Modal Contracts for Component-based Design

Gregor Goessler and Jean-Baptiste Raclet
INRIA Grenoble – Rhône-Alpes
France
Email: `firstname.lastname@inria.fr`

***Abstract*—Contracts and their composition may be used with different goals in a component-based design flow: on the one hand, a *component* contract describes the guarantees a component is able to give, depending on its environment. On the other hand, a *requirement contract*, or *aspect*, specifies a guarantee that must be ensured under some hypothesis. For component contracts we define a "best effort" composition satisfying the property of *independent implementability*. For aspects we define a composition based on modal conjunction, which is shown to be *sound*. Modal contracts allow keeping a larger design space than their non-modal version, and thus avoid prematurely ruling out possible implementations.**

***Index Terms*—Component-based design, contract, modalities, composition.**

I. Introduction

Contracts have first been introduced as a type system for classes [1]: a method guarantees some post-condition under the assumption that its pre-condition is satisfied. In the component-based programming community, contracts are increasingly focus of research as a means to achieve one of the main goals of the component paradigm, namely the deployment and reuse of components in different, a priori unknown contexts. As components may interact under various models of communication, the notion of contract has been generalized from pre- and post-conditions in the form of predicates to *behavioral interfaces* such as *interface automata* [2], allowing to reason about the temporal behavior of environments with which a component can be composed.

In contrast to a *specification* defining how a component *must* behave, contracts can be seen as implications, providing a guarantee depending on an assumption on the context. Accordingly, different semantics of contract composition are conceivable, with the two special cases of *conjunction of implications* yielding a *lazy* composition, and *implication of a conjunction* for an *eager* composition. The latter approach is adopted by [3], where the assumption of the composed contract is defined as the weakest assumption ensuring the conjunction of both guarantees. In the present work we choose the former approach: a component satisfying the composition of two contracts must satisfy each guarantee if and only if the corresponding assumption holds. This notion of composition is consistent with the component paradigm mentioned above, enabling the component to offer different guarantees depending on the context.

This work was funded by the European STREP COMBEST project number 215543.
Accepted at SEFM'09.

In a component-based design flow, contracts and their composition may be used with different goals in the design flow: on the one hand, a contract may be used to describe the guarantees a component is able to give, depending on its environment. We call this a *component contract*. Component contracts can be used top-down to successively refine an abstract component, or bottom-up to build a system from previously constructed components. On the other hand, a contract may specify a *requirement* as a guarantee that must be ensured under some hypothesis. We call this a requirement contract, or *aspect*. Aspects are usually implemented top-down. Therefore, contracts are an elegant way to combine bottom-up and top-down design. Although syntactically there is no difference between both kinds of contracts, the difference comes from the way they are composed.

For component contracts over disjoint components we define a "best effort" composition operation that is parametrized by an *interaction model* inspired by the BIP framework [4], [5]. The composition ensures each guarantee depending on the satisfaction of its assumption, provided that the guarantee is feasible under the specified interaction model. We show that this operation satisfies the property of *independent implementability*.

For aspects on the same component or sub-system we define a composition operation based on modal conjunction to ensure that the composition refines both contracts. This is motivated by the fact that different aspects express different requirements whose conjunctions is to be satisfied. The same operation serves to compose an aspect with a component. It is shown to be *sound* and, under some conditions, complete. To our knowledge this is the first work formalizing and allowing to effectively combine both types of contract composition.

Furthermore, we define a prioritized composition of aspects whose result refines the composed aspects, such that in case of inconsistencies among them, an aspect of higher priority "overrides" a lower-priority contract.

We define contracts in terms of *modal automata* [6] extending automata with a modality that indicates for each transition whether it *may* or *must* be implemented. This additional typing has the advantage of keeping a larger design space, whereas a premature choice of implementing or not a given transition would prematurely narrow the design space, and rule out possible implementations.

Related work. Modal specifications benefit from a well-established theory and a set of results that we build upon, in particular work on modal residuation [7]. A detailed discussion of benefits of modalities for interface theories, in particular for fitting together contracts over different action vocabularies, can be found in [8]. Since we introduce two distinct operations for composing contracts over disjoint interaction models and a common interaction model, respectively, we do not encounter this issue here.

Verification based on modal contracts in BIP is studied in [9], where a decomposition of contracts is used to define compositional refinement of component contracts. [10] discusses a contract-based design flow for a rich component framework. Aspects are formalized in terms of pairs (assumption, guarantee) of sets of traces. Conjunction of non-modal specifications has been introduced in [11] with the goal of enabling heterogeneous specifications mixing operational and logical parts.

This paper is organized as follows. Section II defines modal specifications. Section III introduces contracts as pairs of modal specifications. Furthermore, weak implication between modal specifications is introduced, and its properties are discussed. Using these results, Section IV defines several composition operations for contracts, and Section V concludes.

II. Modal Specifications

Automata enriched with modalities on transitions have been introduced in [6] twenty years ago (see [12] for a complete survey). Basically, modal specifications possess two types of transitions: may-transitions that are optional, as opposed to $must$-transitions that are obligatory.

Definition 1 (Modal specification)**.** *A* modal specification *is a tuple $\mathcal{S} = \langle Q, q^0, \Sigma, \Delta^m, \Delta^M \rangle$ where Q is a finite set of states, $q^0 \in Q$ is the unique initial state, Σ is a finite set of actions (or ports), and $\Delta^m, \Delta^M \subseteq Q \times \Sigma \times Q$ are respectively the set of may-transitions and of $must$-transitions. We require that Δ^m is deterministic (that is, $(q, a, q') \in \delta \wedge (q, a, q'') \in \delta \Longrightarrow q' = q''$) and that $\Delta^M \subseteq \Delta^m$.*

The condition $\Delta^M \subseteq \Delta^m$ naturally imposes that every required transition (i.e., in Δ^M) is also allowed (i.e., is also in Δ^m).

For short, we shall write or draw: $q \xrightarrow{a} q'$ when $(q, a, q') \in \Delta^M$; $q \overset{a}{\dashrightarrow} q'$ when $(q, a, q') \in \Delta^m \setminus \Delta^M$; $q \overset{a}{\nrightarrow}$ when $\forall q' : (q, a, q') \notin \Delta^m$.

Example 1 (adapted from [13])**.** *Consider a communication channel whose alphabet of actions includes* msg *for a sending request and two kinds of acknowledgment for transmission:* ack *in case of success and* nack *in case of failure. The upper modal specification in Fig. 1 specifies that every message sent must be acknowledged.*

Definition 2 (Behavior)**.** *We call* behavior *a modal specification $\langle Q, q^0, \Sigma, \Delta^m, \Delta^M \rangle$ where $\Delta^M = \Delta^m$.*

When composing specifications (several composition operations will be introduced later on), inconsistencies between

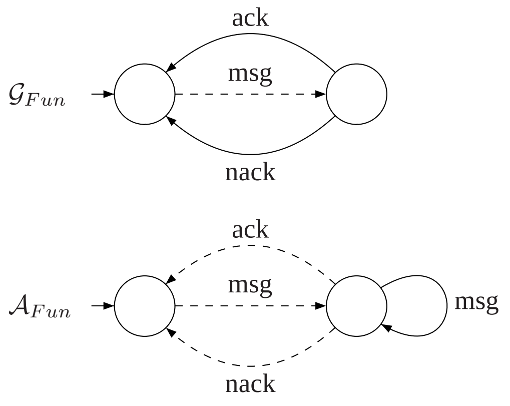


Fig. 1. Communication channel – functional aspect $\mathcal{C}_{Fun}$

the modalities may appear. We thus consider an extension of modal specification called *pseudo-modal specifications*, which possesses a distinct *inconsistent* state $\perp$.

