# Assembly Theories for Communication-Safe Component Systems*

Rolf Hennicker[1], Alexander Knapp[2], and Martin Wirsing[1]

[1] Ludwig-Maximilians-Universität München
{hennicke,wirsing}@pst.ifi.lmu.de
[2] Universität Augsburg
knapp@informatik.uni-augsburg.de

*Dedicated to Joseph Sifakis*

**Abstract.** We propose an abstract notion of an assembly theory that formalizes rudimentary requirements for systems of interacting components. Among these are a composition operator for assemblies, a communication-safety predicate to express the absence of communication errors, a refinement relation for assemblies, and a packing operation to encapsulate assemblies into components thus allowing hierarchical system constructions. We establish laws that must be satisfied by any concrete assembly theory in order to support compositionality of communication-safety, of encapsulation and of refinement. Moreover, refinement must behave well w.r.t. communication-safety and encapsulation. As a concrete instance we investigate a modal assembly theory using modal I/O-interfaces (MIOs) for modeling observable component behaviors and MIOs with possible error states (indicating communication errors) for modeling assembly behaviors. We show that all rules of an assembly theory are satisfied by modal assemblies, in particular the compositionality requirements hold.

## 1 Introduction

In his recent article [20], Joseph Sifakis advocates "rigorous system design" to build systems of guaranteed quality. The abstract principles of component-based design and correctness-by-construction are two main ingredients of his approach. He writes that "components are essential for enhanced productivity and correctness through reuse and architectures" and aims at "theory and rules for building complex designs ... by composing properties of simpler designs." As a concrete instance of these principles Joseph and his research group at Verimag have developed the BIP component framework (see, e.g., [3,20]). BIP allows the modeling of composite, hierarchically structured systems from atomic components characterized by their behavior and their interface.

Several other approaches have been proposed for specifying structural as well as behavioral aspects of components and their interfaces; an overview of a collection of

* This work has been partially sponsored by the European Union under the FP7-project ASCENS, 257414.

component frameworks, that have been applied to a common component modeling example, is given in [19]. The specification of networks of components with arbitrarily (finitely) many members and their interactions is supported by BIP [3,13] as well as by other formalisms for modeling distributed component systems, like CFSMs [7], team automata [8], component-interaction automata [9], or modal assemblies [15].

Abstract principles for the construction of component-based concurrent systems have been investigated and formalized as so-called interface theories by de Alfaro and Henzinger [11,12]. As in Joseph's approach, compositionality requirements are at the heart of these formalisms expressing, e.g., the principles of incremental design and independent implementability at an abstract level. Interfaces for complex components are constructed from simpler ones by interface composition. The result of an interface composition yields again an interface which intuitively describes the visible (black-box) behavior of a composite component. However, in spite of their importance for distributed component systems, there is no abstract formalization of the interaction behavior of component networks with arbitrarily (finitely) many members.

In this work we investigate networks of interacting components in the spirit of Joseph's correctness-by-construction approach and de Alfaro's and Henzinger's interface theories: we develop novel compositional principles for the safe interaction of networks and formalize them by the abstract notion of assembly theory; moreover, we present so-called modal assemblies as a concrete instance of an assembly theory.

An assembly theory formalizes basic requirements for systems of interacting components. In particular, an assembly theory comprises a composition operator for assemblies, a communication-safety predicate for expressing the absence of communication errors, a refinement relation for assemblies, and a packing operation for encapsulating assemblies into components thus allowing hierarchical system constructions. Any assembly theory must satisfy compositionality and compatibility laws for communication-safety, encapsulation and refinement.

We instantiate our framework by a novel concrete assembly theory which uses an extension of MIOs (modal I/O-transition systems; see [17]) to model interface and assembly behaviors. Modal assemblies have already been considered in [15], but the new approach is a significant enhancement. We consider now assembly composition, a new definition of assembly behaviors which explicitly takes into account communication-errors and a new, much more flexible refinement notion for modal assemblies. As a consequence, we get novel results for compositionality of the communication-safety property and for assembly refinement. Moreover, [15] does not define a rigorous, abstract meta-theory for assemblies but provides only some first ideas in that direction.

*Outline of the paper.* In Sect. 2 we develop the general concepts and laws of an assembly theory. Sect. 3 summarizes the basic notions of modal I/O-transition systems needed in Sect. 4 to build a modal assembly theory as an instance of our abstract framework. Finally, in Sect. 5, we finish with some concluding remarks.

*Personal Note.* The third author has known Joseph for several years and collaborates with him and his Verimag research group since 2010. Initial ideas for the cooperation started in 2008 at a workshop of the EC Coordinated Action INTERLINK. Joseph gave a keynote speech on Rigorous System Design while MW was coordinating a Working

Group on software-intensive systems and was giving a talk about Ensemble Engineering. Two years later both (together with Ugo Montanari, Rocco De Nicola, and others) teamed up for planning a joint EU project on the systematic construction of autonomous systems. The project proposal was successful: the FP7 Integrated Project ASCENS [2] on "Autonomic Service-Component Ensembles" is coordinated by MW; Joseph and his group are responsible for the work package on "Correctness of Service Components and Service Component Ensembles". Design techniques ensuring correctness-by-construction play a main role in ASCENS; current results comprise an extension of BIP for modeling dynamic architectures [6] and a novel implementation of the D-Finder tool for compositional deadlock detection in concurrent systems [5].

Working with Joseph is an excellent experience; we admire his deep insights and technical precision, and are looking forward to many further inspiring exchanges.

## 2 Assembly Theories

We develop a general framework that is intended to capture and formalize rudimentary properties that we believe should be satisfied by any concrete framework for distributed component systems to form a reasonable assembly theory. For this purpose we will consider some abstract domains, operators, relations and laws that altogether form a meta-theory for assemblies. Our starting assumption is that an assembly consists of a finite set of components which can interact. Since components are encapsulated units, we represent them by interface specifications (shortly called interfaces). Therefore we consider an assembly as a (non-empty) finite set of interface specifications which fit syntactically together according to some composability criterion.[1] In the following we are interested to collect a number of general properties that must be satisfied by a concrete framework to form an assembly theory.

As a basis we assume given a class $\mathcal{F}$ of *interface specifications* together with a reflexive and transitive *interface refinement* relation $\preccurlyeq \subseteq \mathcal{F} \times \mathcal{F}$. For two interfaces $F$ and $G$, $F \preccurlyeq G$ means that $F$ is a refinement of the interface specification $G$. We denote by $\wp_{\mathrm{fin}}(\mathcal{F})$ the class of the finite subsets of $\mathcal{F}$. In general, not all elements of $\wp_{\mathrm{fin}}(\mathcal{F})$ form assemblies. Usually there are some syntactic composability conditions required for the members of an assembly. Hence, any assembly theory must first define a particular class $\mathcal{A} \subseteq \wp_{\mathrm{fin}}(\mathcal{F})$ whose elements form valid interface assemblies. We require that any assembly must have at least one element, that any interface induces a (singleton) assembly and that non-empty subsets of assemblies are assemblies as well. These conditions are stated in the first item of Def. 1. In order to combine assemblies to larger ones we require a partial assembly composition operator $\boxtimes$ which is defined, if and only if, the union of two assemblies is an admissible assembly again. We require that an assembly theory must offer a packing operation $pack : \mathcal{A} \to \mathcal{F}$, which allows us to encapsulate an assembly into a component interface by hiding the internals of the assembly. Thus hierarchical assemblies can be constructed by using packed assemblies as their components. To address behavioral compatibility of the interacting members of an assembly, we introduce a communication-safety predicate $cs \subseteq \mathcal{A}$ on assemblies.

