. Let $\Sigma_1 = (I_1 : \sigma_1 \mid \cdots \mid I_n : \rho_n) \setminus L$ and let $\Sigma_2 = (J_1 : \tau_1 \mid \cdots \mid J_n : \tau_n) \setminus L'$. Consider a computation $\rho \mid \Sigma_2 \Longrightarrow \rho' \mid \Sigma'_2$ where $\Sigma'_2 = (J_1 : \tau'_1 \mid \cdots \mid J_n : \tau'_n) \setminus L$. By unzipping this computation we obtain that there exists a sequence φ of actions such that $\rho \overset{\varphi}{\Longrightarrow} \rho'$ and $\Sigma_2 \overset{\overline{\varphi}}{\Longrightarrow} \Sigma'_2$. By unzipping the computation of Σ_2 with respect to all of its participants we obtain n sequences $\varphi_1, \dots, \varphi_n$ of actions such that $\tau_i \overset{\varphi_i}{\Longrightarrow} \tau'_i$ for every $1 \leq i \leq n$. From condition 1 in the definition of choreography refinement the names in φ must be in $\mathtt{int}(\Sigma_1)$. From condition 3 in the definition of choreography refinement we know that participants in Σ_2 cannot communicate if not by means of names that were already present in Σ_1. Hence, for every $1 \leq i \leq n$, $\mathtt{names}(\varphi_i) \subseteq I_i$ and $\sigma_i \overset{\varphi_i}{\Longrightarrow}$. By zipping these derivations we obtain that $\Sigma_1 \overset{\overline{\varphi}}{\Longrightarrow}$ as well.

Let $\rho' \mid \Sigma'_2 \not\longrightarrow$. If $\Sigma_1 \uparrow \overline{\varphi}$ then, by $\rho \dashv \Sigma_1$ we derive $\{\mathtt{e}\} = \mathtt{init}(\rho')$. If $\Sigma'_1 \downarrow \overline{\varphi}$, then from $\Sigma_2 \not\longrightarrow$ and condition 2 in the definition of choreography refinement, we obtain that for every $1 \leq i \leq n$ there exists σ'_i such that $\sigma_i \overset{\varphi_i}{\Longrightarrow} \sigma'_i \not\longrightarrow$ and $\mathtt{init}(\sigma'_i) \subseteq \mathtt{init}(\tau'_i)$. Take $\Sigma'_1 = (I_1 : \sigma'_1 \mid \cdots \mid I_n : \sigma'_n) \setminus L$. Then $\rho \mid \Sigma_1 \Longrightarrow \rho' \mid \Sigma'_1 \not\longrightarrow$ and from $\rho \dashv \Sigma_1$ we conclude $\{\mathtt{e}\} \subseteq \mathtt{init}(\rho')$.

Suppose $\Sigma'_2\uparrow$. This may happen either because one (or more) participants diverge autonomously, or because two (or more) participants interact infinitely often. By definition of refinement and using the same arguments as above, we obtain that there exists Σ'_1 such that $\Sigma_1 \overset{\overline{\varphi}}{\Longrightarrow} \Sigma'_1$ and $\Sigma'_1\uparrow$. This may happen either because one (or more) participants diverge autonomously, or because two (or more) participants interact infinitely often (recall that the refined choreography cannot exhibit more synchronizations than the original one). Hence we conclude $\{e\} = \mathtt{init}(\rho')$. □

6 Concluding remarks

In this contribution we have studied a formal theory of Web services contracts. The subcontract preorder used in the theory arises semantically as in the testing setting, except that the notion of "passing a test" is essentially different and reflects more faithfully the interaction of clients and services. We have given two different characterizations of the subcontract preorder, one directly induced by the notion of compliance, the other one that is more amenable for an algorithmic implementation. The subcontract relation is effectively and efficiently applicable in any query-based system for service discovery because it is supported by a notion of principal dual contract. It is also applicable in choreographies for replacing contracts with larger ones.