Definition 3 (Pseudo-modal specification)**.** *A* pseudo-modal specification *is a tuple $p\mathcal{S} = \langle Q \cup \{\perp\}, q^0, \Sigma, \Delta^m, \Delta^M \rangle$ where $Q \cup \{\perp\}$ is a finite set of states with $\perp \notin Q$, $q^0 \in Q$ is the unique initial state and $\Delta^m, \Delta^M \subseteq Q \times \Sigma \times (Q \cup \{\perp\})$ are two transition relations such that Δ^m is deterministic and $\Delta^M \subseteq \Delta^m$.*

Modal specifications correspond to the subclass of pseudo-modal specifications for which $\perp$ is not a reachable state. As a consequence, the definitions below for pseudo-modal specifications also apply to modal specifications. For $q \in Q$, we denote:

- $may(q)$ the set of actions a for which there exists a state $q' \in Q \cup \{\perp\}$ with $(q, a, q') \in \Delta^m$;
- $must(q)$ the set of actions a for which there exists a state $q' \in Q \cup \{\perp\}$ with $(q, a, q') \in \Delta^M$;
- $mustnot(q)$ the set of actions $\Sigma \setminus may(q)$.

Definition 4 (Refinement)**.** *A pseudo-modal specification $p\mathcal{S}_1 = \langle Q_1 \cup \{\perp_1\}, q_1^0, \Sigma, \Delta_1^m, \Delta_1^M \rangle$ refines a pseudo-modal specification $p\mathcal{S}_2 = \langle Q_2 \cup \{\perp_2\}, q_2^0, \Sigma, \Delta_2^m, \Delta_2^M \rangle$, written $p\mathcal{S}_1 \preceq p\mathcal{S}_2$, if there exists a simulation relation $\theta \subseteq Q_1 \times Q_2$ such that: $(q_1^0, q_2^0) \in \theta$, and for all $(q_1, q_2) \in \theta$, the following holds:*

- *for every $(q_2, a, q_2') \in \Delta_2^M$ there exists $(q_1, a, q_1') \in \Delta_1^M$ and $(q_1', q_2') \in \theta$;*
- *for every $(q_1, a, q_1') \in \Delta_1^m$ there exists $(q_2, a, q_2') \in \Delta_2^m$ and $(q_1', q_2') \in \theta$.*

Remark that, since the simulation relation θ is defined over $Q_1 \times Q_2$, the inconsistent states $\perp_1$ and $\perp_2$ cannot appear in the pairs of states of θ. Moreover, for any pair $(q_1, q_2) \in \theta$ we have by definition:

$$may_1(q_1) \subseteq may_2(q_2)$$
$$must_1(q_1) \supseteq must_2(q_2)$$

Definition 5 (Model relation)**.** *A behavior $\mathcal{B}$ is a* model *of a* *pseudo-modal specification $p\mathcal{S}$, denoted $\mathcal{B} \models p\mathcal{S}$, if $\mathcal{B} \preceq p\mathcal{S}$. The set of models of $p\mathcal{S}$ is denoted by* $\mathsf{Mod}(p\mathcal{S})$.

We write $\mathcal{S}_\top$ for the modal specification $\langle \{q^0\}, q^0, \Sigma, \Delta^m, \Delta^M \rangle$ with $(q^0, a, q^0) \in \Delta^m$ for all $a \in \Sigma$ and $\Delta^M = \emptyset$; $\mathcal{S}_\top$ is greater for $\preceq$ than all modal specifications and it admits every behavior as a model. On the other hand, let $\mathcal{S}_\bot$ be a modal specification with $\mathsf{Mod}(\mathcal{S}_\bot) = \emptyset$, refining all modal specifications.

We call $\rho(p\mathcal{S})$ the reduction of $p\mathcal{S}$. It follows the observation that if there is a must-transition $q \xrightarrow{a} \bot$ in $p\mathcal{S}$ then q cannot belong to a simulation relation θ.

Definition 6 (Reduction ρ)**.** *For $U \subseteq Q$, let*

$$pre^M(U) = \{q \in Q \mid (q, a, q') \in \Delta^M \ \wedge \ q' \in U\},$$

$pre_0^M(U) = U$, $pre_{k+1}^M(U) = pre^M(pre_k^M(U))$ *for* $k \geq 0$, *and* $pre_*^M(U) = \bigcup_k pre_k^M(U)$.

The reduction $\rho(p\mathcal{S})$ of a pseudo-modal specification $p\mathcal{S} = \langle Q, q^0, \Sigma, \Delta^m, \Delta^M \rangle$ is defined as $\mathcal{S}_\bot$ if $q^0 \in pre_*^M(\{\bot\})$, and as the modal specification $\langle Q \setminus pre_*^M(\{\bot\}), q^0, \Sigma, \Delta_\rho^m, \Delta_\rho^M \rangle$ otherwise, where $\Delta_\rho^m \subseteq \Delta^m$ and $\Delta_\rho^M \subseteq \Delta^M$ are the sets of transitions of $p\mathcal{S}$ whose source state and destination state do not belong to $pre_*^M(\{\bot\})$.

This construction is similar to the one for the synthesis of a most permissive controller [14] with $\bot$ interpreted as the unique bad state and with must-transitions as transitions labeled by uncontrollable events.

A pseudo-modal specification can be reduced into a modal specification with preservation of its semantics.

Lemma 1 (Consistency)**.** *The modal specification $\rho(p\mathcal{S})$ possesses the same set of models as $p\mathcal{S}$ [7].*

In the sequel, pseudo-modal specifications may serve as an artifice when composing modal specifications; conflicts between the modalities of the composed modal specifications may be represented thanks to a transition to an inconsistent $\bot$ state. By then applying the reduction operation, a semantically equivalent modal specification is obtained.

Remark 1. *The main reason for assuming determinism in the may-transition relation for modal specifications is that the modal refinement then coincides with the inclusion of sets of models [7]: let $\mathcal{S}_1$ and $\mathcal{S}_2$ be two modal specifications:*

$$\mathcal{S}_1 \preceq \mathcal{S}_2 \ \Leftrightarrow \ \mathsf{Mod}(\mathcal{S}_1) \subseteq \mathsf{Mod}(\mathcal{S}_2)$$

If nondeterminism is allowed in $\mathcal{S}_1$ or $\mathcal{S}_2$, modal refinement is not complete [15].

Modal specifications equipped with modal refinement form a complete lattice; let $\mathcal{S}_1 = \langle Q_1, q_1^0, \Sigma, \Delta_1^m, \Delta_1^M \rangle$ and $\mathcal{S}_2 = \langle Q_2, q_2^0, \Sigma, \Delta_2^m, \Delta_2^M \rangle$ be two modal specifications, their infimum and their supremum are the following:

Definition 7 (Greatest lower bound $\wedge$)**.** *The* greatest lower bound *of $\mathcal{S}_1$ and $\mathcal{S}_2$ is $\mathcal{S}_1 \wedge \mathcal{S}_2 = \rho(\mathcal{S}_1 \& \mathcal{S}_2)$ where $\mathcal{S}_1 \& \mathcal{S}_2 = \langle (Q_1 \times Q_2) \cup \{\bot\}, (q_1^0, q_2^0), \Sigma, \Delta^m, \Delta^M \rangle$ with the transitions given by Table I.*

Observe that then for all state of the form (q_1, q_2) of $\mathcal{S}_1 \& \mathcal{S}_2$ we have:

$$may((q_1, q_2)) = may_1(q_1) \cap may_2(q_2)$$
$$must((q_1, q_2)) = must_1(q_1) \cup must_2(q_2)$$

Remark 2. $\mathsf{Mod}(\mathcal{S}_1 \wedge \mathcal{S}_2) = \mathsf{Mod}(\mathcal{S}_1) \cap \mathsf{Mod}(\mathcal{S}_2)$*; thus greatest lower bound operation coincide with a logical conjunction [16].*

Definition 8 (Least upper bound $\vee$)**.** *The* least upper bound *$\mathcal{S}_1 \vee \mathcal{S}_2$ of $\mathcal{S}_1$ and $\mathcal{S}_2$ is the tuple $\langle (Q_1 \times Q_2) \cup Q_1 \cup Q_2, (q_1^0, q_2^0), \Sigma, \Delta^m, \Delta^M \rangle$, where $\Delta^m = \Delta_1^m \cup \Delta_2^m \cup (\Delta^m)'$, $\Delta^M = \Delta_1^M \cup \Delta_2^M \cup (\Delta^M)'$, and $(\Delta^m)'$ and $(\Delta^M)'$ are defined by Table II.*

Then, all states (q_1, q_2) of $\mathcal{S}_1 \vee \mathcal{S}_2$ satisfy:

$$may((q_1, q_2)) = may_1(q_1) \cup may_2(q_2)$$
$$must((q_1, q_2)) = must_1(q_1) \cap must_2(q_2)$$

III. Contracts as Pairs of Modal Specifications

We now enrich our component-based framework with a notion of contracts. As briefly indicated in the introduction, a contract is a pair of specifications: one describes an assumption on the usage of the component made by its environment; the other one corresponds to a guarantee offered by the component as long as the assumption is satisfied.