[1] To be as abstract as possible, we deliberately take this simplified view not considering other ingredients like ports, connectors etc.

Similarly to interfaces, an assembly theory must also offer a reflexive and transitive refinement relation for assemblies, denoted by $\sqsubseteq \subseteq \mathcal{A} \times \mathcal{A}$.

Some crucial properties relating composition, encapsulation, communication-safety, and refinement are required for any concrete assembly theory. The properties (A1), (A2) and (A3) specify in their (a) part straightforward rules for singleton assemblies. Their (b) and (c) parts state compositionality requirements: (A1)(c) states that communication-safe assemblies can be packed piecewise if the two components obtained from packing $A$ and $B$ are communication-safe. At the end the still visible boundary of the single packed assemblies must be hidden by applying another pack.

(A2)(b) deals with compositionality of communication-safety. If two communication-safe assemblies $A$ and $B$ are combinable, then for the result to be communication-safe it suffices to check that the two components obtained from packing $A$ and $B$ are communication-safe. Hence, once $A$ and $B$ are locally "fine", it only remains to consider the interactions on the boundary between $A$ and $B$. This important property supports also efficient communication-safety checking, since in concrete applications it is often possible to consider minimized versions of $pack(A)$ and $pack(B)$. (A3)(b) and (c) formulate a compositionality requirement for refinement of communication-safe assemblies. In part (b) local refinements are given and the other assumptions are the same as for (A1)(c) and (A2)(b). (A4) is straightforward requiring that encapsulation of communication-safe assemblies, which are in refinement relation, leads to interfaces which are also in refinement relation. Another important property is expressed by (A5) guaranteeing that refinements of communication-safe assemblies are communication-safe.

**Definition 1 (Assembly theory).** *An* assembly theory $(\mathcal{A}, \boxtimes, pack, cs, \sqsubseteq)$ *over* $(\mathcal{F}, \preccurlyeq)$ *is given by*

- *a class* $\mathcal{A} \subseteq \wp_{\mathrm{fin}}(\mathcal{F})$ *of* assemblies*, such that*
  1. $\emptyset \notin \mathcal{A}$*,*
  2. *for all* $F \in \mathcal{F}, \{F\} \in \mathcal{A}$*, and*
  3. $\mathcal{A}$ *is closed under the formation of non-empty subsets, i.e., if* $A \in \mathcal{A}$ *and* $\emptyset \neq B \subseteq A$*, then* $B \in \mathcal{A}$*;*
- *a partial* assembly composition *operator* $\boxtimes : \mathcal{A} \times \mathcal{A} \rightharpoonup \mathcal{A}$ *defined by* $A \boxtimes B = A \cup B$ *if* $A \cup B \in \mathcal{A}$*, undefined otherwise;*[2]
- *an* encapsulation *operation* $pack : \mathcal{A} \to \mathcal{F}$*,*
- *a* communication-safety *predicate* $cs \subseteq \mathcal{A}$ *(we will write* $cs(A)$ *for* $A \in cs$*), and*
- *a reflexive and transitive* assembly refinement *relation* $\sqsubseteq \subseteq \mathcal{A} \times \mathcal{A}$*,*

*such that for all* $F, G \in \mathcal{F}$ *and* $A, B, A_1, A_2, B_1, B_2 \in \mathcal{A}$ *the following holds:*

*A1.* Compositionality of encapsulation:
  *(a)* $pack(\{F\}) = F$*.*
  *(b) If* $A \boxtimes B$ *is defined, then* $\{pack(A)\} \boxtimes \{pack(B)\}$ *is defined.*

[2] $\boxtimes$ is commutative in the sense that for all $A, B \in \mathcal{A}$, if $A \boxtimes B$ is defined then $B \boxtimes A$ is defined and $A \boxtimes B = B \boxtimes A$. $\boxtimes$ is also associative in the sense that for all $A, B, C \in \mathcal{A}$, if $A \boxtimes B$ and $(A \boxtimes B) \boxtimes C$ are defined, then $B \boxtimes C$ and $A \boxtimes (B \boxtimes C)$ are defined and $(A \boxtimes B) \boxtimes C = A \boxtimes (B \boxtimes C)$. This follows from the subset-closedness condition.

(c) *If* $A \boxtimes B$ *is defined, and if* $cs(A)$, $cs(B)$ *and* $cs(\{pack(A)\} \boxtimes \{pack(B)\})$, *then* $pack(A \boxtimes B) = pack(\{pack(A)\} \boxtimes \{pack(B)\})$.

*A2.* Compositionality of communication-safety:

(a) $cs(\{F\})$.

(b) *If* $A \boxtimes B$ *is defined, and if* $cs(A)$, $cs(B)$ *and* $cs(\{pack(A)\} \boxtimes \{pack(B)\})$, *then* $cs(A \boxtimes B)$.

*A3.* Compositionality of refinement:

(a) *If* $F \preccurlyeq G$, *then* $\{F\} \sqsubseteq \{G\}$.

(b) *If* $B_1 \boxtimes B_2$ *is defined, and if* $A_i \sqsubseteq B_i$ *for* $i \in \{1, 2\}$, *then* $A_1 \boxtimes A_2$ *is defined.*

(c) *If* $B_1 \boxtimes B_2$ *is defined, and if* $cs(B_1)$, $cs(B_2)$, $cs(\{pack(B_1)\} \boxtimes \{pack(B_2)\})$, *and if* $A_i \sqsubseteq B_i$ *for* $i \in \{1, 2\}$, *then* $A_1 \boxtimes A_2 \sqsubseteq B_1 \boxtimes B_2$.

*A4.* Preservation of refinement by encapsulation:
*If* $A \sqsubseteq B$ *and* $cs(B)$, *then* $pack(A) \preccurlyeq pack(B)$.

*A5.* Preservation of communication-safety by refinement:
*If* $A \sqsubseteq B$ *and* $cs(B)$, *then* $cs(A)$.

From the laws of an assembly theory it follows that communication-safe assemblies can be constructed in an incremental manner, i.e. by enlarging the assembly by one interface at a time, each time checking that the packed assembly up to now is communication-safe with the additional interface.

*Incremental design:* Let $A \in \mathcal{A}$ be an assembly and let $F \in \mathcal{F}$ such that $A \cup \{F\} \in \mathcal{A}$. If $cs(A)$ and $cs(\{pack(A), F\})$, then $cs(A \cup \{F\})$.

Similarly, the following law of independent implementability is also a consequence of the properties of an assembly theory.

*Independent implementability:* Let $A, B \in \mathcal{A}$ such that $A \sqsubseteq B$ and let $F, G \in \mathcal{F}$ such that $F \preccurlyeq G$ and $B \cup \{G\} \in \mathcal{A}$. If $cs(B)$ and $cs(\{pack(B), G\})$, then $A \cup \{F\} \sqsubseteq B \cup \{G\}$.

