# Contract based Multi-party Service Composition*

Mario Bravetti Gianluigi Zavattaro

Department of Computer Science, University of Bologna, Italy

**Abstract.** In the context of Service Oriented Computing, contracts are descriptions of the observable behaviour of services. Contracts have been already successfully exploited to solve the problem of client/service composition. In this paper we consider services where the choice to perform an output may not depend on the environment and we revisit the theory of contracts in order to tackle the problem of composition of multiple services (not only one client with one service). Moreover, we relate our theory of contracts with the theory of testing preorder (interpreted as a subcontract relation) and we show that a compliant group of contracts is still compliant if every contract is replaced by one of its subcontract.

## 1 Introduction

Service Oriented Computing (SOC) is a novel paradigm for distributed computing based on services intended as autonomous and heterogeneous components that can be published and discovered via standard interface languages and publish/discovery protocols. One of the peculiarities of Service Oriented Computing, distinguishing it from other distributed computing paradigms (such as component based software engineering), is that it is centered around the so-called *message oriented architectures*. This means that, given a set of collaborating services, the current state of their interaction is stored inside the exchanged messages and not only within the services. From a practical viewpoint, this means that it is necessary to include, in the exchanged messages, the so-called correlation information that permits to a service to associate a received message to the correct session of interaction (in fact, the same service could be contemporarily involved in different sessions at the same time).

Web Services is the most prominent service oriented technology: Web Services publish their interface expressed in WSDL, they are discovered through the UDDI protocol, and they are invoked using SOAP.

Even if one of the declared goal of Web Services is to support the automatic discovery of services, this is not yet practically achieved. Two main problems are still to be satisfactorily solved. The first one, investigated by the semantic web research community, is concerned with the lack of semantic information in the description of services. The second problem, addressed in this paper, is concerned with the problem of guaranteeing that the interacting services are compliant in

* Research partially funded by EU Integrated Project Sensoria, contract n. 016004.

the sense that their behaviours are complementary. In particular, it is important to check whether a set of services, once combined in order to collaborate, are stuck-free.

In order to be able to check the compliance of the composed services, it is necessary that the services expose in their interface also the description of their expected behaviour. In the service oriented computing literature, this kind of information is referred to as the *service contract* [CL06]. More precisely, the service contract describes the sequence of input/output operations that the service intends to execute within a session of interaction with other services.

Contracts have been already investigated in the context of client-service interaction [CCL+06]. In this paper, we consider a different scenario: instead of analysing client-service architectures, we assume that several services interact according to a peer-to-peer architecture. Through the analysis of the service contracts we want to define a theory that, on the one hand, permits to formally verify whether the composed services are compliant (thus giving rise to a correct composition) and, on the other hand, permits to replace a service with another one without affecting the correctness of the overall system. In this case we say that the initially expected contract is replaced with one of its *subcontract*.

We foreseen at least two main applications for our theory of contracts. On the one hand, it can be exploited in the *service discovery* phase. Consider, for instance, a service system defined in terms of the contracts that should be exposed by each of the service components. The actual services to be combined could be retrieved independently one from the other (e.g. querying contemporarily different service registries) collecting that services that either exposes the expected contract, or one of its subcontract. On the other hand, the notion of subcontract could be useful in *service updates* in order to ensure backward compatibility. Consider, e.g., a service that should be updated in order to provide new functionalities; if the new version exposes a subcontract of the previous service, our theory ensures that the new service is a correct substitute for the previous one.

### 1.1 Technical contribution

We define two process calculi, one for contracts and one for systems composed of contracts. The latter is an extension of the former. The calculus for contracts is a typical process calculus distinguishing between deadlock and successful termination. This distinction is necessary in order to model the fact that a service could internally deadlock due to its internal parallelism. Another peculiarity of the calculus is that the decision to execute output actions may not depend on the other services in the system. In more technical terms, we do not admit mixed choices $a+\overline{b}$ where $a$ is an input action while $\overline{b}$ is an output. This reflects the fact that the current service oriented technologies (such as Web Services) are based on asynchronous communication and, as formally discussed in [Nes00], mixed choice is not reasonably implementable in asynchronously communicating concurrent systems. We avoid mixed choices imposing that all output actions $\overline{b}$ are preceded by an internal $\tau$ action; in this way a service first decide to execute the output, and only subsequently the message becomes actually available.

The second calculus for systems permits to compose in parallel contracts that interact within a session, and to add restrictions in order to model local channel names, that can be used only inside subsessions involving a proper subset of the composed contracts.

The calculus is used to define our notion of correct composed system; a system is correct if all the composed contracts are ensured to reach successful completion. Then, we introduce contract refinement as the possibility to independently replace each contract with a subcontract without breaking correctness. In general, this notion of refinement is rather complex to be checked mainly because it is defined in order to permit the contemporary and independent refinement of the contracts. Our main result is that in our calculus this notion of refinement coincides with a more treatable subcontract relation in which only one contract is replaced and the other ones are left unchanged. This new notion characterizes refinement in a testing scenario, thus giving us the possibility to resort to the theory of testing [DH84,RV05]. In particular, we show how to use the theory of *should-testing* [RV05] to prove that one contract is a subcontract of another one.

### 1.2 Related work

As stated above, we resort to the theory of testing. There are some relevant differences between our form of testing and the traditional one proposed by De Nicola-Hennessy [DH84]. The most relevant difference is that, besides requiring the success of the test, we impose also that the tested process should successfully complete its execution. This further requirement has important consequences; for instance, we do not distinguish between the always unsuccessful process $\mathbf{0}$ and other processes, such as $a.\mathbf{1} + a.b.\mathbf{1}$,[1] for which there are no guarantees of successful completion in any possible context. Another relevant difference is in the treatment of divergence: we do not follow the traditional catastrophic approach, but the fair approach introduced by the theory of should-testing of Rensink-Vogler [RV05]. In fact, we do not impose that all computations must succeed, but that all computations can always be extended in order to reach success.

Contracts have been investigated also by Fournet et al. [FHR$^+$04] and Carpineti et al. [CCL$^+$06].

In [FHR$^+$04] contracts are CCS-like processes; a generic process $P$ is defined as compliant to a contract $C$ if, for every tuple of names $\tilde{a}$ and process $Q$, whenever $(\nu\tilde{a})(C|Q)$ is stuck-free then also $(\nu\tilde{a})(P|Q)$ is. Our notion of contract refinement differs from stuck-free conformance mainly because we consider a different notion of stuckness. In [FHR$^+$04] a process state is stuck (on a tuple of channel names $\tilde{a}$) if it has no internal moves (but it can execute at least one action on one of the channels in $\tilde{a}$). In our approach, an end-states different from successful termination is stuck (independently of any tuple $\tilde{a}$). Thus, we distinguish between internal deadlock and successful completion while this is not the case in [FHR$^+$04]. Another difference follows from the exploitation of

[1] We use $\mathbf{0}$ to denote unsuccessful termination and $\mathbf{1}$ for successful completion.

the restriction $(\nu\tilde{a})$; this is used in [FHR$^+$04] to explicitly indicate the local channels of communication used between the contract $C$ and the process $Q$. In our context we can make a stronger *closed-world* assumption (corresponding to a restriction on all channel names) because service contracts do not describe the entire behaviour of a service, but the flow of execution of its operations inside one session of communication.