A. Contracts

Definition 9 (Contract)**.** *A* contract *$\mathcal{C}$ over Σ is a pair $(\mathcal{A}, \mathcal{G})$ of modal specifications over Σ, where $\mathcal{A}$ is called* assumption *and $\mathcal{G}$* guarantee.

A contract $\mathcal{C} = (\mathcal{A}, \mathcal{G})$ on a component with alphabet Σ guarantees any implementation to satisfy $\mathcal{G}$, provided that $\mathcal{A}$ is satisfied. Since the context in which the component will be deployed is unknown at design time, $\mathcal{A}$ can only make assumptions about the locally observable behavior of the component when integrated in its environment, not about the behavior of the environment itself.

Example 2. *The contract $(\mathcal{A}_{Fun}, \mathcal{G}_{Fun})$ consisting of the pair of modal specifications in Fig. 1 specifies that every message sent must be acknowledged, provided that a sent message is re-emitted as long as it has not been acknowledged.*

Definition 10 (Implementation)**.** *A modal specification $\mathcal{S}$ satisfies a contract $\mathcal{C} = (\mathcal{A}, \mathcal{G})$ if $\mathcal{A} \wedge \mathcal{S} \preceq \mathcal{G}$. Models $\mathcal{B}$ of such an $\mathcal{S}$ are then called implementations of $\mathcal{C}$, denoted $\mathcal{B} \models \mathcal{C}$.*

Refining a contract means weakening assumptions and strengthening guarantees:

Definition 11 (Refinement)**.** *A contract $(\mathcal{A}', \mathcal{G}')$ refines $(\mathcal{A}, \mathcal{G})$, written $(\mathcal{A}', \mathcal{G}') \preceq (\mathcal{A}, \mathcal{G})$ if $\mathcal{A} \preceq \mathcal{A}'$ and $\mathcal{G}' \preceq \mathcal{G}$.*

$\wedge$	$q_2 \overset{a}{\rightarrow}_2 q_2'$	$q_2 \overset{a}{\dashrightarrow}_2 q_2'$	$q_2 \overset{a}{\nrightarrow}_2$
$q_1 \overset{a}{\rightarrow}_1 q_1'$	$(q_1,q_2) \overset{a}{\rightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\rightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\rightarrow} \bot$
$q_1 \overset{a}{\dashrightarrow}_1 q_1'$	$(q_1,q_2) \overset{a}{\rightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\dashrightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\nrightarrow}$
$q_1 \overset{a}{\nrightarrow}_1$	$(q_1,q_2) \overset{a}{\rightarrow} \bot$	$(q_1,q_2) \overset{a}{\nrightarrow}$	$(q_1,q_2) \overset{a}{\nrightarrow}$

TABLE I
TRANSITIONS RELATIONS OF $\mathcal{S}_1 \wedge \mathcal{S}_2$

$\vee$	$q_2 \overset{a}{\rightarrow}_2 q_2'$	$q_2 \overset{a}{\dashrightarrow}_2 q_2'$	$q_2 \overset{a}{\nrightarrow}_2$
$q_1 \overset{a}{\rightarrow}_1 q_1'$	$(q_1,q_2) \overset{a}{\rightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\dashrightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\dashrightarrow} q_1'$
$q_1 \overset{a}{\dashrightarrow}_1 q_1'$	$(q_1,q_2) \overset{a}{\dashrightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\dashrightarrow} (q_1',q_2')$	$(q_1,q_2) \overset{a}{\dashrightarrow} q_1'$
$q_1 \overset{a}{\nrightarrow}_1$	$(q_1,q_2) \overset{a}{\dashrightarrow} q_2'$	$(q_1,q_2) \overset{a}{\dashrightarrow} q_2'$	$(q_1,q_2) \overset{a}{\nrightarrow}$

TABLE II
TRANSITIONS RELATIONS OF $\mathcal{S}_1 \vee \mathcal{S}_2$

Contracts can also be ordered by comparing their sets of implementations:

Definition 12 (Model inclusion)**.** *We write $(\mathcal{A}', \mathcal{G}') \sqsubseteq (\mathcal{A}, \mathcal{G})$ if every implementation of $(\mathcal{A}', \mathcal{G}')$ is also an implementation of $(\mathcal{A}, \mathcal{G})$. We write $(\mathcal{A}', \mathcal{G}') \equiv (\mathcal{A}, \mathcal{G})$ if $(\mathcal{A}', \mathcal{G}') \sqsubseteq (\mathcal{A}, \mathcal{G})$ and $(\mathcal{A}, \mathcal{G}) \sqsubseteq (\mathcal{A}', \mathcal{G}')$.*

Proposition 1. *$(\mathcal{A}', \mathcal{G}') \preceq (\mathcal{A}, \mathcal{G}) \Longrightarrow (\mathcal{A}', \mathcal{G}') \sqsubseteq (\mathcal{A}, \mathcal{G})$. The converse is not true.*

One important question following Definition 10 is then: can we compute a modal specification having the same set of implementations as a given contract? This question is addressed by studying the operation inverse to conjunction $\wedge$.

B. Weak Implication

This section introduces a new operation on modal specification called *weak implication* which is a partial adjoint of the conjunction. In other words, the weak implication $\mathcal{S} \div \mathcal{S}_1$ solves the equation $\mathcal{S}_1 \wedge X \leq \mathcal{S}$.

Definition 13 (Weak implication $\div$)**.** *The* weak implication *of two modal specifications $\mathcal{S}_1$ and $\mathcal{S}_2$ is $\mathcal{S}_1 \div \mathcal{S}_2 = \langle (Q_1 \times Q_2) \cup \{\top\}, (q_1^0, q_2^0), \Sigma, \Delta^m, \Delta^M \rangle$ with transitions given by Table III, and $\top \overset{a}{\dashrightarrow} \top$ for all $a \in \Sigma$.*
Let $\mathcal{S}_\bot \div \mathcal{S} = \mathcal{S}_\bot$ when $\mathcal{S} \neq \mathcal{S}_\bot$ and $\mathcal{S} \div \mathcal{S}_\bot = \mathcal{S}_\top$.

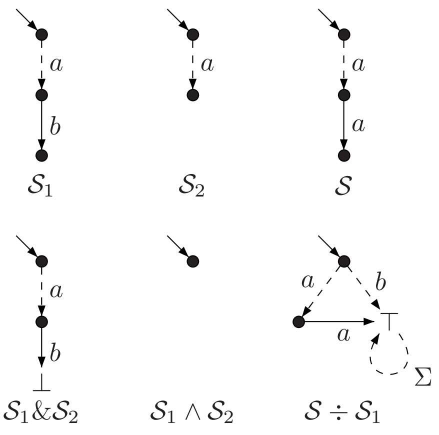
The rules in Table III are such that for $\mathcal{S}_1 \div \mathcal{S}_2$ with $\mathcal{S}_1, \mathcal{S}_2 \neq \mathcal{S}_\bot$, we have:

$$may((q_1, q_2)) = may_1(q_1) \cup mustnot_2(q_2)$$
$$must((q_1, q_2)) = must_1(q_1) \setminus must_2(q_2)$$

Intuitively, for a given pair of states, the modality of each action is defined as the weakest modality such that the conjunction with $\mathcal{S}_2$ refines $\mathcal{S}_1$.