The idea to consider assemblies as sets of components, automata, or interfaces is present in many approaches in the literature; see e.g. CFSMs [7], the BIP framework [3,13], team automata [8], component-interaction automata [9], and modal assemblies [15]. So it is an interesting question to what extent the concepts and laws from above appear in the different frameworks. In [7] communication protocols are studied based on collections of communicating finite state machines. Communication is asynchronous via queues and the focus there is particularly on communication properties, like specified reception (and how to check this), which could be used as a communication-safety predicate in our sense. In the BIP framework systems of components are considered together with particular interaction models, which are not (yet) incorporated into our notion of an assembly. BIP provides, as required for an assembly theory, a composition operator, it deals with certain properties of systems, like interaction safety, and focuses on compositionality results much in the spirit of an assembly theory. Compositionality results are also studied in [8] for systems of reactive transition systems (playing the role of interfaces). Our notion of an assembly could be instantiated by the concept of a composable system, and communication-safety by the notion of a compatible system. Different synchronization strategies are applicable and interpreted

via team automata. For the case of the synchronous product, [8] shows, in Cor. 9, a compositionality result, which is very similar to property (A2) required for assembly theories. In [9] systems of composable component-interaction automata are used as assemblies. [9] focuses merely on substitutability of components which is very much related to our principle of independent implementability. Communication-safety is not an issue there. In [15] both communication-safety and refinement are studied using modal I/O-transition systems (MIOs) as interfaces and systems of connectable MIOs as assemblies. We have defined there an ad hoc refinement relation for assemblies requiring that the interfaces of abstract and concrete assemblies must be related by pairwise interface refinements. This refinement relation is, however, far too restrictive and we will propose a more flexible one in Sect. 4.3. We will see that the new assembly refinement relation needs a reconsideration of the behavior model for assemblies indicating communication errors such that the property (A5) for preservation of communication-safety is satisfied.

## 3 Modal I/O-Transition Systems and Weak Modal Refinement

We give a short introduction to modal I/O-transition systems (MIOs) and their refinement. MIOs will be used hereafter as a basic framework to build a modal assembly theory. Modal transition systems (MTS) have been introduced in [18] and later extended by Input/Output alphabets in [17]. As a verification tool we use the MIO-workbench presented first in [4]. We have chosen MIOs as our basic formalism since they allow us to distinguish between transitions which are optional (*may*) or mandatory (*must*) and thus support very well loose specifications and refinements. Like any labeled transition system also MIOs model actions by labels on the transitions. We distinguish four kinds of actions and hence labels: input labels, output labels, communication labels and the internal action $\tau$. In contrast to Larsen et al. [17], internal actions are not explicitly named here but represented by the invisible action $\tau$ and communication labels are added in our approach to model synchronous communication.

Each MIO is based on an *I/O-labeling* $L = (I_L, O_L, T_L)$ consisting of pairwise disjoint sets of *input labels* $I_L$, *output labels* $O_L$, and *communication labels* $T_L$, such that $\tau \notin I_L \cup O_L \cup T_L$. We write $\bigcup L$ for the set $I_L \cup O_L \cup T_L$ of all labels of $L$. The I/O-labeling of a MIO will be pictorially shown on its frame. For easier readability, input labels will be suffixed with "?" and output labels with "!" on the transitions.

A *modal I/O-transition system* $M = (L_M, S_M, s_{0,M}, \dashrightarrow_M, \longrightarrow_M)$ consists of an I/O-labeling $L_M = (I_M, O_M, T_M)$, a set of *states* $S_M$, an *initial state* $s_{0,M} \in S_M$, a *may-transition relation* $\dashrightarrow_M \subseteq S_M \times (\bigcup L_M \cup \{\tau\}) \times S_M$, and a *must-transition relation* $\longrightarrow_M \subseteq \dashrightarrow_M$, i.e. any must-transition is also a may-transition. A MIO $M$ is called an *implementation* if all transitions are must-transitions, i.e. $\longrightarrow_M = \dashrightarrow_M$. The set of the *reachable states* from the initial state $s_{0,M}$ of $M$ w.r.t. may-transitions is denoted by $\mathscr{R}(M)$. For $l \in \bigcup L_M \cup \{\tau\}$, we write $s \overset{l}{\dashrightarrow}_M s'$ for $(s, l, s') \in \dashrightarrow_M$ and $s \overset{l}{\longrightarrow}_M s'$ for $(s, l, s') \in \longrightarrow_M$. Since $\longrightarrow_M \subseteq \dashrightarrow_M$, $s \overset{l}{\longrightarrow}_M s'$ implies $s \overset{l}{\dashrightarrow}_M s'$.

We consider two operators on MIOs, synchronous composition and hiding of communication labels.

*Synchronous composition.* Two MIOs $M, N$ with labelings $L_M = (I_M, O_M, T_M)$ and $L_N = (I_N, O_N, T_N)$ resp. are composable, if their labels overlap only on complementary

types, i.e. $\bigcup L_M \cap \bigcup L_N = (I_M \cap O_N) \cup (I_N \cap O_M)$. Hence, whenever a label is shared, then it is either an input label of the first MIO and an output label of the second or conversely. The synchronous composition of two composable MIOs $M$ and $N$ is denoted by $M \otimes^{\text{sy}} N$ and defined as the usual product of automata such that transitions with shared actions are performed (only) simultaneously. After composition the shared labels become communication labels. A synchronization transition in $M \otimes^{\text{sy}} N$ is a must-transition only if both of the single synchronizing transitions are must-transitions.

Composability and synchronous composition are straightforwardly extended to finite sets of MIOs: A non-empty finite set $A = \{M_1, \ldots, M_n\}$ of MIOs is composable, if the single MIOs $M_i$ are pairwise composable. Then labels of each $M_i$ can only be shared with at most one other MIO $M_j$ ($j \neq i$). Since the synchronous composition is commutative and associative (up to a bijection between states) the synchronous composition of $A$ can be inductively defined by $\bigotimes^{\text{sy}} A = M_1 \otimes^{\text{sy}} \ldots \otimes^{\text{sy}} M_n$.

*Hiding.* Hiding is used to build abstractions of labeled transition systems. Usually, hiding is obtained by considering a specified set of previously visible actions as invisible. In the case of MIOs we use a simple, uniform hiding operator which makes communication (obtained by previous compositions) invisible.

Formally, the *hiding* of communication labels of an I/O-labeling $L = (I_L, O_L, T_L)$ is given by $L\xi = (I_L, O_L, \emptyset)$ and the *hiding* of communications labels of a MIO $M$, denoted by $M\xi$, is defined by moving all communication labels on the transitions of $M$ to $\tau$.

*Weak modal refinement.* The basic idea of modal refinement is that required (*must*) transitions of an abstract specification must also occur in the concrete specification. Conversely, allowed (*may*) transitions of the concrete specification must be allowed by the abstract specification, but can be omitted in the concrete one. We will use the weak form of modal refinement introduced by Hüttel and Larsen in [16] which supports observational abstraction, i.e., internal transitions can be dropped and inserted as long as the modalities and the simulation relation are preserved. Their definition assumes distinguished sets of external and internal actions; here, external actions are given by the input, output and communication labels of MIOs and the internal actions are given by the single label $\tau$. Since communication labels are considered to be visible, they must be respected in the same way as input/output labels. This is important when we consider assembly refinement which should respect communications.

For denoting sequences of transitions that abstract from silent transitions, we use the following notation. Let $M$ be a MIO with I/O-labeling $L_M = (I_M, O_M, T_M)$.

1. We write $s \overset{\widehat{\tau}}{\dashrightarrow}_M s'$ if there is a (possibly empty) sequence of may-transitions from $s$ to $s'$ all labeled by $\tau$, and likewise for must-transitions. For $l \in \bigcup L_M$, we write $s \overset{\widehat{l}}{\dashrightarrow}_M s'$ for $s \overset{\widehat{\tau}}{\dashrightarrow}_M r \overset{l}{\dashrightarrow}_M t \overset{\widehat{\tau}}{\dashrightarrow}_M s'$, and likewise for must-transitions.
2. To express that a sequence of transitions is obtained by an arbitrary order of single transitions involving only labels of a given set $X \subseteq \bigcup L_M$ or $\tau$, we write $s \overset{\widehat{X}}{\dashrightarrow}_M s'$ for $s \overset{\widehat{l_1}}{\dashrightarrow}_M \cdots \overset{\widehat{l_n}}{\dashrightarrow}_M s'$ with $n \geq 0$ and $l_1, \ldots, l_n \in X$.