The closed-world assumption is considered also in [CCL$^+$06] where, as in our case, a service oriented scenario is considered. In particular, in [CCL$^+$06] a theory of contracts is defined for investigating the compatibility between one client and one service. Our paper consider multi-party composition where several services are composed in a peer-to-peer manner. Moreover, we impose service substitutability as a mandatory property for our notion of refinement; this does not hold in [CCL$^+$06] where it is not in general possible to substitute a service exposing one contract with another one exposing a subcontract. Another relevant difference is that the contracts in [CCL$^+$06] comprises also mixed choices.

**Structure of the paper.** Section 2 reports syntax and semantics of the process calculi. In Section 3 we describe our theory of contracts and we prove our main results. Finally, Section 4 reports some conclusive remarks.

## 2 Syntax and Semantics of the Process Calculi

In this Section we introduce incrementally the two calculi. The first one is the calculus for contracts; it is a typical calculus comprising two possible final states (failure or success), input and output prefixes, sequencing, choice, parallel composition, restriction and repetition.

### 2.1 Definition of Contracts

We assume a denumerable set of names $\mathcal{N} = \{a, b, c, ...\}$. The set $\mathcal{N}_{loc} = \{a_* \mid a \in \mathcal{N}\}$ is the set of local names. We take $\alpha$ to range over the set of all names $\mathcal{N}_{all} = \mathcal{N} \cup \mathcal{N}_{loc}$. The set $\mathcal{A} = \mathcal{N} \cup \{\overline{a} \mid a \in \mathcal{N}\}$ is the set of input and output actions. The set $\mathcal{A}_{loc} = \mathcal{N}_{loc} \cup \{\overline{a}_* \mid a_* \in \mathcal{N}_{loc}\}$ is the set of input and output local actions. We take $\beta$ to range over the set of all actions $Act = \mathcal{A}_{loc} \cup \mathcal{A} \cup \{\tau\}$, where $\tau$ denotes an internal computation.

**Definition 1. (Contracts)** *The syntax of contracts is defined by the following grammar*

$$\begin{aligned} C \ ::= \ & \mathbf{0} \ \mid \ \mathbf{1} \ \mid \ \tau \ \mid \ \alpha \ \mid \ \tau;\overline{\alpha} \ \mid \\ & C;C \ \mid \ C{+}C \ \mid \ C|C \ \mid \ C\backslash M \ \mid \ C^* \end{aligned}$$

*where $M \subseteq \mathcal{N}_{loc}$. The set of all contracts $C$ is denoted by $\mathcal{P}_{con}$.*

We consider four possible atoms: unsuccessful termination $\mathbf{0}$, successful termination $\mathbf{1}$, input action $a$, and silent move $\tau$. Output actions always appear after an internal move $\tau$. The operators are: sequencing $\_;\_$, choice $\_+\_$, parallel $\_|\_$, restriction $\_ \setminus M$, and repetition $\_^*$.

In the following we will omit trailing "$\mathbf{1}$" when writing contracts and, given a set of names $M$, with $\overline{M} = \{\overline{a} \mid a \in M\}$ we denote the set output actions performable on those names.

The operational semantics of contracts is defined by the rules in Table 1 (plus the omitted symmetric rules). We take $\lambda$ to range over the set of labels $\mathcal{L} = Act \cup \{\surd\}$, where $\surd$ denotes successful termination.

$$\mathbf{1} \xrightarrow{\surd} \mathbf{0} \qquad\qquad \beta \xrightarrow{\beta} \mathbf{1}$$

$$\frac{C \xrightarrow{\lambda} C'}{C+D \xrightarrow{\lambda} C'} \qquad \frac{C \xrightarrow{\lambda} C' \quad \lambda \neq \surd}{C;D \xrightarrow{\lambda} C';D} \qquad \frac{C \xrightarrow{\surd} C' \quad D \xrightarrow{\lambda} D'}{C;D \xrightarrow{\lambda} D'}$$

$$\frac{C \xrightarrow{a_*} C' \quad D \xrightarrow{\overline{a}_*} D'}{C|D \xrightarrow{\tau} C'|D'} \qquad \frac{C \xrightarrow{\surd} C' \quad D \xrightarrow{\surd} D'}{C|D \xrightarrow{\surd} C'|D'} \qquad \frac{C \xrightarrow{\lambda} C' \quad \lambda \neq \surd}{C|D \xrightarrow{\lambda} C'|D}$$

$$\frac{C \xrightarrow{\lambda} C' \quad \lambda \notin M \cup \overline{M}}{C \setminus M \xrightarrow{\lambda} C' \setminus M} \qquad C^* \xrightarrow{\surd} \mathbf{0} \qquad \frac{C \xrightarrow{\lambda} C'}{C^* \xrightarrow{\lambda} C';C^*}$$

**Table 1.** Semantic rules for contracts (symmetric rules omitted).

The operational semantics is rather standard for process calculi with sequential composition, where the $\surd$ label is used to explicitly denote completion. The unique relevant remark is that synchronization within a contract is permitted only on local names $a_*$; the synchronization on global names between different contracts will be considered in the next calculus used to model the composition of contracts.

In the remainder of the paper we use the following notations: $C \xrightarrow{\lambda}$ to mean that there exists $C'$ such that $C \xrightarrow{\lambda} C'$ and, given a sequence of labels $w = \lambda_1\lambda_2 \cdots \lambda_{n-1}\lambda_n$ (possibly empty, i.e., $w = \varepsilon$), we use $C \xrightarrow{w} C'$ to denote the sequence of transitions $C \xrightarrow{\lambda_1} C_1 \xrightarrow{\lambda_2} \cdots \xrightarrow{\lambda_{n-1}} C_{n-1} \xrightarrow{\lambda_n} C'$ (in case of $w = \varepsilon$ we have $C' = C$, i.e., $C \xrightarrow{\varepsilon} C$).

The main results reported in this paper are consequences of a property of contracts that we call *output persistency*. This property states that once a contract decides to execute an output, its actual execution is mandatory in order to successfully complete the execution of the contract. In order to formally prove this property we need to formalize two (easy to prove) preliminary lemmata.

**Lemma 1.** *Let $C \in \mathcal{P}_{con}$ s.t. $C \xrightarrow{\overline{a}}$ and $C \xrightarrow{\lambda} C'$, then $\lambda \neq \surd$.*

**Lemma 2.** *Let $C \in \mathcal{P}_{con}$ s.t. $C \xrightarrow{\overline{a}}$ and $C \xrightarrow{\beta} C'$ with $\beta \neq \overline{a}$, then $C' \xrightarrow{\overline{a}}$.*

*Output persistency* is a trivial consequence of these two lemmata.