Proposition 2. *Given three modal specifications $\mathcal{S}_1, \mathcal{S}_2$ and $\mathcal{S}$, the following implication holds:*

$$\mathcal{S}_2 \preceq \mathcal{S} \div \mathcal{S}_1 \Longrightarrow \mathcal{S}_1 \wedge \mathcal{S}_2 \preceq \mathcal{S}.$$



a
b
a
a
a
$\mathcal{S}_1$
$\mathcal{S}_2$
$\mathcal{S}$
a
b
$\bot$
a
b
a
$\top$
Σ
$\mathcal{S}_1 \& \mathcal{S}_2$
$\mathcal{S}_1 \wedge \mathcal{S}_2$
$\mathcal{S} \div \mathcal{S}_1$

Fig. 2. Counterexample showing the incompleteness of weak implication $\div$

The converse does not hold in general. Consider the modal specifications in Fig. 2 defined over the alphabet $\{a, b\}$, we have $\mathcal{S}_1 \wedge \mathcal{S}_2 \preceq \mathcal{S}$ but $\mathcal{S}_2 \not\preceq \mathcal{S} \div \mathcal{S}_1$ as after a first occurrence of a, the action a is required in $\mathcal{S} \div \mathcal{S}_1$ whereas, in $\mathcal{S}_2$, a is forbidden.

We now define a relation between modal specifications called *non-conflicting* under which completeness of weak implication is ensured. Intuitively two modal specifications $\mathcal{S}_1$ and $\mathcal{S}_2$ are *non-conflicting* if any action required by one is not forbidden by the other. In [17], this relation is called *independence*.

Definition 14 (Non-conflicting)**.** *Two modal specifications $\mathcal{S}_1$ and $\mathcal{S}_2$ are* non-conflicting *if there exists a relation $\Gamma \subseteq Q_1 \times Q_2$ such that $(q_1^0, q_2^0) \in \Gamma$ and for all pairs $(q_1, q_2) \in \Gamma$:*

- *for every $(q_1, a, q_1') \in \Delta_1^M$ there exists $(q_2, a, q_2') \in \Delta_2^m$ and $(q_1', q_2') \in \Gamma$;*
- *for every $(q_2, a, q_2') \in \Delta_2^M$ there exists $(q_1, a, q_1') \in \Delta_1^m$ and $(q_1', q_2') \in \Gamma$;*
- *if $(q_1, a, q_1') \in \Delta_1^m$ and $(q_2, a, q_2') \in \Delta_2^m$ then $(q_1', q_2') \in$*

$\div$	$q_2 \xrightarrow{a}_2 q'_2$	$q_2 \overset{a}{\dashrightarrow}_2 q'_2$	$q_2 \overset{a}{\not\rightarrow}_2$
$q_1 \xrightarrow{a}_1 q'_1$	$(q_1, q_2) \overset{a}{\dashrightarrow} (q'_1, q'_2)$	$(q_1, q_2) \xrightarrow{a} (q'_1, q'_2)$	$(q_1, q_2) \xrightarrow{a} \top$
$q_1 \overset{a}{\dashrightarrow}_1 q'_1$	$(q_1, q_2) \overset{a}{\dashrightarrow} (q'_1, q'_2)$	$(q_1, q_2) \overset{a}{\dashrightarrow} (q'_1, q'_2)$	$(q_1, q_2) \overset{a}{\dashrightarrow} \top$
$q_1 \overset{a}{\not\rightarrow}_1$	$(q_1, q_2) \overset{a}{\not\rightarrow}$	$(q_1, q_2) \overset{a}{\not\rightarrow}$	$(q_1, q_2) \overset{a}{\dashrightarrow} \top$

TABLE III
TRANSITIONS RELATIONS OF $\mathcal{S}_1 \div \mathcal{S}_2$

Γ.

Example 3. *The modal specifications $\mathcal{S}_1$ and $\mathcal{S}_2$ in Fig. 2 are not non-conflicting as, after the occurrence of a, the action b is required in $\mathcal{S}_1$ and forbidden in $\mathcal{S}_2$.*

If $\mathcal{S}_1$ and $\mathcal{S}_2$ are non-conflicting then the computation of the greatest lower bound $\mathcal{S}_1 \wedge \mathcal{S}_2$ does not produce inconsistency, as the rule $q_1 \xrightarrow{a} q'_1$ and $q_2 \overset{a}{\not\rightarrow}$, which entails a must-transition $(q_1, q_2) \xrightarrow{a} \bot$, is never applied.

Proposition 3. *Given three modal specifications $\mathcal{S}_1, \mathcal{S}_2$ and $\mathcal{S}$ such that $\mathcal{S}_1$ and $\mathcal{S}_2$ are non-conflicting:*

$$\mathcal{S}_1 \wedge \mathcal{S}_2 \preceq \mathcal{S} \Longrightarrow \mathcal{S}_2 \preceq \mathcal{S} \div \mathcal{S}_1.$$

Weak implication is called a *partial* adjoint of conjunction as it is correct (Prop. 2) but complete only under certain assumption (Prop. 3). A correct and complete construction which would then be an adjoint of conjunction does not exist for modal specifications.

Lemma 2. *For modal specifications $\mathcal{S}, \mathcal{S}', \mathcal{S}_1$, and $\mathcal{S}_2$, $\mathcal{S} \wedge \mathcal{S}_1 \preceq \mathcal{S}'$ and $\mathcal{S} \wedge \mathcal{S}_2 \preceq \mathcal{S}'$ does not imply $\mathcal{S} \wedge (\mathcal{S}_1 \vee \mathcal{S}_2) \preceq \mathcal{S}'$ in general.*

Proof: Consider the modal specifications in Fig. 3: $\mathcal{S} \wedge \mathcal{S}_1 \preceq \mathcal{S}'$ and $\mathcal{S} \wedge \mathcal{S}_2 \preceq \mathcal{S}'$ but $\mathcal{S} \wedge (\mathcal{S}_1 \vee \mathcal{S}_2) \not\preceq \mathcal{S}'$. ■

Theorem 1. *Conjunction for modal specification does not have an adjoint.*

Proof: Suppose that there exists a construction denoted $\diamond$ which is an adjoint of conjunction. We would thus have, for any modal specifications $\mathcal{S}, \mathcal{S}'$ and $\mathcal{S}_i$, $i = 1, 2$:

$$\mathcal{S} \wedge \mathcal{S}_i \preceq \mathcal{S}' \iff \mathcal{S}_i \preceq \mathcal{S}' \diamond \mathcal{S}$$

If $\mathcal{S}_i \preceq \mathcal{S}' \diamond \mathcal{S}$ then $\mathcal{S}_1 \vee \mathcal{S}_2 \preceq \mathcal{S}' \diamond \mathcal{S}$, by definition of the least upper bound. By correctness of $\diamond$ we would then have: $\mathcal{S} \wedge (\mathcal{S}_1 \vee \mathcal{S}_2) \preceq \mathcal{S}'$, in contradiction with Lemma 2. ■

Proposition 4. *Given four modal specifications $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ and $\mathcal{S}_4$:*

1) $\mathcal{S}_1 \preceq \mathcal{S}_2$ *and* $\mathcal{S}_3 \preceq \mathcal{S}_4 \implies \mathcal{S}_1 \div \mathcal{S}_4 \preceq \mathcal{S}_2 \div \mathcal{S}_3$
2) $\mathcal{S}_2 \wedge (\mathcal{S}_1 \div \mathcal{S}_2) \preceq \mathcal{S}_1 \preceq \mathcal{S}_1 \div \mathcal{S}_2$

For a given contract $\mathcal{C} = (\mathcal{A}, \mathcal{G})$, the weak implication $\mathcal{G} \div \mathcal{A}$ is called the *implicit form* of $\mathcal{C}$. It is refined by all specification $\mathcal{S}$ satisfying $\mathcal{C}$ and such that $\mathcal{S}$ and $\mathcal{A}$ are non-conflicting.