Let $M$ and $N$ be MIOs with the same I/O-labeling. A relation $R \subseteq S_M \times S_N$ is a *weak modal refinement relation* between $M$ and $N$ if for all $(s_M, s_N) \in R$ and for all $l \in \bigcup L_M = \bigcup L_N$ the following holds:

R1. $s_N \xrightarrow{l}_N s'_N \Rightarrow \exists s'_M \in S_M \,.\, s_M \xrightarrow{\widehat{l}}_M s'_M \wedge (s'_M, s'_N) \in R.$
R2. $s_N \xrightarrow{\tau}_A s'_N \Rightarrow \exists s'_M \in S_M \,.\, s_M \xrightarrow{\widehat{\tau}}_M s'_M \wedge (s'_M, s'_N) \in R.$
R3. $s_M \overset{l}{\dashrightarrow}_M s'_M \Rightarrow \exists s'_N \in S_N \,.\, s_N \overset{\widehat{l}}{\dashrightarrow}_N s'_N \wedge (s'_M, s'_N) \in R.$
R4. $s_M \overset{\tau}{\dashrightarrow}_M s'_M \Rightarrow \exists s'_N \in S_N \,.\, s_N \overset{\widehat{\tau}}{\dashrightarrow}_N s'_N \wedge (s'_M, s'_N) \in R.$

Recall that any must-transition is also a may-transition. Hence, by (R3) and (R4), must-transitions in $M$ must be allowed by corresponding may-transitions in $N$.

$M$ is a *weak modal refinement* of $N$, written $M \leq^*_{\mathrm{m}} N$, if there exists a weak modal refinement relation $R$ between $M$ and $N$ such that $(s_{0,M}, s_{0,N}) \in R$. If all transitions of $M$ and $N$ are must-transitions, weak modal refinement coincides with weak bisimulation. Obviously, weak modal refinement is reflexive and transitive. Two MIOs $M$ and $N$ are *equivalent*, written $M \approx^*_{\mathrm{m}} N$, if $M \leq^*_{\mathrm{m}} N$ and $N \leq^*_{\mathrm{m}} M$, i.e. $M$ co-simulates $N$.

Weak modal refinement is preserved by synchronous composition and by the hiding operator. The first statement extends the compositionality result of [16] to the case of products with visible communication labels. The second statement follows from the fact that any weak modal refinement relation witnessing $M \leq^*_{\mathrm{m}} N$ is also a weak modal refinement relation witnessing $M\xi \leq^*_{\mathrm{m}} N\xi$.

**Proposition 1 (Preservation of weak modal refinement).**

1. *For $i = 1, 2$, let $M_i, N_i$ be MIOs such that $M_i \leq^*_{\mathrm{m}} N_i$ and let $M_1$ and $M_2$ (and hence $N_1$ and $N_2$) be composable. Then $M_1 \otimes^{\mathrm{sy}} M_2 \leq^*_{\mathrm{m}} N_1 \otimes^{\mathrm{sy}} N_2$.*
2. *Let $M, N$ be MIOs such that $M \leq^*_{\mathrm{m}} N$. Then $M\xi \leq^*_{\mathrm{m}} N\xi$.* □

## 4 A Modal Assembly Theory

### 4.1 Modal Interfaces and Modal Assemblies

A *modal interface* $F$ is given by a modal I/O-transition system (MIO), whose I/O-labeling $L_F = (I_F, O_F, \emptyset)$ does not show communication labels. The labeling restriction to the empty set of communication labels reflects the blackbox characteristics of modal interfaces abstracting from communication. The class of all modal interfaces is denoted by $\mathcal{F}^{\mathrm{m}}$. The notion of weak modal refinement (see Sect. 3) is directly applicable to define refinement for modal interfaces. A modal interface $F$ *refines* a modal interface $G$, written $F \preccurlyeq^{\mathrm{m}} G$, if $F \leq^*_{\mathrm{m}} G$.[3]

We will now build a modal assembly theory over $(\mathcal{F}^{\mathrm{m}}, \preccurlyeq^{\mathrm{m}})$. Only those (finite) sets of modal interfaces are allowed to form a modal assembly, whose members are pairwise composable; see Sect. 3.

[3] The notation distinguishes between $\preccurlyeq^{\mathrm{m}}$ and $\leq^*_{\mathrm{m}}$ (Sect. 3), since $\preccurlyeq^{\mathrm{m}}$ is only applicable if the labelings do not show communication labels.

**Definition 2 (Modal assemblies).** *The class* $\mathcal{A}^{\mathrm{m}} \subseteq \wp_{\mathrm{fin}}(\mathcal{F}^{\mathrm{m}})$ *of* modal assemblies *consists of all non-empty, composable (and hence finite) subsets* $A \subseteq \mathcal{F}^{\mathrm{m}}$. □

This satisfies the requirements of Def. 1, since (1) $\emptyset \notin \mathcal{A}^{\mathrm{m}}$, (2) $\{F\} \in \mathcal{A}^{\mathrm{m}}$ for all $F \in \mathcal{F}^{\mathrm{m}}$, and (3) $\mathcal{A}^{\mathrm{m}}$ is closed under non-empty subsets. The composition of two modal assemblies $A$ and $B$ is denoted by $A \boxtimes^{\mathrm{m}} B$. Hence $A \boxtimes^{\mathrm{m}} B = A \cup B$ if $A \cup B \in \mathcal{A}^{\mathrm{m}}$ and undefined otherwise. Fig. 1 show the pictorial representation of a modal assembly consisting of three pairwise composable interfaces $F_1$, $F_2$, and $F_3$.

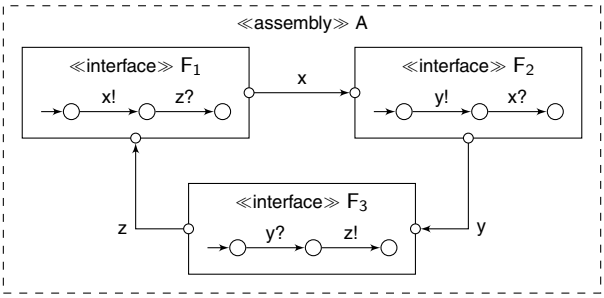



**Fig. 1.** A communication-safe modal assembly

### 4.2 Communication-Safety and Encapsulation of Modal Assemblies

In this work we define a communication-safety notion which is equivalent to the one in [15]. We will, however, use a different technical definition based on the explicit introduction of MIOs with error states. With this new definition we can generalize weak modal refinement to take into account errors states, which will lead to a new and powerful refinement notion for modal assemblies such that communication-safety is preserved. Our notion of communication-safe assembly is inspired by the notion of weak modal compatibility in [4]. This compatibility notion, as well as the compatibility notions in [10,12] and [17], rely on the assumption that outputs are autonomous and must be accepted by a communication partner while inputs are subject to external choice and need not to be served. Hence the discrimination of inputs and outputs is essential. Strong modal compatibility is based on the idea that whenever one component wants to send an output it finds the communication partner in a state, in which it must take the corresponding input *immediately*. Weak modal compatibility is more liberal, since it is sufficient if the communication partner must accept the message possibly after performing first some silent must-transitions. But in practice this compatibility requirement is still too strong. Therefore we generalize weak compatibility further and allow the communication partner to take the input only after performing silent must-transitions *and/or* mandatory communications with other components of the assembly *and/or* outputs on must-transitions which are directed outside of the assembly. This works well because, assuming communication-safe developments, these (open) outputs are again guaranteed to be taken, possibly after a delay, when an assembly is further extended.