**Proposition 1. (Output persistency)** *Let $C \in \mathcal{P}_{con}$ be a contract such that $C \xrightarrow{w} C' \xrightarrow{\overline{a}}$. We have that, for every $C''$ such that $C' \xrightarrow{w'} C''$ and $C'' \xrightarrow{\surd}$, the string $w'$ must include $\overline{a}$.*

### 2.2 Composing Contracts

We now introduce the calculus for modeling systems of composed contracts. This is an extension of the previous calculus; the basic terms are contracts under execution denoted with $[C]$.

Besides the parallel composition operator $\|$, we consider also restriction $\setminus\!\!\setminus$ in order to model the possibility to open local channels of interaction among contracts. This operator of restriction distinguishes between input and output operations; this allows us, e.g., to model a system composed of two contracts $C_1$ and $C_2$ such that channel $a$ is used for communications from $C_1$ to $C_2$ and channel $b$ is used for communications along the opposite directions:

$$([C_1]\setminus\!\!\setminus\{a, \overline{b}\}) \parallel ([C_2]\setminus\!\!\setminus\{\overline{a}, b\})$$

**Definition 2. (Contract composition)** *The syntax of contract compositions is defined by the following grammar*

$$P \quad ::= \quad [C] \quad | \quad P\|P \quad | \quad P\setminus\!\!\setminus L$$

*where $L \subseteq \mathcal{A}$.*

In the following we will sometimes omit parenthesis"[ ]" when writing contract compositions and we will call *system* a composition of contracts.

The operational semantics of systems is defined by the rules in Table 2 (plus the omitted symmetric rules).

$$\frac{C \xrightarrow{\lambda} C' \quad \lambda \notin \mathcal{A}_{loc}}{[C] \xrightarrow{\lambda} [C']} \qquad \frac{P \xrightarrow{\lambda} P' \quad \lambda \neq \surd}{P\|Q \xrightarrow{\lambda} P'\|Q}$$

$$\frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\overline{a}} Q'}{P\|Q \xrightarrow{\tau} P'\|Q'} \qquad \frac{P \xrightarrow{\surd} P' \quad Q \xrightarrow{\surd} Q'}{P\|Q \xrightarrow{\surd} P'\|Q'} \qquad \frac{P \xrightarrow{\lambda} P' \quad \lambda \notin L}{P\setminus\!\!\setminus L \xrightarrow{\lambda} P'\setminus\!\!\setminus L}$$

**Table 2.** Semantic rules for contract compositions (symmetric rules omitted).

Note that, due to the absence of internal communication of actions of $\mathcal{A}$ inside contracts, when we apply external restriction directly to a contract $C$, i.e. we consider $[C]\backslash\backslash L$ for some $L \subseteq \mathcal{A}$, we obtain a transition system isomorphic to that of the contract $C\{\mathbf{0}/\beta|\beta \in L\} \setminus \mathcal{N}_{loc}$ or, equivalently, to that of $[C\{\mathbf{0}/\beta|\beta \in L\}]$, where $C\{\mathbf{0}/\beta|\beta \in L\}$ represents the syntactical substitution of $\mathbf{0}$ for every occurence of any subterm $\beta$ such that $\beta \in L$.

We are now ready to define our notion of correct composition of contracts. Intuitively, a system composed of contracts is correct if all possible computations may guarantee completion; this means that the system is both deadlock and livelock free (there could be an infinite computation, but given any possible prefix of this infinite computation, it can be extended to reach a successfully completed computation).

**Definition 3. (Correct contract composition)** *A system $P$ is a correct contract composition, denoted $P\downarrow$, if for every $P'$ such that $P \xrightarrow{\tau}{}^{*} P'$ there exists $P''$ such that $P' \xrightarrow{\tau}{}^{*} P'' \xrightarrow{\surd}$.*

As examples of correct contract compositions, you can consider $C_1\|C_2$ with

$$\begin{array}{ll} C_1 = a + b & C_2 = (\tau;\overline{a}) + (\tau;\overline{b}) \\ C_1 = a; b & C_2 = \tau;\overline{a};\tau;\overline{b} \\ C_1 = a + b + c & C_2 = (\tau;\overline{a}) + (\tau;\overline{b}) \\ C_1 = (a;b) + (b;a) & C_2 = (\tau;\overline{a}) \,|\, (\tau;\overline{b}) \\ C_1 = (a;\tau;\overline{b})^* & C_2 = \tau;\overline{a};(b;\tau;\overline{a})^*;b \end{array}$$

## 3 Contract Refinement

In this Section we introduce our theory of contracts. The basic idea is to have a notion of refinement of contracts such that, given a system composed of the contracts $C_1, \cdots, C_n$, we can replace each contract $C_i$ with one of its refinements $C_i'$ without breaking the correctness of the system.

Some simple example of refinement follows. Consider the correct system $C_1\|C_2$ with

$$C_1 = a + b \qquad C_2 = (\tau;\overline{a}) + (\tau;\overline{b})$$

We can replace $C_1$ with $C_1' = a+b+c$ or $C_2$ with $C_2' = \tau;\overline{a}$ without breaking the correctness of the system. This example shows a first important intuition: a contract could be replaced with another one that has more external nondeterminism and/or less internal nondeterminism.

Consider now

$$D_1 = a + b + c \qquad D_2 = (\tau;\overline{a}) + (\tau;\overline{b})$$

where we can refine $D_1$ with $D_1' = a + b + d$. Clearly, this refinement does not hold in general because we could have another correct system

$$D_1 = a + b + c \qquad D_2' = (\tau;\overline{a}) + (\tau;\overline{b}) + (\tau;\overline{c})$$

where such a refinement does not hold. This second example shows that refinement is influenced by the potential actions that could be executed by the other contracts in the system. Indeed, $D'_1$ is not a correct substitute for $D_1$ because $D'_2$ has the possibility to produce $\overline{c}$.

Based on this intuition, we parameterize our notion of subcontract relation $C' \leq_{I,O} C$ on the set $I$ of inputs, and the set $O$ of outputs, that could be potentilly executed by the other contracts in the system. We will see that $D'_1 \leq_{\emptyset,\{a,b\}} D_1$ but $D'_1 \not\leq_{\emptyset,\{a,b,c\}} D_1$.

### 3.1 Subcontract pre-order

We first define two auxiliary functions that extract from contracts and systems the set of names used in input and output actions, respectively.