Example 4. *The implicit form of the contract $\mathcal{C}_{Fun}$ in Fig. 1 is depicted in Fig. 4. While* ack *and* nack *are forbidden in the initial state of the assumption $\mathcal{A}_{Fun}$, the implicit form of*

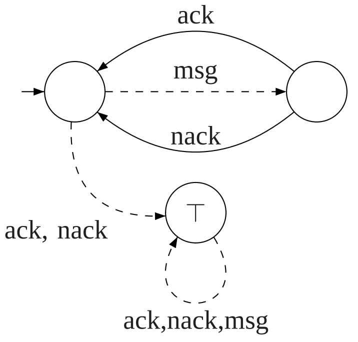


Fig. 4. Implicit form of the contract $\mathcal{C}_{Fun}$

$\mathcal{C}_{Fun}$ has a may-transition from the initial state to a $\top$-state labelled by ack *and* nack. *This represents the consequence of the violation of the assumption: if* ack *or* nack *occur then the contract is relaxed and no more guarantee is provided.*

IV. Composition of Contracts

Composition operations for contracts may be used with different goals in a design flow, and hence, be given different semantics: on the one hand, a contract may describe the guarantees a component is able to give, depending on its environment. Accordingly, the *component composition* of two contracts is the *strongest* contract satisfied by the composition of any pair of implementations of both contracts.

On the other hand, a contract may be used to specify a *requirement* as a guarantee that must be ensured under some hypothesis. We call this a requirement contract, or *aspect*. We define two operations for *aspect composition* of two contracts: one with a conjunctive semantics, defined as the *weakest* common refinement of both contracts, and *priority composition*, allowing a contract of higher priority to override a lower-priority contract in case of conflict.

A. Component Composition

For disjoint component contracts we define a "best effort" composition operation that is parametrized by an *interaction model* inspired by the BIP framework [4], [5]. The composition ensures each guarantee depending on the satisfaction of its assumption, provided that the guarantee is feasible under the specified interaction model.

Definition 15 (Interaction model). *An* interaction *over an alphabet of actions Σ is a non-empty subset of Σ. An* interaction

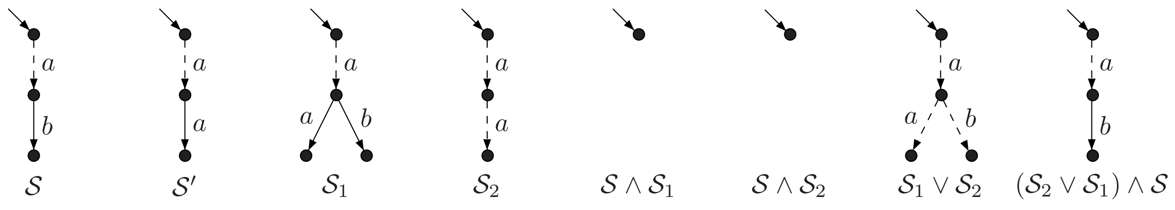

a
b
$\mathcal{S}$
a
a
$\mathcal{S}'$
a
a
b
$\mathcal{S}_1$
a
a
$\mathcal{S}_2$
$\mathcal{S} \wedge \mathcal{S}_1$
$\mathcal{S} \wedge \mathcal{S}_2$
a
a
b
$\mathcal{S}_1 \vee \mathcal{S}_2$
a
b
$(\mathcal{S}_2 \vee \mathcal{S}_1) \wedge \mathcal{S}$

Fig. 3. Example where $\mathcal{S} \wedge \mathcal{S}_1 \preceq \mathcal{S}'$ and $\mathcal{S} \wedge \mathcal{S}_2 \preceq \mathcal{S}'$ but $\mathcal{S} \wedge (\mathcal{S}_1 \vee \mathcal{S}_2) \npreceq \mathcal{S}'$.

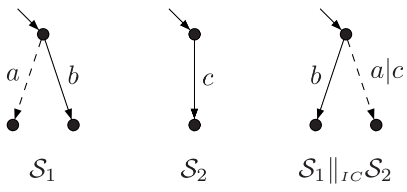


Fig. 5. Example of composition $\|_{IC}$

model *over* Σ *is a set* $IC \subseteq 2^{\Sigma}$ *of interactions over* Σ *such that* $\bigcup_{\alpha \in IC} \alpha = \Sigma$.

An interaction between behaviors is a set of actions taking place simultaneously. An interaction model defines the set of all possible interactions. For disjoint alphabets Σ_i, $i = 1, ..., n$, and an interaction α over $\bigcup_i \Sigma_i$, let $\alpha[i] = \alpha \cap \Sigma_i$.

Definition 16 (Composition $\|_{IC}$)**.** *The composition of two modal specifications* $\mathcal{S}_i = \langle Q_i, q_i^0, \Sigma_i, \Delta_i^m, \Delta_i^M \rangle$, $i = 1, 2$ *with disjoint alphabets, under an interaction model* IC *over* $\Sigma_1 \uplus \Sigma_2$ *is* $\mathcal{S}_1 \|_{IC} \mathcal{S}_2 = \langle Q_1 \times Q_2, (q_1^0, q_2^0), IC, \Delta^m, \Delta^M \rangle$, *where the transition relations* Δ^m *and* Δ^M *are obtained by synchronizing respectively* Δ_i^m *and* Δ_i^M*: for any* $\alpha \in IC$,

- $((q_1, q_2), \alpha, (q_1', q_2')) \in \Delta^m$ *if* $(q_i, \alpha[i], q_i') \in \Delta_i^m$ *when* $\alpha[i] \neq \emptyset$, *and* $q_i = q_i'$ *otherwise,* $i = 1, 2$;
- $((q_1, q_2), \alpha, (q_1', q_2')) \in \Delta^M$ *if* $(q_i, \alpha[i], q_i') \in \Delta_i^M$ *when* $\alpha[i] \neq \emptyset$, *and* $q_i = q_i'$ *otherwise,* $i = 1, 2$.

Let $\mathcal{S} \|_{IC} \mathcal{S}_\perp = \mathcal{S}_\perp \|_{IC} \mathcal{S} = \mathcal{S}_\perp$.

Example 5. *Fig. 5 shows the modal specifications* $\mathcal{S}_1$ *and* $\mathcal{S}_2$, *defined over the alphabets* $\{a, b\}$ *and* $\{c, d\}$, *respectively, and their composition* $\mathcal{S}_1 \|_{IC} \mathcal{S}_2$ *under the interaction model*

$$IC = \big\{\{b\}, \{a, c\}, \{b, d\}\big\}$$

where a|c *and* b|d *stand for* $\{a, c\}$ *and* $\{b, d\}$.

Refinement is a congruence with respect to composition (this property is also called *stepwise refinement* of modal specifications):

Proposition 5. *For modal specifications* $\mathcal{S}$ *over* Σ, $\mathcal{S}_1, \mathcal{S}_2$ *over* Σ', *and* IC *over* $\Sigma \uplus \Sigma'$,

$$\mathcal{S}_1 \preceq \mathcal{S}_2 \implies \mathcal{S}_1 \|_{IC} \mathcal{S} \preceq \mathcal{S}_2 \|_{IC} \mathcal{S}$$

Consider two behaviors $\mathcal{B}_1$ and $\mathcal{B}_2$ that are respectively implementations of contracts $\mathcal{C}_1$ and $\mathcal{C}_2$. When a system is built bottom-up by composing $\mathcal{B}_1$ and $\mathcal{B}_2$ via an interaction model IC, one obvious question is: what can be inferred on $\mathcal{B}_1 \|_{IC} \mathcal{B}_2$ thanks to $\mathcal{C}_1$ and $\mathcal{C}_2$? To this end, we define the composition operation $\otimes_{IC}$ on contracts.