For the technical definition of communication-safety we will first introduce a formal definition of the behavior of a modal assembly, which will be represented by a MIO

extended by explicit error states in the case that communication errors occur during execution of the assembly. Let $A = \{M_1, \ldots, M_n\}$ be a composable set of MIOs and let $M_j \in A$. Then the rest $A \setminus \{M_j\}$ plays the role of the environment for $M_j$. We must ensure that in any reachable state of the product $\bigotimes^{\mathrm{sy}} A$, whenever $M_j$ wants to send an output $l$, then $Env_j = \bigotimes^{\mathrm{sy}} A \setminus \{M_j\}$ must be able to take $l$ as an input possibly after some autonomous must-transitions which do not concern the communication with $M_j$. These autonomous transitions can be silent must-transitions of $Env_j$ or must-communication transitions of $Env_j$ obtained from communication inside $A \setminus \{M_j\}$, but also must-outputs of $Env_j$ are admitted which are not shared with the inputs of $M_j$. If such a sequence of autonomous actions *cannot* be performed by $Env_j$ there is a communication error.

**Definition 3 (Communication errors).** *Let $A = \{M_1, \ldots, M_n\}$ be a composable set of MIOs (with $n \geq 1$). If $n = 1$ there is no communication error. Otherwise, for each $1 \leq j \leq n$, let $Env_j = \bigotimes^{\mathrm{sy}} A \setminus \{M_j\}$. The* communication errors *$\mathscr{E}(A)$ are given by the set of pairs $((s_1, \ldots, s_n), l)$ such that $(s_1, \ldots, s_n) \in \mathscr{R}(\bigotimes^{\mathrm{sy}} A)$ and there is $1 \leq j \leq n$ with $l \in O_{M_j} \cap I_{Env_j}$, a state $s'_j \in S_{M_j}$ with $s_j \overset{l}{\dashrightarrow}_{M_j} s'_j$ but there are no transitions*

$$(s_1, \ldots, s_{j-1}, s_{j+1}, \ldots, s_n) \xrightarrow{\widehat{X}_j}_{Env_j} \cdot \xrightarrow{l}_{Env_j} (s'_1, \ldots, s'_{j-1}, s'_{j+1}, \ldots, s'_n)$$

*with $X_j = T_{Env_j} \cup (O_{Env_j} \setminus I_{F_j})$.*[4] □

Note that only communication errors occurring in the reachable part of the synchronous product of $A$ are considered.

**Definition 4 (MIOs with error states).** *A* MIO with error states *(*EMIO*) is a pair $(M, E)$ consisting of a MIO $M$ and a set of* error states *$E \subseteq S_M$.* □

The error composition of MIOs is obtained by taking their synchronous product enriched by error states (if there are any) which are then reached by the un-accepted communication labels $l$. The idea is similar to the consent operator introduced in [1] to compose languages by indicating communication errors in traces.

**Definition 5 (Error-composition of MIOs).** *Let $A = \{M_1, \ldots, M_n\}$ be a composable set of MIOs and let $P = \bigotimes^{\mathrm{sy}} A$. The* error-composition *of $A$ is given by the EMIO*

$$\bigotimes^{\mathrm{err}} A = ((L_P, S_P \cup \mathscr{E}(A), s_{0,P}, \dashrightarrow, \longrightarrow_P), \mathscr{E}(A))$$

*with may-transition relation $\dashrightarrow = \dashrightarrow_P \cup \{(p, l, (p, l)) \mid (p, l) \in \mathscr{E}(A)\}$.* □

The behavior of a modal assembly is given by the error composition of the modal interfaces of the assembly. It may also be considered as the semantics of the assembly. If no communication-error state appears in the assembly behavior, the assembly is communication-safe.

[4] Recall that $T_{Env_j}$ are the communication labels of $Env_j$ and $(O_{Env_j} \setminus I_{M_j})$ the output labels of $Env_j$ unshared with the input labels of $M_j$, i.e., not used for communication between $Env_j$ and $M_j$. The silent must-transitions of $Env_j$ are anyway subsumed in the notation $\xrightarrow{\widehat{X}_j}_{Env_j}$; see Sect. 3.

**Definition 6 (Behavior of a modal assembly and communication-safety).** *Let* $A = \{F_1, \ldots, F_n\} \in \mathcal{A}^{\mathrm{m}}$ *be a modal assembly. The* behavior *of* $A$ *is given by* $beh(A) = \bigotimes^{\mathrm{err}} A$. *$A$ is* communication safe*, written as* $cs^{\mathrm{m}}(A)$*, if* $\mathscr{E}(A) = \emptyset$. □

As an example, consider the assembly A in Fig. 1 and its (reachable) behavior shown in Fig. 2. The assembly is communication-safe since there is no error state. In fact all interfaces will be able to send their messages, possibly after a delay. For instance, $\mathsf{F}_1$ can send x to $\mathsf{F}_2$ after $\mathsf{F}_2$ has communicated the message y to $\mathsf{F}_3$.

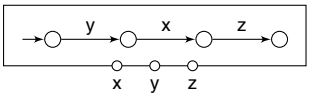



**Fig. 2.** Behavior of the assembly A in Fig. 1

Consider now a slight variation of the assembly in Fig. 1 such that the order of the input y? and output z! in $\mathsf{F}_3$ is reversed. Let us call this assembly $\mathsf{A}'$; see Fig. 3. The EMIO representing the (reachable) behavior of $\mathsf{A}'$ is shown in Fig. 4; it contains three error states. These are induced by the cyclic wait of the single interfaces in $\mathsf{A}'$. Hence the assembly $\mathsf{A}'$ is not communication-safe. This example shows also that one cannot deduce from pairwise communication-safety of the interfaces of an assembly that the whole assembly is communication-safe. Indeed all pairs of interfaces in $\mathsf{A}'$ would form a communication-safe assembly.

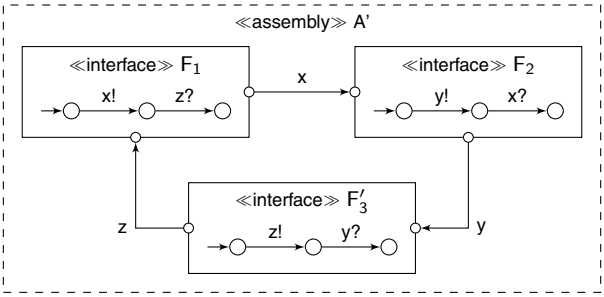



**Fig. 3.** Communication-safety does not follow from pairwise communication-safety

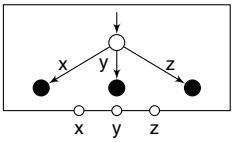
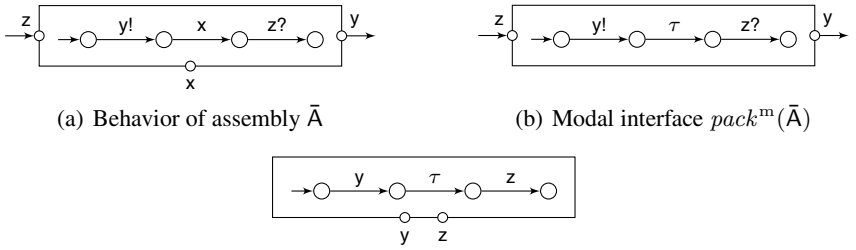



**Fig. 4.** Behavior of the assembly $\mathsf{A}'$

The encapsulation of a modal assembly $A$ by means of the modal pack operator is simply defined by hiding communication labels (see Sect. 3) in the behavior of $A$ and forgetting the explicit discrimination of error states.