**Definition 4. (Input and Output sets)** *Given the contract $C \in \mathcal{P}_{con}$, we define $I(C)$ (resp. $O(C)$) as the subset of $\mathcal{N}$ of the potential input (resp. ouput) actions of $C$. Formally, we define $I(C)$ as follows ($O(C)$ is defined similarly):*

$$I(\mathbf{0}) = I(\mathbf{1}) = I(\tau) = I(\tau;\overline{\alpha}) = \emptyset \qquad I(\alpha) = \textit{if } \alpha{\in}\mathcal{N} \textit{ then } \{\alpha\} \textit{ else } \emptyset$$
$$I(C;C') = I(C{+}C') = I(C|C') = I(C){\cup}I(C') \qquad I(C\setminus M) = I(C^*) = I(C)$$

*Note that the set $M$ in $C \setminus M$ does not influence $I(C \setminus M)$ because it contains only local names outside $\mathcal{N}$. Given the system $P$, we define $I(P)$ (resp. $O(P)$) as the subset of $\mathcal{N}$ of the potential input (resp. ouput) actions of $P$. Formally, we define $I(P)$ as follows ($O(P)$ is defined similarly):*

$$I([C]) = I(C) \qquad I(P\|P') = I(P) \cup I(P') \qquad I(P\backslash\backslash L) = I(P) - L$$

We are now ready to define the notion of subcontract pre-order $C'_i \leq_{I,O} C_i$ in which the substitutability of contract $C_i$ with $C'_i$ is parameterized in the possible input and output actions $I$ and $O$ of the other contracts in the considered system.

More precisely, we consider a correct system $C_1\backslash\backslash I_1 \cup \overline{O}_1 \parallel \ldots \parallel C_n\backslash\backslash I_n \cup \overline{O}_n$ composed of the contracts $C_1, \cdots, C_n$ following a particular name discipline: the names in $I_i$ (resp. $O_i$) cannot be used in input (resp. output) actions by the contract $C_i$. This discipline is guaranteed restricting each contract $C_i$ on the set of actions $I_i \cup \overline{O_i}$. In this particular system, we want to be able to subtitute each of the contract $C_i$ with any contract $C'_i$ such that $C'_i \leq_{I,O} C_i$ where $I$ and $O$ comprise the possible input and output actions that can be executed by the other contracts $C_j$ with $j \neq i$. This last condition can be ensured imposing that

$$\Big(\bigcup_{j\neq i} I(C_j){-}I_j\Big){-}O_i \subseteq I \quad \wedge \quad \Big(\bigcup_{j\neq i} O(C_j){-}O_j\Big){-}I_i \subseteq O$$

This kind of formula is considered in the subsequent definition that formalizes the notion of subcontract pre-order family.

**Definition 5. (Subcontract pre-order family)** *A family* $\{\leq_{I,O} \mid I, O \subseteq \mathcal{N}\}$ *of pre-orders over* $\mathcal{P}_{con}$ *is a subcontract pre-order family if, for any* $n \geq 1$, *contracts* $C_1, \ldots, C_n \in \mathcal{P}_{con}$ *and* $C'_1, \ldots, C'_n \in \mathcal{P}_{con}$ *and input and output names* $I_1, \ldots, I_n \subseteq \mathcal{N}$ *and* $O_1, \ldots, O_n \subseteq \mathcal{N}$, *we have*

$$
\begin{aligned}
&(C_1 \setminus\!\!\setminus I_1 \cup \overline{O}_1 \parallel \ldots \parallel C_n \setminus\!\!\setminus I_n \cup \overline{O}_n)\!\downarrow \;\wedge \\
&\forall i.\, C'_i \leq_{I'_i, O'_i} C_i \;\wedge\; (\textstyle\bigcup_{j \neq i} I(C_j) - I_j) - O_i \subseteq I'_i \;\wedge\; (\textstyle\bigcup_{j \neq i} O(C_j) - O_j) - I_i \subseteq O'_i \\
&\qquad\qquad \Rightarrow \quad (C'_1 \setminus\!\!\setminus I_1 \cup \overline{O}_1 \parallel \ldots \parallel C'_n \setminus\!\!\setminus I_n \cup \overline{O}_n)\!\downarrow
\end{aligned}
$$

We will prove that there exists a maximal subcontract pre-order family; this is a direct consequence of the output persistency property. In fact, if we consider mixed choice it is easy to prove that there exists no maximal subcontract pre-order family. Consider, e.g., the trivially correct system $C_1 \| C_2$ with $C_1 = a$ and $C_2 = \overline{a}$; we could have two subcontract pre-order families $\leq^1$ and $\leq^2$ such that

$$\leq^1_{\{a,c\}\{a,c\}} \text{ with } a + c.\mathbf{0} \leq^1_{\{a,c\}\{a,c\}} a \text{ and } \overline{a} + c.\mathbf{0} \leq^1_{\{a,c\}\{a,c\}} \overline{a}$$

and

$$\leq^2_{\{a,c\}\{a,c\}} \text{ with } a + \overline{c}.\mathbf{0} \leq^2_{\{a,c\}\{a,c\}} a \text{ and } \overline{a} + \overline{c}.\mathbf{0} \leq^2_{\{a,c\}\{a,c\}} \overline{a}$$

but there exists no subcontract pre-order family that includes both the refinements of $\leq^1_{\{a,c\}\{a,c\}}$ and $\leq^2_{\{a,c\}\{a,c\}}$.

We will show that the maximal subcontract pre-order family can be achieved defining a coarser form of refinement in which, given any system composed of a set of contracts, refinement is applied to one contract only (thus leaving the other unchanged). We call this form of refinement *singular subcontract pre-order*.

Intuitively a family of pre-orders $\{\leq_{I,O} \mid I, O \subseteq \mathcal{N}\}$ is a singular subcontract pre-order family whenever the correctness of systems is preserved by refining just one of the contracts. More precisely, for any $n \geq 1$, $C_1, \ldots, C_n \in \mathcal{P}_{con}$, $I_1, \ldots, I_n \subseteq \mathcal{N}$, $O_1, \ldots, O_n \subseteq \mathcal{N}$, $1 \leq i \leq n$ and $C'_i \in \mathcal{P}_{con}$ we require

$$
\begin{aligned}
&(C_1 \setminus\!\!\setminus I_1 \cup \overline{O}_1 \parallel \ldots \parallel C_i \setminus\!\!\setminus I_i \cup \overline{O}_i \parallel \ldots \parallel C_n \setminus\!\!\setminus I_n \cup \overline{O}_n)\!\downarrow \;\wedge \\
&C'_i \leq_{I,O} C_i \;\wedge\; (\textstyle\bigcup_{j \neq i} I(C_j) - I_j) - O_i \subseteq I \;\wedge\; (\textstyle\bigcup_{j \neq i} O(C_j) - O_j) - I_i \subseteq O \\
&\qquad\qquad \Rightarrow \quad (C_1 \setminus\!\!\setminus I_1 \cup \overline{O}_1 \parallel \ldots \parallel C'_i \setminus\!\!\setminus I_i \cup \overline{O}_i \parallel \ldots \parallel C_n \setminus\!\!\setminus I_n \cup \overline{O}_n)\!\downarrow
\end{aligned}
$$