Definition 17 (Composition $\otimes_{IC}$)**.** *Given contracts* $\mathcal{C}_1 = (\mathcal{A}_1, \mathcal{G}_1)$ *and* $\mathcal{C}_2 = (\mathcal{A}_2, \mathcal{G}_2)$ *on disjoint alphabets and an interaction model* IC, *we define* $\mathcal{C}_1 \otimes_{IC} \mathcal{C}_2 = (\mathcal{A}, \mathcal{G})$ *where:*

- $\mathcal{G} = (\mathcal{G}_1 \div \mathcal{A}_1) \|_{IC} (\mathcal{G}_2 \div \mathcal{A}_2)$
- $\mathcal{A} = (\mathcal{A}_1 \|_{IC} \mathcal{A}_2) \div \mathcal{G}$

Proposition 6. *The composition* $\otimes_{IC}$ *is associative and commutative.*

The property of *independent implementability* (also referred to as *constructivity* in [18]) allows to obtain an implementation of the composition of contracts as the composition of their implementations.

Theorem 2 (Independent implementability)**.** *Let* $\mathcal{C}_1, \mathcal{C}_2$ *be two contracts on disjoint alphabets and* IC *an interaction model, if* $\mathcal{B}_1 \models \mathcal{C}_1$ *and* $\mathcal{B}_2 \models \mathcal{C}_2$ *then* $\mathcal{B}_1 \|_{IC} \mathcal{B}_2 \models \mathcal{C}_1 \otimes_{IC} \mathcal{C}_2$.

Since the contract $\mathcal{C}_1 \otimes_{IC} \mathcal{C}_2$ is satisfied by the composition under IC of any pair of implementations of $\mathcal{C}_1$ and $\mathcal{C}_2$, the composition operation $\otimes_{IC}$ allows to reason about contracts in a bottom-up manner.

Proposition 7 (Stepwise refinement)**.** *Given three contracts* $\mathcal{C}, \mathcal{C}_1, \mathcal{C}_2$, *the following holds:*

$$\mathcal{C}_1 \preceq \mathcal{C}_2 \implies \mathcal{C}_1 \otimes_{IC} \mathcal{C} \preceq \mathcal{C}_2 \otimes_{IC} \mathcal{C}$$

Example 6. *Consider a client whose alphabet consists of the three actions:* send *for a message to be transmitted, and two kinds of responses:* ok *if the message has been received, and* fail *otherwise. The contract in Fig. 6 states that, under the hypothesis that* fail *never occurs, every transmitted message is well received (i.e.,* send *is acknowledged by* ok*).*

The interactions between the client and the communication channel are given by the interaction model

$$IC = \{\{msg, send\}, \{ack, ok\}, \{nack, fail\}\}$$

The result of composing $\mathcal{C}_{Fun}$ *of Fig. 1 and* $\mathcal{C}_{Client}$ *of Fig. 6 is depicted in Fig. 7. In the obtained guarantee,* msg|send *is*

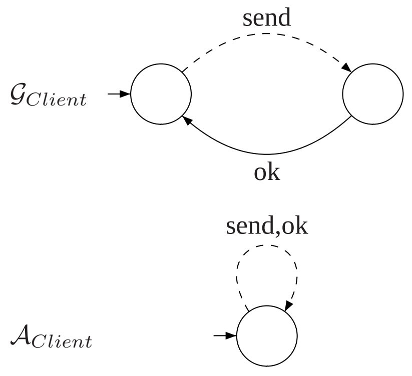


Fig. 6. A contract $\mathcal{C}_{Client}$ for a client component

followed by ack|ok *unless* nack|fail *occurs, which constitutes a violation of* $\mathcal{A}_{Client}$*. From then on,* msg|send *may be followed by* ack|ok *or* nack|fail.

By independent implementability, the composition of any pair of implementations of $\mathcal{C}_{Fun}$ *and* $\mathcal{C}_{Client}$ *satisfies* $\mathcal{C}_{Fun} \otimes_{IC} \mathcal{C}_{Client}$*. This may look surprising, since the assumption* $\mathcal{A}_{\otimes_{IC}}$ *allows* ack|ok *to take place from the initial state, whereas* ack *was not allowed to happen from the initial state of* $\mathcal{A}_{Fun}$*. This is because this behavior is ruled out by* $\mathcal{C}_{Client}$*, which guarantees only* send *to take place. In other words, the information of* $\mathcal{C}_{Client}$ *is used to weaken the assumption* $\mathcal{A}_{\otimes_{IC}}$*.*

B. Aspect Composition

It is current engineering practice to model different aspects of a specification separately. In terms of contracts, this amounts to attach several contracts to a single component. A central question is whether a set of contracts is consistent, and how to compute a common implementation, or *shared refinement* [19]. We define the composition of contracts $\mathcal{C}_1$ and $\mathcal{C}_2$ as the *weakest* contract C refining each aspect, that is, making the guarantee of each contract provided that the corresponding assumption is met.

Definition 18 (Composition $\oplus$)**.** *Given two contracts* $\mathcal{C}_1 = (\mathcal{A}_1, \mathcal{G}_1)$ *and* $\mathcal{C}_2 = (\mathcal{A}_2, \mathcal{G}_2)$ *over the same alphabet, let*

$$\mathcal{C}_1 \oplus \mathcal{C}_2 = \big(\mathcal{A}_1 \vee \mathcal{A}_2,\ (\mathcal{G}_1 \div \mathcal{A}_1) \wedge (\mathcal{G}_2 \div \mathcal{A}_2)\big)$$

A component satisfying the composition of two contracts must satisfy each guarantee if and only if the corresponding assumption holds. This notion of composition is consistent with the component paradigm mentioned above, enabling the component to offer different guarantees depending on the context.

Proposition 8. *The composition* $\oplus$ *is associative and commutative. Moreover, it is* $\equiv$*-idempotent: for any contract* $\mathcal{C}$*,* $\mathcal{C} \oplus \mathcal{C} \equiv \mathcal{C}$*.*

Theorem 3 (Correctness of $\oplus$)**.** *Each implementation of* $\mathcal{C}_1 \oplus \mathcal{C}_2$ *is an implementation of* $\mathcal{C}_1$ *and of* $\mathcal{C}_2$*, that is,* $\mathcal{C}_1 \oplus \mathcal{C}_2 \sqsubseteq \mathcal{C}_i$*,* $i = 1, 2$*.*