(a) Behavior of assembly $\bar{\mathsf{A}}$

(b) Modal interface $pack^{\mathrm{m}}(\bar{\mathsf{A}})$

(c) Behavior of assembly $\bar{\mathsf{C}}$

**Fig. 5.** Modal behaviors for assembly $\mathsf{A}$ in Fig. 1

**Definition 7 (Encapsulation of modal assemblies).** *The modal pack operator* $pack^{\mathrm{m}} : \mathcal{A}^{\mathrm{m}} \to \mathcal{F}^{\mathrm{m}}$ *is defined by* $pack^{\mathrm{m}}(A) = M\xi$ *with* $(M, E) = beh(A)$. □

We have to verify that the required properties (A1) and (A2) of an assembly theory are satisfied. First, we check (A1): (a) $pack^{\mathrm{m}}(\{F\}) = F$ holds by definition. (b) Let $A$ and $B$ be modal assemblies such that $A \boxtimes^{\mathrm{m}} B$ is defined; then $\{pack^{\mathrm{m}}(A), pack^{\mathrm{m}}(B)\} \in \mathcal{A}^{\mathrm{m}}$ since the MIOs underlying $beh(A)$ and $beh(B)$ are composable and since hiding preserves composability. (c) Now let additionally $cs^{\mathrm{m}}(A)$, $cs^{\mathrm{m}}(B)$, and $cs^{\mathrm{m}}(\{pack^{\mathrm{m}}(A), pack^{\mathrm{m}}(B)\})$ hold. Let $(M_A, E_A) = beh(A)$, $(M_B, E_B) = beh(B)$, $(M_{AB}, E_{AB}) = beh(\{M_A\xi, M_B\xi\})$, and $(M_{A\cup B}, E_{A\cup B}) = beh(A \cup B)$; then $\emptyset = E_A = E_B = E_{A\cup B} = E_{AB}$ and $M_A = \bigotimes^{\mathrm{sy}} A$, $M_B = \bigotimes^{\mathrm{sy}} B$, $M_{A\cup B} = \bigotimes^{\mathrm{sy}}(A \cup B) = M_A \otimes^{\mathrm{sy}} M_B$, and $M_{AB} = M_A\xi \otimes^{\mathrm{sy}} M_B\xi$ by the required communication-safety of $A$, $B$, and $\{pack^{\mathrm{m}}(A), pack^{\mathrm{m}}(B)\}$, and the resulting communication-safety of $A \boxtimes^{\mathrm{m}} B$ using (A2)(b). Thus

$$pack^{\mathrm{m}}(A \boxtimes^{\mathrm{m}} B) = M_{A\cup B}\xi = M_{AB}\xi = pack^{\mathrm{m}}(\{pack^{\mathrm{m}}(A)\} \boxtimes^{\mathrm{m}} \{pack^{\mathrm{m}}(B)\}) .$$

We now check (A2): (a) $cs^{\mathrm{m}}(\{F\})$ is obvious since no communication errors arise from a single modal interface. (b) Let $A$ and $B$ be modal assemblies such that $A \boxtimes^{\mathrm{m}} B$ is defined; then $\{pack^{\mathrm{m}}(A), pack^{\mathrm{m}}(B)\} \in \mathcal{A}^{\mathrm{m}}$ follows from (A1)(b). Now let additionally $cs^{\mathrm{m}}(A)$, $cs^{\mathrm{m}}(B)$, and $cs^{\mathrm{m}}(\{pack^{\mathrm{m}}(A), pack^{\mathrm{m}}(B)\})$ hold. Then $cs^{\mathrm{m}}(A \boxtimes^{\mathrm{m}} B)$ holds, since any communication error in $A \boxtimes^{\mathrm{m}} B$ would show up either in $A$ or in $B$ or at the boundary of $A$ and $B$ which would be captured by a communication error of $\{pack^{\mathrm{m}}(A), pack^{\mathrm{m}}(B)\}$.

Let us demonstrate how the principle of incremental design (see Sect. 2) works for the example assembly in Fig. 1. We start with the assembly $\bar{\mathsf{A}} = \{\mathsf{F}_1, \mathsf{F}_2\}$. The behavior of this assembly is shown in Fig. 5(a). Obviously, $\bar{\mathsf{A}}$ is communication-safe. We now want to add the interface $\mathsf{F}_3$ to $\bar{\mathsf{A}}$. First, we pack the assembly $\bar{\mathsf{A}}$ which yields the modal interface $pack^{\mathrm{m}}(\bar{\mathsf{A}})$ shown in Fig. 5(b). Then we consider the assembly $\bar{\mathsf{C}} = \{pack^{\mathrm{m}}(\bar{\mathsf{A}}), \mathsf{F}_3\}$ whose behavior is shown in Fig. 5(c). Obviously $\bar{\mathsf{C}}$ is communication-safe and therefore, by the law of incremental design, the assembly $\mathsf{A} = \{\mathsf{F}_1, \mathsf{F}_2, \mathsf{F}_3\}$ is also communication-safe. The incremental communication-safety check would, in general, be much more efficient if we would minimize packed assemblies w.r.t. silent transitions.

Consider once more the assembly $\mathsf{A}'$ in Fig. 3 and assume that we want to construct it in an incremental way. Then we could start again with the assembly $\bar{\mathsf{A}} = \{\mathsf{F}_1, \mathsf{F}_2\}$ which is communication-safe. But now, for adding the interface $\mathsf{F}'_3$, we have to consider the assembly $\bar{\mathsf{C}}' = \{pack^{\mathrm{m}}(\bar{\mathsf{A}}), \mathsf{F}'_3\}$ and to check communication-safety. The behavior of $\bar{\mathsf{C}}'$ is shown in Fig. 6; it has two error states. Hence, the incremental design step would not succeed and anyway, as we know from before, the assembly $\mathsf{A}'$ is not communication-safe.

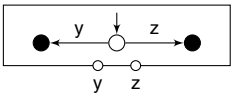
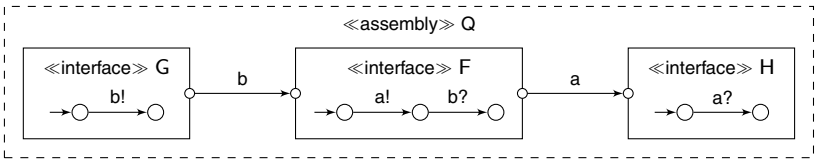
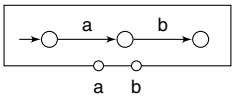



**Fig. 6.** Behavior of assembly $\bar{\mathsf{C}}'$

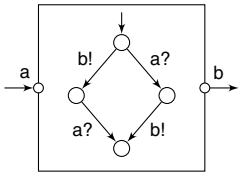
Conversely, we can not deduce from the communication-safety of an assembly $A \boxtimes^{\mathrm{m}} B$ that (i) $\{pack^{\mathrm{m}}(A), pack^{\mathrm{m}}(B)\}$ is communication-safe and we can also not deduce that (ii) the sub-assemblies $A$, $B$ are communication-safe. Hence the converse direction of (A2)(b) does not hold. A counter-example for (i) is shown in Fig. 7(a). We can observe that the assembly $\mathsf{Q}$ is communication-safe; its (reachable) behavior, see Fig. 7(b), contains no error states. If we pack the sub-assembly $\{\mathsf{G}, \mathsf{H}\}$ we obtain the modal interface shown in Fig. 7(c). But the assembly $\{\mathsf{F}, pack^{\mathrm{m}}(\{\mathsf{G}, \mathsf{H}\})\}$ is not communication-safe. The reason is that $pack^{\mathrm{m}}(\{\mathsf{G}, \mathsf{H}\})$ has an output $\mathsf{b!}$ in its initial state, but the interface $\mathsf{F}$ can never accept this particular output as an input. It can only perform an $\mathsf{a}$ communication with $pack^{\mathrm{m}}(\{\mathsf{G}, \mathsf{H}\})$ and then accept "another" $\mathsf{b!}$ output of $pack^{\mathrm{m}}(\{\mathsf{G}, \mathsf{H}\})$ issued in another state.