By exploiting commutativity and associativity of parallel composition, and the fact that the internal behavior of $C_1 \setminus\!\!\setminus I_1 \cup \overline{O}_1 \parallel \ldots \parallel C_n \setminus\!\!\setminus I_n \cup \overline{O}_n$ is the same as that of $C_1 \| ((C_2\{\mathbf{0}/\beta | \beta \in I_2 \cup \overline{O}_2\} \| \ldots \| C_n\{\mathbf{0}/\beta | \beta \in I_n \cup \overline{O}_n\}) \setminus\!\!\setminus O_1 \cup \overline{I}_1)$ we can group the contracts which are not being refined and denote them with a generic term $P$ taken from $\mathcal{P}_{conpar}$, the set of the systems of the form $(C_1 \| \ldots \| C_n) \setminus\!\!\setminus I \cup \overline{O}$, with $C_i \in \mathcal{P}_{con}$ for all $i \in \{1, \ldots, n\}$ and $I, O \subseteq \mathcal{N}$. Moreover we note that, given $P = (C_1 \| \ldots \| C_n) \setminus\!\!\setminus I \cup \overline{O} \in \mathcal{P}_{conpar}$, we have $I(P) = (\bigcup_{1 \leq i \leq n} I([C_i])) - I$ and $O(P) = (\bigcup_{1 \leq i \leq n} O([C_i])) - O$.

**Definition 6. (Singular subcontract pre-order family)** *A family of pre-orders* $\{\leq_{I,O} \mid I, O \subseteq \mathcal{N}\}$ *is a singular subcontract pre-order family if, for any* $C, C' \in \mathcal{P}_{con}$, $P \in \mathcal{P}_{conpar}$ *we have*

$$(C \| P)\!\downarrow \;\wedge\; C' \leq_{I,O} C \;\wedge\; I(P) \subseteq I \;\wedge\; O(P) \subseteq O \quad \Rightarrow \quad (C' \| P)\!\downarrow$$

In order to prove the existence of the maximal subcontract pre-order family, we prove that every family of pre-orders that is a subcontract family is also a singular subcontract family (Theorem 1). Moreover we show that there exists a maximal singular subcontract family and we prove that it also a subcontract family (Theorem 2).

**Theorem 1.** *If a family of pre-orders $\{\leq_{I,O} \mid I, O \subseteq \mathcal{N}\}$ is a subcontract pre-order family then it is also a singular subcontract pre-order family.*

*Proof.* *Suppose that $\{\leq_{I,O} \mid I, O \subseteq \mathcal{N}\}$ is a subcontract pre-order family. Consider $n \geq 1$, $P \in \mathcal{P}_{conpar}$, $C, C' \in \mathcal{P}_{con}$. From $(C \| P)\!\downarrow$ and $C' \leq_{I,O} C$, where $I(P) \subseteq I$ and $O(P) \subseteq O$, we can derive $(C' \| P)\!\downarrow$ by just taking in the definition of subcontract pre-order family, $C_1 = C$, $C'_1 = C'$, $C_2 \dots C_n$ and $I_1$ and $O_1$ to be such that $P = (C_2 \| \dots \| C_n) \backslash\!\backslash O_1 \cup \overline{I}_1$; $I_2 \dots I_n$ and $O_2 \dots O_n$ to be the emptyset; and finally $C'_i$ to be $C_i$ for every $i \geq 2$ (since $\leq_{I,O}$ are pre-orders we have $C \leq_{I,O} C$ for every $I, O$ and $C$).*

From the simple structure of their definition we can easily deduce that singular subcontract pre-order families have maximum, i.e. there exists a singular subcontract pre-order family such that every pre-order $\leq_{I,O}$ in the family includes all the corresponding pre-order $\leq_{I,O}$ of the other singular subcontract pre-order families. In the following we let $\mathcal{P}_{conpar,I,O}$ denote the subset of processes of $\mathcal{P}_{conpar}$ such that $I(P) \subseteq I$ and $O(P) \subseteq O$.

**Definition 7. (Input-Output Subcontract relation)** *A contract $C'$ is a subcontract of a contract $C$ with respect to a set of input channel names $I \subseteq \mathcal{N}$ and output channel names $O \subseteq \mathcal{N}$, denoted $C' \preceq_{I,O} C$, if*

$$\forall P \in \mathcal{P}_{conpar,I,O}. \quad (C \| P)\!\downarrow \quad \Rightarrow \quad (C' \| P)\!\downarrow$$

It is trivial to verify that the family of pre-orders $\{\preceq_{I,O} \mid I, O \subseteq \mathcal{N}\}$ is a singular subcontract pre-order family and is the maximum of all the singular subcontract pre-order families.

The following Proposition states an intuitive contravariant property: given $\preceq_{I',O'}$, and the greater sets $I$ and $O$ (i.e. $I' \subseteq I$ and $O' \subseteq O$) we obtain a smaller pre-order $\preceq_{I,O}$ (i.e. $\preceq_{I,O} \subseteq \preceq_{I',O'}$). This follows from the fact that extending the sets of input and output actions means considering a greater set of discriminating contexts.

**Proposition 2.** *Let $C, C' \in \mathcal{P}_{con}$ be two contracts, $I, I' \subseteq \mathcal{N}$ be two sets of input channel names such that $I' \subseteq I$ and $O, O' \subseteq \mathcal{N}$ be two sets of output channel names such that $O' \subseteq O$. We have:*

$$C' \preceq_{I,O} C \quad \Rightarrow \quad C' \preceq_{I',O'} C$$

The following Propositition states that a subcontract is still a subcontract even if we restrict its actions in order to consider only the inputs and outputs already available in the supercontract. The result about the possibility to restrict the outputs will be extensively used in the remainder of the paper.

**Proposition 3.** *Let $C, C' \in \mathcal{P}_{con}$ be contracts and $I, O \subseteq \mathcal{N}$ be sets of input and output names. We have*

$$\begin{array}{lcl} C' \preceq_{I,O} C & \Rightarrow & C' \backslash\!\backslash \overline{(I(C') - I(C))} \preceq_{I,O} C \\ C' \preceq_{I,O} C & \Rightarrow & C' \backslash\!\backslash \overline{(O(C') - O(C))} \preceq_{I,O} C \end{array}$$

*Proof.* We discuss the result concerned with restriction of outputs (the proof for the restriction of inputs is symmetrical). Let $C' \preceq_{I,O} C$. Given any $P \in \mathcal{P}_{conpar,I,O}$ such that $(C \| P) \downarrow$, we will show that $(C' \backslash\!\backslash \overline{(O(C') - O(C))} \parallel P) \downarrow$. We first observe that $(C \parallel P \backslash\!\backslash (O(C') - O(C))) \downarrow$. Since $C' \preceq_{I,O} C$, we derive $(C' \parallel P \backslash\!\backslash (O(C') - O(C))) \downarrow$. As a consequence $(C' \backslash\!\backslash \overline{(O(C') - O(C))} \parallel P \backslash\!\backslash (O(C') - O(C))) \downarrow$. We can conclude $(C' \backslash\!\backslash \overline{(O(C') - O(C))} \parallel P) \downarrow$.