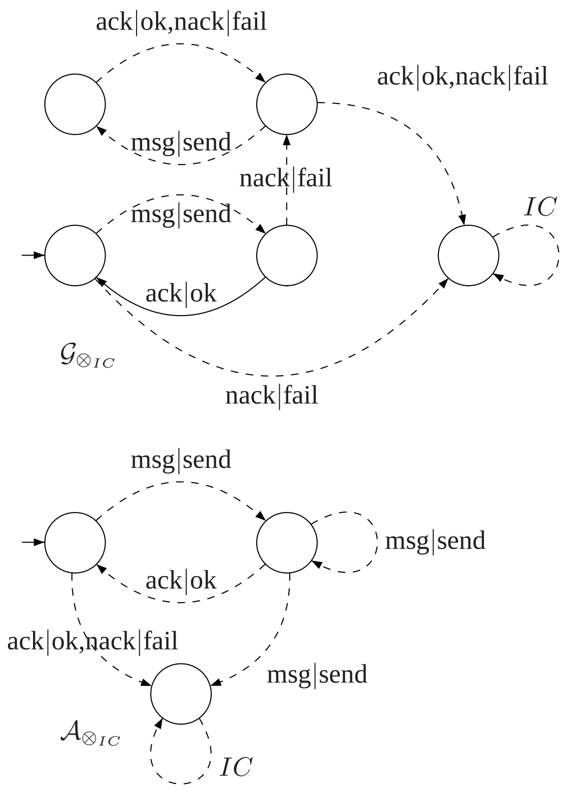


Fig. 7. The contract $\mathcal{C}_{Fun} \otimes_{IC} \mathcal{C}_{Client}$

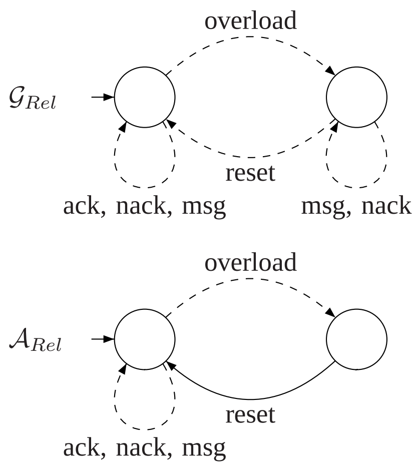


Fig. 8. Communication channel – reliability aspect $\mathcal{C}_{Rel}$

Theorem 4 (Completeness of $\oplus$)**.** *Consider two contracts* $\mathcal{C}_1 = (\mathcal{A}_1, \mathcal{G}_1)$ *and* $\mathcal{C}_2 = (\mathcal{A}_2, \mathcal{G}_2)$ *over the same alphabet. If* $\mathcal{S}$ *is a modal specification that satisfies* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *and is non-conflicting with* $\mathcal{A}_1$ *and* $\mathcal{A}_2$*, then* $\mathcal{S}$ *satisfies* $\mathcal{C}_1 \oplus \mathcal{C}_2$*.*

Proposition 9 (Stepwise refinement)**.** *For contracts* $\mathcal{C}, \mathcal{C}_1, \mathcal{C}_2$

over the same alphabet,

$$\mathcal{C}_1 \preceq \mathcal{C}_2 \implies \mathcal{C}_1 \oplus \mathcal{C} \preceq \mathcal{C}_2 \oplus \mathcal{C}$$

Example 7. *Consider now a second contract $\mathcal{C}_{Rel}$ in Fig. 8 dealing with the reliability of a communication channel. Two new actions are introduced:* overload *which occurs when the maximal capacity of the communication channel is reached and* reset *for the re-initialization of the system. The contract $\mathcal{C}_{Rel}$ specifies the following:*

- *assumption $\mathcal{A}_{Rel}$: on* overload*, the system must be reset.*
- *guarantee $\mathcal{G}_{Rel}$: on* overload*, the communication channel only produces* nack*; or, equivalently, messages can be positively acknowledged only when the system is not overloaded.*

We revisit contract $\mathcal{C}_{Fun}$ of Fig. 1 and extend its alphabet by adding may*-self-loops labeled with* overload *and* reset *on every state of $\mathcal{A}_{Fun}$ and $\mathcal{G}_{Fun}$. Call the resulting contract $\mathcal{C}'_{Fun}$.*

The guarantee obtained by composing the two aspects $\mathcal{C}'_{Fun}$ and $\mathcal{C}_{Rel}$ is depicted in Fig. 9. It can be decomposed in four blocks:

- *the upper left block corresponds to the situation where none of the assumptions made in the two contracts is violated and thus, both guarantees are verified;*
- *the upper right block corresponds to the case where the assumptions of $\mathcal{C}'_{Fun}$ has been violated. As a consequence, the contract $\mathcal{C}'_{Fun}$ is disabled and only $\mathcal{G}_{Rel}$ is ensured;*
- *similarly, in the bottom left block, the assumption of $\mathcal{C}_{Rel}$ have been violated and only $\mathcal{G}'_{Fun}$ is ensured;*
- *last, the bottom right block is a $\top$-state corresponding to the situation where both assumptions of $\mathcal{C}'_{Fun}$ and $\mathcal{C}_{Rel}$ have been violated and both contracts are disabled.*

In practice, aspects are not equally important. For instance, an aspect "safety" may be chosen to override an aspect "quality of service". The operation of *priority* composes aspects in a hierarchical order, such that in case of inconsistency, an aspects of higher priority overrides a lower-priority contract. A similar operation for the four-valued Belnap logic has been introduced in [20] to compose access control policies.

Definition 19 (Priority composition)**.** *Let $\mathcal{S}_1 = \langle Q_1, q_1^0, \Sigma, \Delta_1^m, \Delta_1^M \rangle$ and $\mathcal{S}_2 = \langle Q_2, q_2^0, \Sigma, \Delta_2^m, \Delta_2^M \rangle$ be two modal specifications, $\mathcal{S}_1 < \mathcal{S}_2$ is the tuple $\langle (Q_1 \times Q_2) \cup Q_2, (q_1^0, q_2^0), \Sigma, \Delta^m, \Delta^M \rangle$ with $\Delta^m = \Delta_2^m \cup (\Delta^m)'$, $\Delta^M = \Delta_2^M \cup (\Delta^M)'$, and $(\Delta^m)'$ and $(\Delta^M)'$ are defined by Table IV.*

Intuitively, $\mathcal{S}_1 < \mathcal{S}_2$ behaves, for each action, like $\mathcal{S}_2$ whenever the modality in $\mathcal{S}_2$ is different from "may" (undetermined), otherwise it behaves like $\mathcal{S}_1$.

Definition 20 (Priority $<$ on contracts)**.** *Given two contracts $\mathcal{C}_1 = (\mathcal{A}_1, \mathcal{G}_1)$ and $\mathcal{C}_2 = (\mathcal{A}_2, \mathcal{G}_2)$ over the same alphabet, we define:*

$$\mathcal{C}_1 < \mathcal{C}_2 = (\ \mathcal{A}_1 \vee \mathcal{A}_2,\ \mathcal{G}_1 < \mathcal{G}_2\ )$$

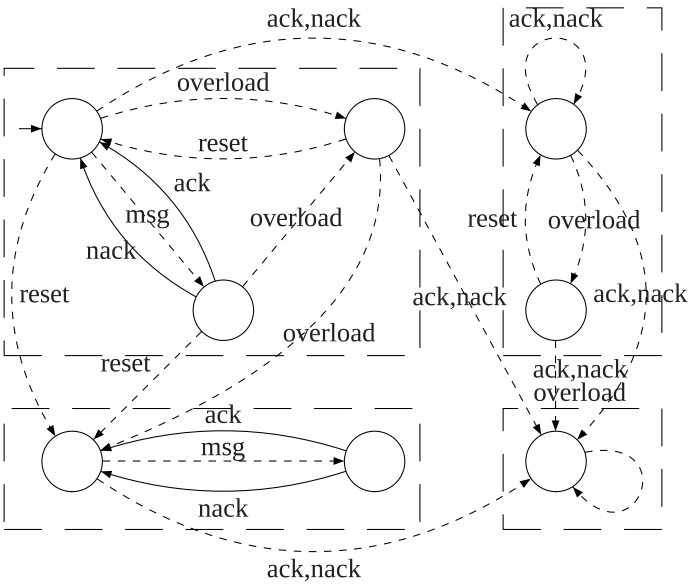


Fig. 9. The guarantee of $\mathcal{C}'_{Fun} \oplus \mathcal{C}_{Rel}$.