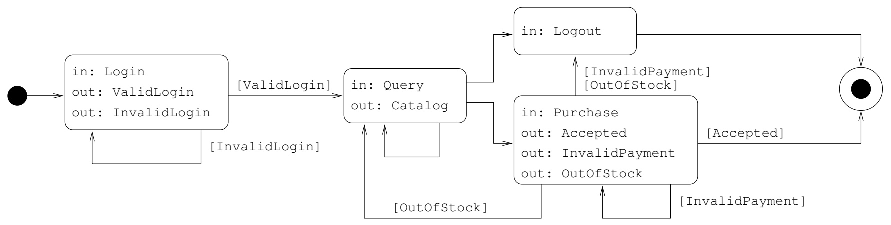

Fig. 1. Contract of a simple e-commerce service as a WSCL diagram.

The theory may be used as a foundation of Web services technologies, such as WSDL and WSCL. In [6] we already discussed how to encode WSDL message-exchange patterns and acyclic WSCL diagrams into the recursion-free fragment of the contract language. The language in this paper allows us to express also cyclic WSCL conversations by means of recursion. Figure 1 shows a WSCL diagram describing the protocol of a service requiring clients to login before they can issue a query. After the query, the service returns a catalog. From this point on, the client can decide whether to purchase an item from the catalog or to logout and leave. In case of purchase, the service may either report that the purchase was successful, or that the item is out-of-stock, or that the client's payment was refused. By interpreting names as message types, this e-commerce service can be encoded in our language by a contract whose behavior is:

$$\begin{aligned}&\mathsf{rec}\ x.\mathtt{Login}.(\overline{\mathtt{InvalidLogin}}.x \oplus \overline{\mathtt{ValidLogin}}.\mathsf{rec}\ y.\\&\quad \mathtt{Query}.\overline{\mathtt{Catalog}}.(y + \mathtt{Logout} + \mathsf{rec}\ z.\mathtt{Purchase}.\\&\qquad \overline{\mathtt{Accepted}} \oplus \overline{\mathtt{InvalidPayment}}.(z + \mathtt{Logout}) \oplus \overline{\mathtt{OutOfStock}}.(y + \mathtt{Logout})))\end{aligned}$$

We notice the correspondence between unlabeled (respectively, labeled) transitions in Figure 1 and external (respectively, internal) choices in the contract. We also notice how recursion is used for expressing iteration (the cycles in the figure) so that the client is given another chance whenever an action fails for some reason.

Several future research directions stem from this work. On the technical side, we would like to define and investigate an axiomatic characterization of $\preceq$. We expect such axiomatization to closely resemble the one for the $\sqsubseteq_{\mathtt{must}}$ preorder [19, 14]. On the linguistic side we would like to explore new process constructions that could take into account information available with contracts. For instance, imagine a client that wants to use a service exporting the contract $a \oplus b$; in the simple language of Section 2 the client cannot specify that it wants to connect on b if available, and on a otherwise, for the choice operators are symmetric. It is unclear to which extent such constructs affect the $\preceq$ preoder over contracts. The discussion of Proposition 2 suggests that there are interesting connections between the properties of the $\preceq$ preorder and the boolean operators over sets of compliant clients. We aim to further explore such set-theoretic interpretation of contracts and to devise a query language for service discovery that provides primitive operators for union, intersection, and negation for contracts. The authors of [6] provide a type system to extract the contract out of a recursion-free fragment of CCS-like process calculus. We aim to extend such type system to full CCS, although the task is not trivial because CCS processes may exhibit a non-regular behavior. When the behavior cannot be captured accurately by our contract language, regular under- and over-estimations must be provided. Finally, a major task is to move our investigation from a CCS-like formalism to a π-calculus one for taking into account and generalizing the forthcoming version of WSDL which enables the possibility to describe higher-order Web services.

References

1. A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, , A. Guízar, N. Kartha, C. K. Liu, R. Khalaf, D. König, M. Marin, V. Mehta, S. Thatte, D. van der Rijn, P. Yendluri, and A. Yiu. *Web Services Business Process Execution Language Version 2.0*, Jan. 2007. `http://docs.oasis-open.org/wsbpel/2.0/CS01/wsbpel-v2.0-CS01.html`.