All the results discussed so far do not depend on the output persistency property. The first relevant result depending on this peculiarity is reported in the following Proposition. It states that if we substitute a contract with one of its subcontract, the latter cannot activate outputs that were not included in the potential outputs of the supercontract.

**Proposition 4.** *Let $C, C' \in \mathcal{P}_{con}$ be contracts and $I, O \subseteq \mathcal{N}$ be sets of input and output names. If $C' \preceq_{I,O} C$ we have that, for every $P \in \mathcal{P}_{conpar,I,O}$ such that $(C \| P) \downarrow$,*

$$(C' \| P) \xrightarrow{\tau}{}^{*} (C'_{der} \| P_{der}) \quad \Rightarrow \quad \forall\, a \in O(C') - O(C).\, C'_{der} \overset{\overline{a}}{\not\longrightarrow}$$

*Proof.* We proceed by contradiction. Suppose that there exist $C'_{der}, P_{der}$ such that $(C' \| P) \xrightarrow{\tau}{}^{*} (C'_{der} \| P_{der})$ and $C'_{der} \xrightarrow{\overline{a}}$ for some $a \in O(C') - O(C)$. We further suppose (without loss of generality) that such a path is minimal, i.e. no intermediate state $(C'_{der2} \| P_{der2})$ is traversed, such that $C'_{der2} \xrightarrow{\overline{a}}$ for some $a \in O(C') - O(C)$. This implies that the same path must be performable by $(C' \backslash\!\backslash \overline{(O(C') - O(C))} \parallel P)$, thus reaching the state $(C'_{der} \backslash\!\backslash \overline{(O(C') - O(C))} \parallel P_{der})$. However, since in the state $C'_{der}$ of contract $C'$ we have $C'_{der} \xrightarrow{\overline{a}}$ for some $a \in O(C') - O(C)$ and the execution of $\overline{a}$ is disallowed by restriction, due to output persistency, the contract will never be able to reach success (no matter what contracts in $P$ will do). Therefore $(C' \backslash\!\backslash \overline{(O(C') - O(C))} \parallel P) \not\downarrow$ and (due to Proposition 3) we reached a contradiction.

The following Proposition permits to conclude that the set of potential inputs of the other contracts in the system is an information that does not influence the subcontract relation.

**Proposition 5.** *Let $C \in \mathcal{P}_{con}$ be contracts, $O \subseteq \mathcal{N}$ be a set of output names and $I, I' \subseteq \mathcal{N}$ be two sets of input names such that $O(C) \subseteq I, I'$. We have that for every contract $C' \in \mathcal{P}_{con}$,*

$$C' \preceq_{I,O} C \quad \Longleftrightarrow \quad C' \preceq_{I',O} C$$

*Proof.* *Let us suppose $C' \preceq_{I',O} C$ (the opposite direction is symmetric). Given any $P \in \mathcal{P}_{conpar,I,O}$ such that $(C\|P)\downarrow$, we will show that $(C'\|P)\downarrow$. We first observe that $(C \parallel P\backslash\backslash(I - O(C)))\downarrow$. Since $C' \preceq_{I',O} C$ and $O(C) \subseteq I'$, we derive $(C' \parallel P\backslash\backslash(I - O(C)))\downarrow$. Due to Proposition 4 we have that $(C' \parallel P\backslash\backslash(I - O(C)))$ can never reach by $\tau$ transitions a state where outputs in $O(C') - O(C)$ are executable by some derivative of $C'$, so we conclude $(C' \parallel P)\downarrow$.*

We are now in place to prove the main result of this paper, i.e. that the *Input-Output Subcontract relation* defined in the Definition 7 is also a subcontract pre-order family.

**Theorem 2.** *The family of pre-orders $\{\preceq_{I,O} \mid I, O \subseteq \mathcal{N}\}$ is a subcontract pre-order family.*

*Proof.* *Consider $n \geq 1$, $C_1, \ldots, C_n \in \mathcal{P}_{con}$, $C'_1, \ldots, C'_n \in \mathcal{P}_{con}$, $I_1, \ldots, I_n \subseteq \mathcal{N}$ and $O_1, \ldots, O_n \subseteq \mathcal{N}$. For any $i$ we let $P_i = C_i\backslash\backslash I_i \cup \overline{O_i}$ and $P'_i = C'_i\backslash\backslash I_i \cup \overline{O}_i$. If $(P_1\| \ldots \|P_n)\downarrow$ and for all $i$ we have that $C'_i \preceq_{I'_i,O'_i} C_i$, with $I'_i$ and $O'_i$ satisfying the constraint on names as specified in Definition 5, we can derive $(P'_1\| \ldots \|P'_n)\downarrow$ as follows. For every $i$ from 1 to $n$ we show that*

$$(P'_1\backslash\backslash\overline{(O(C'_1) - O(C_1))}\| \ldots \|P_i\| \ldots \|P'_n\backslash\backslash\overline{(O(C'_n) - O(C_n))})\downarrow$$

*by multiply applying the definition of singular subcontract pre-order family to any $C_j$ with $j \neq i$. For instance if $i$ is 1, from $(P_1\| \ldots \|P_n)\downarrow$ we derive*

$$(P_1\|P'_2\backslash\backslash\overline{(O(C'_2) - O(C_2))}\|P_3\| \ldots \|P_n)\downarrow$$

*by applying the definition of singular subcontract pre-order to refine $C_2$ and by using Proposition 3. We then use this intermadiate result to re-apply the definition of singular subcontract pre-order family for refining $C_3$ and we derive*

$$(P_1\|P'_2\backslash\backslash\overline{(O(P'_2) - O(P_2))}\|P'_3\backslash\backslash\overline{(O(C'_3) - O(C_3))}\|P_4\| \ldots \|P_n)\downarrow$$

*We proceed in this way until we yield*

$$(P_1\|P'_2\backslash\backslash\overline{(O(C'_2) - O(C_2))}\| \ldots \|P'_n\backslash\backslash\overline{(O(C'_n) - O(C_n))})\downarrow$$

*For $i \in \{2 \ldots n\}$ we proceed in a similar way to obtain*

$$(P'_1\backslash\backslash\overline{(O(C'_1) - O(C_1))}\| \ldots \|P_i\| \ldots \|P'_n\backslash\backslash\overline{(O(C'_n) - O(C_n))})\downarrow$$