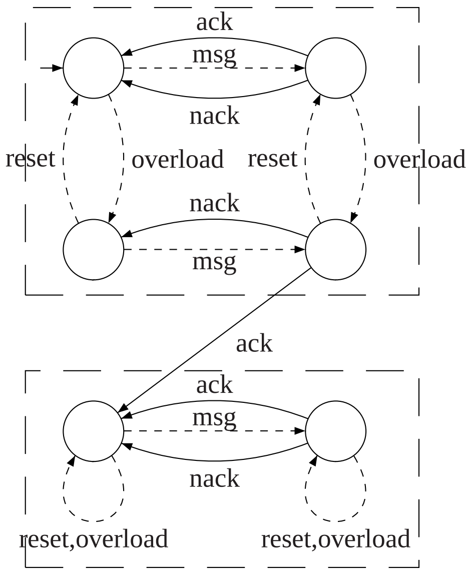


Fig. 10. The guarantee of $\mathcal{C}_{Rel} < \mathcal{C}'_{Fun}$

Example 8. *The guarantee obtained by composing $\mathcal{C}_{Rel} < \mathcal{C}'_{Fun}$ is depicted in Fig. 10. When the communication channel is on overload, a positive acknowledgment* ack *may occur as specified in the higher-priority contract $\mathcal{C}'_{Fun}$, which was impossible in $\mathcal{C}_{Rel}$. The remaining guarantee is $\mathcal{G}'_{Fun}$ (lower block).*

Proposition 10. 1) *$\mathcal{C}_1 < \mathcal{C}_2 \sqsubseteq \mathcal{C}_2$;*
2) *$\mathcal{C} < (\mathcal{S}_\bot, \mathcal{S}_\top) = \mathcal{C}$.*

That is, all models of the priority composition satisfy the contract of higher priority. Whenever the latter does not make

$<$	$q_2 \overset{a}{\longrightarrow}_2 q_2'$	$q_2 \overset{a}{\dashrightarrow}_2 q_2'$	$q_2 \overset{a}{\not\rightarrow}_2$
$q_1 \overset{a}{\longrightarrow}_1 q_1'$	$(q_1, q_2) \overset{a}{\longrightarrow} (q_1', q_2')$	$(q_1, q_2) \overset{a}{\longrightarrow} (q_1', q_2')$	$(q_1, q_2) \overset{a}{\not\rightarrow}$
$q_1 \overset{a}{\dashrightarrow}_1 q_1'$	$(q_1, q_2) \overset{a}{\longrightarrow} (q_1', q_2')$	$(q_1, q_2) \overset{a}{\dashrightarrow} (q_1', q_2')$	$(q_1, q_2) \overset{a}{\not\rightarrow}$
$q_1 \overset{a}{\not\rightarrow}_1$	$(q_1, q_2) \overset{a}{\longrightarrow} q_2'$	$(q_1, q_2) \overset{a}{\not\rightarrow}$	$(q_1, q_2) \overset{a}{\not\rightarrow}$

TABLE IV
TRANSITIONS RELATIONS OF $\mathcal{S}_1 < \mathcal{S}_2$

a choice, the models must satisfy the choice made by $\mathcal{C}_1$.

Proposition 11 (Associativity)**.**

$$(\mathcal{C}_1 < \mathcal{C}_2) < \mathcal{C}_3 \ = \ \mathcal{C}_1 < (\mathcal{C}_2 < \mathcal{C}_3)$$

We can therefore extend priority composition to a hierarchy of an arbitrary number of aspects.

V. DISCUSSION

We have defined modal contracts as a pair of modal specifications, and introduced a new operation on modal specification called *weak implication*. Based on this operation, we have introduced three composition operations between modal contracts, responding to different requirements in the design flow and satisfying different properties:

- *component composition* $\otimes_{IC}$ of contracts over disjoint sub-systems, parametrized with an interaction model IC. The composition is defined as the *strongest* contract satisfying the property of independent implementability;
- *aspect composition* $\oplus$ of contracts over the same sub-system, defined as the *weakest* contract refining both arguments;
- *priority composition* $<$ of contracts over the same sub-system, allowing to guarantee a hierarchy of importance among the contracts.

Modal contracts and their compositions have been illustrated with the example of a communication network.

In the near future, we intend to study assume-guarantee rules for circular reasoning in this framework, lift weak implication from modal specifications to contracts, and apply the framework to significant case studies. Another interesting research direction is the support of a richer deontic logic, for instance, contracts encompassing *contrary-to-duty obligations* [21], [22] that specify how contract violations are handled.

REFERENCES

[1] B. Meyer, "Applying "design by contract"," *IEEE Computer*, vol. 25, no. 10, pp. 40–51, 1992.

[2] L. de Alfaro and T. Henzinger, "Interface automata," in *Proc. FSE'01*. ACM Press, 2001, pp. 109–120.

[3] K. Larsen, U. Nyman, and A. Wasowski, "Interface input/output automata," in *Proc. FM'06*, ser. LNCS, vol. 4085. Springer, 2006, pp. 82–97.

[4] G. Gössler and J. Sifakis, "Composition for component-based modeling," *Science of Computer Programming*, vol. 55, no. 1-3, pp. 161–183, 3 2005.

[5] S. Bliudze and J. Sifakis, "The algebra of connectors — structuring interaction in BIP," in *Proc. EMSOFT'07*. ACM, 2007, pp. 11–20.

[6] K. Larsen, "Modal specifications," in *Proc. International Workshop on Automatic Verification Methods for Finite State Systems*, ser. LNCS, vol. 407. Springer, 1989, pp. 232–246.

[7] J.-B. Raclet, "Residual for component specifications," in *Proc. FACS'07*, ser. ENTCS, 2007.

[8] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, and R. Passerone, "Why modalities are good for interface theories?" in *Proc. ACSD'09*. IEEE, 2009.

[9] S. Quinton and S. Graf, "Contract-based verification of hierarchical systems of components," in *Proc. SEFM'08*. IEEE, 2008, pp. 377–281.

[10] A. Benveniste, B. Caillaud, A. Ferrari, L. Mangeruca, R. Passerone, and C. Sofronis, "Multiple viewpoint contract-based specification and design," in *Proc. FMCO'07*, ser. LNCS, F. de Boer et al., Ed., vol. 5382. Springer, 2008, pp. 200–225.

[11] G. Lüttgen and W. Vogler, "Conjunction on processes: Full abstraction via ready-tree semantics," *TCS*, vol. 373, pp. 19–40, 2007.

[12] A. Antonik, M. Huth, K. Larsen, U. Nyman, and A. Wasowski, "20 years of modal and mixed specifications," *Bulletin of European Association of Theoretical Computer Science*, vol. 1, no. 94, 2008.

[13] A. Benveniste, "Multiple viewpoint contracts and residuation," fIT'08.

[14] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, 1999.

[15] K. Larsen, U. Nyman, and A. Wasowski, "On modal refinement and consistency," in *Proc. CONCUR'07*, ser. LNCS, vol. 4703. Springer, 2007, pp. 105–119.

[16] J.-B. Raclet, "Quotient de spécifications pour la réutilisation de composants," Ph.D. dissertation, Université de Rennes I, december 2007, (In French).

[17] K. Larsen, B. Steffen, and C. Weise, "A constraint oriented proof methodology based on modal transition systems," in *Proc. TACAS'95*, ser. LNCS. Springer, 1995, pp. 17–40.

[18] T. A. Henzinger and J. Sifakis, "The embedded systems design challenge," in *Proc. FM'06)*, ser. Lecture Notes in Computer Science, vol. 4085. Springer, 2006, pp. 1–15.

[19] L. Doyen, T. Henzinger, B. Jobstmann, and T. Petrov, "Interface theories with component reuse," in *Proc. EMSOFT'08*, L. de Alfaro and J. Palsberg, Eds. ACM Press, 2008, pp. 79–88.

[20] G. Bruns, D. Dantas, and M. Huth, "A simple and expressive semantic framework for policy composition in access control," in *Proc. FMSE'07*. ACM, 2007, pp. 12–21.

[21] G. Governatori and Z. Milosevic, "Dealing with contract violations: Formalism and domain specific language," in *Proc. EDOC'05*. IEEE, 2005.

[22] G. Pace and G. Schneider, "Challenges in the specification of full contracts," in *Proc. IFM'09*, ser. LNCS, M. Leuschel and H. Wehrheim, Eds., vol. 5423. Springer, 2009, pp. 292–306.