2. A. Banerji, C. Bartolini, D. Beringer, V. Chopella, et al. *Web Services Conversation Language (*WSCL*) 1.0*, Mar. 2002. `http://www.w3.org/TR/2002/NOTE-wscl10-20020314`.
3. D. Beringer, H. Kuno, and M. Lemon. *Using* WSCL *in a* UDDI *Registry 1.0*, 2001. UDDI Working Draft Best Practices Document, `http://xml.coverpages.org/HP-UDDI-wscl-5-16-01.pdf`.
4. M. Bravetti and G. Zavattaro. Towards a unifying theory for choreography conformance and contract compliance. In *Pre-proceedings of 6th Symposium on Software Composition*, 2007.
5. M. Carbone, K. Honda, and N. Yoshida. Structured communication-centered programming for web services. In *Proceedings of 16th European Symposium on Programming*, LNCS, 2007.
6. S. Carpineti, G. Castagna, C. Laneve, and L. Padovani. A formal account of contracts for Web Services. In *WS-FM, 3rd Int. Workshop on Web Services and Formal Methods*, number 4184 in LNCS, pages 148–162. Springer, 2006.
7. G. Castagna, N. Gesbert, and L. Padovani. A theory of contracts for Web Services. In *Proceedings of 5th ACM SIGPLAN Workshop on Programming Language Technologies for XML*, pages 37–48, 2007.
8. R. Chinnici, H. Haas, A. A. Lewis, J.-J. Moreau, et al. *Web Services Description Language (*WSDL*) Version 2.0 Part 2: Adjuncts*, Mar. 2006. `http://www.w3.org/TR/2006/CR-wsdl20-adjuncts-20060327`.
9. R. Chinnici, J.-J. Moreau, A. Ryman, and S. Weerawarana. *Web Services Description Language (*WSDL*) Version 2.0 Part 1: Core Language*, Mar. 2006. `http://www.w3.org/TR/2006/CR-wsdl20-20060327`.
10. E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. *Web Services Description Language (*WSDL*) 1.1*, 2001. `http://www.w3.org/TR/2001/NOTE-wsdl-20010315`.
11. J. Colgrave and K. Januszewski. Using WSDL in a UDDI registry, version 2.0.2. Technical note, OASIS, 2004. `http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-wsdl-v2.htm`.
12. S. Gay and M. Hole. Subtyping for session types in the π-calculus. *Acta Informatica*, 42(2-3):191–225, 2005.
13. M. Hennessy. Acceptance trees. *JACM: Journal of the ACM*, 32(4):896–928, 1985.
14. M. C. B. Hennessy. *Algebraic Theory of Processes.* Foundation of Computing. MIT Press, 1988.
15. N. Kavantzas, D. Burdett, G. Ritzinger, T. Fletcher, Y. Lafon, and C. Barreto. *Web Services Choreography Description Language 1.0*, 2005. `http://www.w3.org/TR/2005/CR-ws-cdl-10-20051109/`.
16. J.-J. Levy. *Réductions Correctes et Optimales dans le Lambda-Calcul.* PhD thesis, Université Paris VII, Jan. 1978.
17. R. Milner. *A Calculus of Communicating Systems.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1982.
18. R. D. Nicola and M. Hennessy. Testing equivalences for processes. *Theor. Comput. Sci*, 34:83–133, 1984.
19. R. D. Nicola and M. Hennessy. CCS without τ's. In *TAPSOFT '87/CAAP '87: Proceedings of the International Joint Conference on Theory and Practice of Software Development, Volume 1: Advanced Seminar on Foundations of Innovative Software Development I and Colloquium on Trees in Algebra and Programming*, pages 138–152, London, UK, 1987. Springer-Verlag.
20. S. Parastatidis and J. Webber. *MEP SSDL Protocol Framework*, Apr. 2005. `http://ssdl.org`.