*We conclude the proof as follows. For any $i$, since $C'_i \preceq_{I,O} C_i$, by Proposition 4 we have that $(P'_1\backslash\backslash\overline{(O(C'_1) - O(C_1))}\| \ldots \|P'_i\| \ldots \|P'_n\backslash\backslash\overline{(O(C'_n) - O(C_n))})$ can never reach by $\tau$ transitions a state where outputs in $O(C'_1) - O(C_1)$ are executable by the derivative $C'_{i,der}$ of $C'_i$ that is syntactically included in the derivative $P'_{i,der}$ of $P'_i$. If now we consider the behavior of*

$$(P'_1\backslash\backslash\overline{(O(C'_1) - O(C_1))}\| \ldots \|P'_n\backslash\backslash\overline{(O(C'_n) - O(C_n))})$$

*we derive that, for any $i$, we cannot reach by $\tau$ transitions a state*

$$(P'_{1,der}\backslash\backslash\overline{(O(C'_1)-O(C_1))}\| \dots \|(P'_{i,der}\backslash\backslash\overline{(O(C'_i)-O(C_i))}\| \dots \|P'_{n,der}\backslash\backslash\overline{(O(C'_n)-O(C_n))})$$

*where $C'_{i,der}$ (the derivative of $C'_i$ syntactically included in $P'_{i,der}$) can execute outputs in $O(C'_i)-O(C_i)$. Hence the presence of the restriction operators does not affect the internal behavior of $(P'_1\backslash\backslash\overline{(O(C'_1)-O(C_1))}\| \dots \|P'_n\backslash\backslash\overline{(O(C'_n)-O(C_n))})$ with respect to $(P'_1\| \dots \|P'_n)$. Therefore, we can finally derive $(P'_1\| \dots \|P'_n)\downarrow$ from $(P'_1\backslash\backslash\overline{(O(C'_1)-O(C_1))}\| \dots \|P'_n\backslash\backslash\overline{(O(C'_n)-O(C_n))})\downarrow$, that is obtained by furher applying the definition of singular subcontract pre-order to refine $C_i$ in any of the i-indexed statement in the first part of the proof.*

This last Theorem proves that the maximal singular subcontract pre-order family is also a subcontract preorder family; since we proved that every subcontract preorder family is also a singular subcontract pre-order family (see Theorem 1), we can conclude that there exists a maximal subcontract pre-order family and it corresponds to the family $\{\preceq_{I,O} | I, O \subseteq \mathcal{N}\}$.

Moreover, the Proposition 5 permits to abstract away from the index $I$ of $\preceq_{I,O}$ assuming always $I = \mathcal{N}$; formally, let $\preceq_O = \preceq_{\mathcal{N},\mathcal{O}}$. Similarly, we use $\mathcal{P}_{conpar,O}$ to denote the set of processes $\mathcal{P}_{conpar,\mathcal{N},\mathcal{O}}$. Hence, we can characterize the maximal subcontract pre-order family with the following *subcontract relation.*

**Definition 8. (Subcontract relation)** *A contract $C'$ is a subcontract of a contract $C$ with respect to a set of output channel names $O \subseteq \mathcal{N}$, denoted $C' \preceq_O C$, if*

$$\forall P \in \mathcal{P}_{conpar,O}. \quad (C\|P)\downarrow \quad \Rightarrow \quad (C'\|P)\downarrow$$

The remainder of this Section is devoted to the definition of an actual way for proving that two contracts are in subcontract relation. This is achieved resorting to the theory of *should-testing* [RV05]. The main difference of should-testing with respect to the standard must-testing [DH84] is that fairness is taken into account; an (unfair) infinite computation that never gives rise to success is observed in the standard must-testing scenario, while this is not the case in the should-testing scenario. The formal definition of should-testing is reported in the proof of Theorem 3.

We need a preliminary result that essentially proves that $C' \preceq_O C$ if and only if $C'\backslash\backslash\mathcal{N}-O \preceq_{\mathcal{N}} C\backslash\backslash\mathcal{N}-O$.

**Lemma 3.** *Let $C, C'$ be two contracts and $O \subseteq \mathcal{N}$ be a set of output names. We have $C' \preceq_O C$ iff*

$$\forall P \in \mathcal{P}_{conpar}. \quad (C\backslash\backslash\mathcal{N}-O \,\|\, P)\downarrow \quad \Rightarrow \quad (C'\backslash\backslash\mathcal{N}-O \,\|\, P)\downarrow$$

*Proof.* *Given $P \in \mathcal{P}_{conpar}$, we have $(C\backslash\backslash\mathcal{N}-O \,\|\, P)\downarrow \iff (C\backslash\backslash\mathcal{N}-O \,\|\, P\backslash\backslash\overline{\mathcal{N}-O})\downarrow \iff (C \,\|\, P\backslash\backslash\overline{\mathcal{N}-O})\downarrow$ and $(C'\backslash\backslash\mathcal{N}-O \,\|\, P)\downarrow \iff (C'\backslash\backslash\mathcal{N}-O \,\|\, P\backslash\backslash\overline{\mathcal{N}-O})\downarrow \iff (C' \,\|\, P\backslash\backslash\overline{\mathcal{N}-O})\downarrow$. In the particular case of $P \in \mathcal{P}_{conpar,O}$ we have that $P\backslash\backslash\overline{\mathcal{N}-O}$ is isomorphic to $P$.*

In the following we denote with $\preceq_{test}$ the *should-testing* preorder defined in [RV05] where we consider the set of actions used by terms as being $\mathcal{L}$ (i.e. $\surd$ is included in the set of actions of terms under testing as any other action). We denote here with $\surd'$ the special action for the success of the test (denoted by $\surd$ in [RV05]).

In order to resort to the theory defined in [RV05], we define a normal form for contracts of our calculus that corresponds to terms of the language in [RV05]. The normal form of the system $P$ (denoted with $\mathcal{NF}(\mathcal{P})$) is defined as follows, by using the operator $rec_X\theta$ (defined in [RV05]) that represents the value of $X$ in the solution of the minimum fixpoint of the finite set of equations $\theta$,

$$\begin{array}{ll} \mathcal{NF}(\mathcal{P}) = rec_{X_1}\theta & \text{where } \theta \text{ is the set of equations} \\ \quad X_i = \sum_j \lambda_{i,j}; X_{der(i,j)} & \end{array}$$

where, assuming to enumerate the states in the labeled transition system of $P$ starting from $X_1$, each variable $X_i$ corresponds to the $i$-th state of the labeled transition system of $P$, $\lambda_{i,j}$ is the label of the $j$-th outgoing transition from $X_i$, and $der(i,j)$ is the index of the state reached with the $j$-th outgoing transition from $X_i$. We assume empty sums to be equal to $\mathbf{0}$, i.e. if there are no outgoing transitions from $X_i$, we have $X_i = \mathbf{0}$.