(a) Modal assembly $\mathsf{Q}$




(b) Behavior of assembly $\mathsf{Q}$




(c) Modal interface $pack^{\mathrm{m}}(\{\mathsf{G}, \mathsf{H}\})$

**Fig. 7.** Counter-example for (i)

A counter-example for (ii) is shown in Fig. 8. The whole assembly $\mathsf{R}$ is communication-safe, but the sub-assembly $\{\mathsf{G}, \mathsf{F}'\}$ is not. The reason is that $\mathsf{G}$ has an output $\mathsf{b!}$ in its initial state, but $\mathsf{F}'$ has an open input $\mathsf{a?}$ before it can accept $\mathsf{b?}$ which is not allowed. (Inputs are not subject to internal choice and we cannot be sure that an environment will serve this input.)

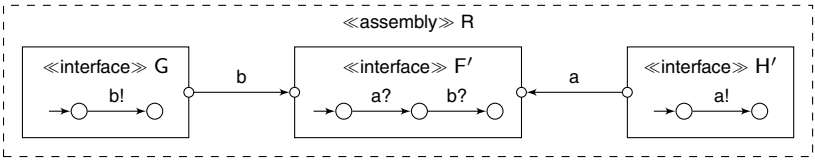



**Fig. 8.** Counter-example for (ii)

### 4.3 Refinement of Modal Assemblies

For the refinement of modal assemblies we compare their behaviors. Since assembly behaviors are MIOs with error states, we first extend the weak modal refinement notion for MIOs (defined in Sect. 3) to EMIOs, such that error states are respected by the refinement relation.

**Definition 8 (Refinement of MIOs with error states).** *Let* $(M_A, E_A)$ *and* $(M_B, E_B)$ *be two EMIOs.* $(M_A, E_A)$ *is a* weak modal refinement *of* $(M_B, E_B)$*, if* $M_A \leq^*_{\mathrm{m}} M_B$ *is a weak modal MIO refinement witnessed by a refinement relation* $R \subseteq ((S_{M_A} \setminus E_A) \times (S_{M_B} \setminus E_B)) \cup (E_A \times E_B)$ *with* $(s_{0,M_A}, s_{0,M_B}) \in R$. □

**Definition 9 (Refinement of modal assemblies).** *A modal assembly* $A$ refines *a modal assembly* $B$*, written as* $A \sqsubseteq^{\mathrm{m}} B$*, if* $beh(A)$ *is a weak modal refinement of* $beh(B)$. □

To get a modal assembly theory it remains to check that the conditions (A3), (A4) and (A5) of an assembly theory are satisfied. We will provide a short proof for each.

(A3): (a) That $F \preccurlyeq^{\mathrm{m}} G$ implies $\{F\} \sqsubseteq^{\mathrm{m}} \{G\}$ is obvious. (b) Now let $B_1 \boxtimes^{\mathrm{m}} B_2$ be defined and let $A_i \sqsubseteq^{\mathrm{m}} B_i$ for $i \in \{1,2\}$. Then $A_1 \boxtimes^{\mathrm{m}} A_2$ is defined, since $A_i$ and $B_i$ have the same I/O-labeling for $i \in \{1,2\}$. (c) Let $cs^{\mathrm{m}}(B_1)$, $cs^{\mathrm{m}}(B_2)$, and $cs^{\mathrm{m}}(\{pack^{\mathrm{m}}(B_1), pack^{\mathrm{m}}(B_2)\})$ hold additionally; then $cs^{\mathrm{m}}(B_1 \boxtimes^{\mathrm{m}} B_2)$ by (A2)(b). Let $(M_{A_i}, E_{A_i}) = beh(A_i)$ and $(M_{B_i}, E_{B_i}) = beh(B_i)$ for $i \in \{1,2\}$, $(M_{A_1 \cup A_2}, E_{A_1 \cup A_2}) = beh(A_1 \cup A_2)$, and $(M_{B_1 \cup B_2}, E_{B_1 \cup B_2}) = beh(B_1 \cup B_2)$. By the communication-safety of $B_1 \boxtimes^{\mathrm{m}} B_2$, $E_{B_1 \cup B_2} = \emptyset$ and hence $M_{B_1 \cup B_2} = \bigotimes^{\mathrm{sy}}(B_1 \cup B_2) = \bigotimes^{\mathrm{sy}} M_{B_1} \otimes^{\mathrm{sy}} \bigotimes^{\mathrm{sy}} M_{B_2}$. From the assumptions $A_i \sqsubseteq^{\mathrm{m}} B_i$, we obtain from (A5), to be proved momentarily, that $cs^{\mathrm{m}}(A_1)$ and $cs^{\mathrm{m}}(A_2)$. Hence, $M_{A_i} = \bigotimes^{\mathrm{sy}} A_i$ and $E_{A_i} = \emptyset$ for $i \in \{1,2\}$. By Prop. 1(1), we have $\bigotimes^{\mathrm{sy}} A_1 \otimes^{\mathrm{sy}} \bigotimes^{\mathrm{sy}} A_2 \leq^*_{\mathrm{m}} \bigotimes^{\mathrm{sy}} B_1 \otimes^{\mathrm{sy}} \bigotimes^{\mathrm{sy}} B_2$. It remains to ensure, that $E_{A_1 \cup A_2} = \emptyset$, i.e., $M_{A_1 \cup A_2} = \bigotimes^{\mathrm{sy}} A_1 \otimes^{\mathrm{sy}} \bigotimes^{\mathrm{sy}} A_2$. But if $(p,l) \in \mathscr{E}(A_1 \cup A_2)$, then $p \in \mathscr{R}(\bigotimes^{\mathrm{sy}} A_1 \otimes^{\mathrm{sy}} \bigotimes^{\mathrm{sy}} A_2)$ and there would be an output may-transition labeled $l$ in one of the interfaces in $A_1 \cup A_2$, say in an interface of $A_1$. Then either $l \in O_{\bigotimes^{\mathrm{sy}} A_1}$ or $l \in T_{\bigotimes^{\mathrm{sy}} A_1}$. By $A_1 \leq^*_{\mathrm{m}} B_1$, such a transition also would have to be available in $B_1$. If $l \in O_{\bigotimes^{\mathrm{sy}} A_1}$, then the output would be accepted using a series of must-transitions in $B_2$ which do not affect $B_1$, since $cs^{\mathrm{m}}(\{pack^{\mathrm{m}}(B_1), pack^{\mathrm{m}}(B_2)\})$; this series of must-transitions would also have to be present in $A_2$ (up to must-$\tau$'s), as $A_2 \leq^*_{\mathrm{m}} B_2$, and hence $(p,l) \notin \mathscr{E}(A_1 \cup A_2)$. If $l \in T_{\bigotimes^{\mathrm{sy}} A_1}$, then the $l$ would be accepted using a series of must-transitions in $A_1$ with possible outputs to $A_2$, since $cs^{\mathrm{m}}(A_1)$, and this series would be present also in $B_1$ by $A_1 \leq^*_{\mathrm{m}} B_1$. Again, these outputs are eventually accepted by $B_2$ by must-transitions, and thus by $A_2$; hence $(p,l) \notin \mathscr{E}(A_1 \cup A_2)$.