**Theorem 3.** *Let $C, C'$ be two contracts and $O \subseteq \mathcal{N}$ be a set of output names. We have*

$$\mathcal{NF}(C' \backslash\!\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C \backslash\!\backslash (\mathcal{N}-O)) \quad \Rightarrow \quad C' \preceq_O C$$

*Proof.* *According to the definition of should-testing of [RV05], since*

$$\mathcal{NF}(C' \backslash\!\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C \backslash\!\backslash (\mathcal{N}-O))$$

*we have that, for every test $t$, if $\mathcal{NF}(C \backslash\!\backslash (\mathcal{N}-O))$ **shd** $t$, then also $\mathcal{NF}(C' \backslash\!\backslash (\mathcal{N}-O))$ **shd** $t$, where $Q$ **shd** $t$ iff*

$$\forall w \in \mathcal{L}^*, Q'. \qquad Q \|_{\mathcal{L}} t \xrightarrow{w} Q' \qquad \Rightarrow \qquad \exists v \in \mathcal{L}^*, Q'' : Q' \xrightarrow{v} Q'' \xrightarrow{\surd'}$$

*where $\|_{\mathcal{L}}$ is the CSP parallel operator: in $R \|_{\mathcal{L}} R'$ transitions of $R$ and $R'$ with the same label $\lambda$ (with $\lambda \neq \tau, \surd'$) are required to synchronize and yield a transition with label $\lambda$.*

*Let us now suppose $P \in P_{conpar}$ with $(C \backslash\!\backslash \mathcal{N} - O \| P) \downarrow$. We consider $t = \mathcal{NF}(P)\{\surd/\surd;\surd'\}$, i.e., the normal form of $P$ where we replace each occurrence of $\surd$ with the sequence $\surd;\surd'$. We denote with $\bar{t}$ the term obtained by turning each $a$ occurring in $t$ into $\overline{a}$, and each $\overline{a}$ into $a$. From the definition of **shd** it immediately follows that $\mathcal{NF}(C \backslash\!\backslash (\mathcal{N}-O))$ **shd** $\bar{t}$. Since $\mathcal{NF}(C' \backslash\!\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C \backslash\!\backslash (\mathcal{N}-O))$, we have that also $\mathcal{NF}(C' \backslash\!\backslash (\mathcal{N}-O))$ **shd** $\bar{t}$. From the definition of **shd** we can conclude that $(C' \backslash\!\backslash \mathcal{N} - O \| P) \downarrow$. The thesis directly follows from Lemma 3.*

Note that the opposite implication

$$C' \preceq_O C \Rightarrow \mathcal{NF}(C' \backslash\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C \backslash\backslash (\mathcal{N}-O))$$

does not hold in general. For example if we take contracts $C = a + a; c$ and $C' = b + b; c$ we have that $C' \preceq_O C$ (and $C \preceq_O C'$) for any $O$ (there is no contract $P$ such that $(C \| P) \downarrow$ or $(C' \| P) \downarrow$), but obviously $\mathcal{NF}(C' \backslash\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C \backslash\backslash (\mathcal{N}-O))$ (and $\mathcal{NF}(C \backslash\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C' \backslash\backslash (\mathcal{N}-O))$ ) does not hold for any $O$ that includes $\{a, b, c\}$. As another example, consider contracts $C = \tau; \mathbf{0} + a$ and $C' = \tau; \mathbf{0} + b$. We have that $C' \preceq_O C$ (and $C \preceq_O C'$) for any $O$ (there is no contract $P$ such that $(C \| P) \downarrow$ or $(C' \| P) \downarrow$), but $\mathcal{NF}(C' \backslash\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C \backslash\backslash (\mathcal{N}-O))$ (and $\mathcal{NF}(C \backslash\backslash (\mathcal{N}-O)) \preceq_{test} \mathcal{NF}(C' \backslash\backslash (\mathcal{N}-O))$ ) does not hold for any $O$ that includes $\{a, b\}$: this can be seen by considering the test $t = \sqrt{}' + b; \mathbf{0}$ ($t = \sqrt{}' + a; \mathbf{0}$).

Finally, we observe that the labeled transition system of each contract $C$ is finite state as we consider the Kleene-star repetition operator and not general recursion. This implies that also $\mathcal{NF}(C \backslash\backslash \mathcal{N}-O)$ is finite state for any $O$. In [RV05] it is proved that for finite state terms *should-testing* preorder is decidable and an actual verification algorithm is presented. This algorithm, in the light of our Theorem 3, represents a sound approach to prove also our subcontract relation.

## 4 Conclusion and Future Work

We have introduced a notion of subcontract relation useful for service oriented computing, where services are to be composed in such a way that deadlocks and livelocks are avoided. In order to be as much flexible as possible, we want to relate with our subcontract relation all those services that could safely replace their supercontracts. In the Introduction we have already discussed the practical impact of our notion of subcontract and we have compared our theory with the related literature.

Here, we simply add some comments about future plans. We intend to investigate the connection between the calculi used in this paper and calculi for service choreography such as those presented in [BGG+05] and [CHY07]. In particular, in [CHY07] an end-point calculus similar to our contract calculus is considered where external choices must be guarded on input operations. Moreover, a subtyping relation is used in [CHY07] to formalize similar aspects: the addition of input guarded branches in external choices is safe as well as the cancellation of output guarded branches in internal choices. Differently from [CHY07] we consider a weaker *should-testing* semantics instead of the more restrictive (bi)simulation approach of [CHY07]. This permits us, for instance, to abstract away from branching information that reveals not significant for contract composition.

## References


[BGG+05] Nadia Busi, Roberto Gorrieri, Claudio Guidi, Roberto Lucchi, and Gianluigi Zavattaro. Choreography and orchestration: A synergic approach for system design. In *ICSOC'05*, volume 3826 of LNCS, pages 228–240, 2005.

[CHY07] Marco Carbone, Kohei Honda, and Nabuko Yoshida. Structured Communication-Centred Programming for Web Services. In *ESOP'07*, volume to appear of LNCS, 2007.

[CCL+06] Samuele Carpineti, Giuseppe Castagna, Cosimo Laneve, and Luca Padovani. A Formal Account of Contracts for Web Services. In *WS-FM'06*, volume 4184 of LNCS, pages 148-162, 2006.

[CL06] Samuele Carpineti and Cosimo Laneve A Basic Contract Language for Web Services. In *ESOP'06*, volume 3924 of LNCS, pages 197–213, 2006.

[DH84] Rocco De Nicola and Matthew Hennessy, Testing Equivalences for Processes. *Theoretical Computer Science*, volume 34: 83–133, 1984.

[FHR+04] Cédric Fournet, C. A. R. Hoare, Sriram K. Rajamani, and Jakob Rehof. Stuck-Free Conformance. In *CAV'04*, volume 3114 of LNCS, pages 242–254, 2004.

[RV05] Arend Rensink and Walter Vogler. Fair testing. *CTIT Technical Report TR-CTIT-05-64*, Department of Computer Science, University of Twente, December 2005.

[Nes00] Uwe Nestmann. What is a "Good" Encoding of Guarded Choice?. *Information and Computation*, volume 156(1-2): 287-319, 2000.