(A4): Let $A \sqsubseteq^{\mathrm{m}} B$ and let $cs^{\mathrm{m}}(B)$ hold. Let $(M_A, E_A) = beh(A)$ and $(M_B, E_B) = beh(B)$. Then $cs^{\mathrm{m}}(B)$ implies $E_B = \emptyset$, and thus, by (A5), $E_A = \emptyset$ since $A \sqsubseteq^{\mathrm{m}} B$. $M_A \leq^*_{\mathrm{m}} M_B$ implies $M_A\xi \leq^*_{\mathrm{m}} M_B\xi$ by Prop. 1(2), i.e., $pack^{\mathrm{m}}(A) \preccurlyeq^{\mathrm{m}} pack^{\mathrm{m}}(B)$.

(A5): Let $A \sqsubseteq^{\mathrm{m}} B$ and let $cs^{\mathrm{m}}(B)$ hold. Let $(M_A, E_A) = beh(A) = \bigotimes^{\mathrm{err}} A$ and $(M_B, E_B) = beh(B) = \bigotimes^{\mathrm{err}} B$. If an error state in $E_A$ would be reachable in $M_A$, then $A \sqsubseteq^{\mathrm{m}} B$ would imply that some error state in $E_B$ is also reachable in $M_B$ since error states must be related to error states by a bisimulation. Thus $cs^{\mathrm{m}}(A)$ holds.

## 5 Conclusions

Our study is motivated by an extension of the abstract concepts of interface theories and interface languages, introduced by de Alfaro and Henzinger, to take into account interface assemblies. As a concrete formalism we have chosen modal I/O-transition systems which we have adapted to take into account not only blackbox interface behaviors but also assembly behaviors with distinguished (synchronous) communication actions. We have shown that the compositionality and compatibility requirements of an assembly theory are satisfied by modal assemblies.

In future work we are interested to study more instantiations of assembly theories, in particular assemblies which rely on asynchronous and multi-cast communication, and dynamic assemblies which may dynamically change the number of components and their connections. A concrete assembly theory using asynchronous communication via channel places can already easily be derived from the results for modal I/O-Petri nets in [14]. In this approach communication-safety is expressed by the property of a "necessarily consuming" Petri net. This property is compositional, decidable and preserved by refinement. We also plan to extend the MIO-workbench [4] to check not only MIOs but also modal assemblies and their communication-safety.

## References


1. Adámek, J., Plasil, F.: Component composition errors and update atomicity: Static analysis. J. Softw. Maint. 17(5), 363–377 (2005)
2. ASCENS project, http://www.ascens-ist.eu
3. Basu, A., Bozga, M., Sifakis, J.: Modeling heterogeneous real-time components in BIP. In: Proc. 4th IEEE Int. Conf. Software Engineering and Formal Methods (SEFM 2006), pp. 3–12. IEEE (2006)
4. Bauer, S.S., Mayer, P., Schroeder, A., Hennicker, R.: On weak modal compatibility, refinement, and the MIO workbench. In: Esparza, J., Majumdar, R. (eds.) TACAS 2010. LNCS, vol. 6015, pp. 175–189. Springer, Heidelberg (2010)
5. Bensalem, S., Griesmayer, A., Legay, A., Nguyen, T.-H., Sifakis, J., Yan, R.: D-Finder 2: Towards efficient correctness of incremental design. In: Bobaru, M., Havelund, K., Holzmann, G.J., Joshi, R. (eds.) NFM 2011. LNCS, vol. 6617, pp. 453–458. Springer, Heidelberg (2011)
6. Bozga, M., Jaber, M., Maris, N., Sifakis, J.: Modeling dynamic architectures using Dy-BIP. In: Gschwind, T., De Paoli, F., Gruhn, V., Book, M. (eds.) SC 2012. LNCS, vol. 7306, pp. 1–16. Springer, Heidelberg (2012)
7. Brand, D., Zafiropulo, P.: On communicating finite-state machines. J. ACM 30(2), 323–342 (1983)

8. Carmona, J., Kleijn, J.: Compatibility in a multi-component environment. Theor. Comput. Sci. 484, 1–15 (2013)
9. Cerná, I., Vareková, P., Zimmerova, B.: Component substitutability via equivalencies of component-interaction automata. Electr. Notes Theor. Comput. Sci. 182, 39–55 (2007)
10. de Alfaro, L., Henzinger, T.A.: Interface automata. In: Proc. 9th ACM SIGSOFT Ann. Symp. Foundations of Software Engineering (FSE 2001), pp. 109–120 (2001)
11. de Alfaro, L., Henzinger, T.A.: Interface theories for component-based design. In: Henzinger, T.A., Kirsch, C.M. (eds.) EMSOFT 2001. LNCS, vol. 2211, pp. 148–165. Springer, Heidelberg (2001)
12. de Alfaro, L., Henzinger, T.A.: Interface-based design. In: Broy, M., Grünbauer, J., Harel, D., Hoare, C.A.R. (eds.) Engineering Theories of Software-intensive Systems. NATO Science Series: Mathematics, Physics, and Chemistry, vol. 195, pp. 83–104. Springer (2005)
13. Gößler, G., Sifakis, J.: Composition for component-based modeling. Sci. Comput. Program. 55(1-3), 161–183 (2005)
14. Haddad, S., Hennicker, R., Møller, M.H.: Specification of asynchronous component systems with Modal I/O-Petri nets. In: Abadi, M., Lluch Lafuente, A. (eds.) TGC 2013. LNCS, vol. 8358. Springer (to appear, 2014)
15. Hennicker, R., Knapp, A.: Modal interface theories for communication-safe component assemblies. In: Cerone, A., Pihlajasaari, P. (eds.) ICTAC 2011. LNCS, vol. 6916, pp. 135–153. Springer, Heidelberg (2011)
16. Hüttel, H., Larsen, K.G.: The use of static constructs in a modal process logic. In: Meyer, A.R., Taitslin, M.A. (eds.) Logic at Botik 1989. LNCS, vol. 363, pp. 163–180. Springer, Heidelberg (1989)
17. Larsen, K.G., Nyman, U., Wąsowski, A.: Modal I/O automata for interface and product line theories. In: De Nicola, R. (ed.) ESOP 2007. LNCS, vol. 4421, pp. 64–79. Springer, Heidelberg (2007)
18. Larsen, K.G., Thomsen, B.: A modal process logic. In: Proc. 3rd Ann. IEEE Symp. Logic in Computer Science (LICS 1988), pp. 203–210. IEEE (1988)
19. Rausch, A., Reussner, R., Mirandola, R., Plášil, F. (eds.): The Common Component Modeling Example. LNCS, vol. 5153. Springer, Heidelberg (2008)
20. Sifakis, J.: Rigorous system design. Foundations and Trends in Electronic Design Automation 6(4), 293–362 (2013)