Specification of Asynchronous Component Systems with Modal I/O-Petri Nets

Serge Haddad[1(✉)], Rolf Hennicker[2], and Mikael H. Møller[3]

[1] LSV, ENS Cachan & CNRS & Inria, Cachan, France
haddad@lsv.ens-cachan.fr
[2] Ludwig-Maximilians-Universität München, Munich, Germany
[3] Aalborg University, Aalborg, Denmark

Abstract. Modal transition systems are an elegant way to formalise the design process of a system through refinement and composition. Here we propose to adapt this methodology to asynchronous composition via Petri nets. The Petri nets that we consider have distinguished labels for inputs, outputs, internal communications and silent actions and "must" and "may" modalities for transitions. The input/output labels show the interaction capabilities of a net to the outside used to build larger nets by asynchronous composition via communication channels. The modalities express constraints for Petri net refinement taking into account observational abstraction from silent transitions. Modal I/O-Petri nets are equipped with a modal transition system semantics. We show that refinement is preserved by asynchronous composition and by hiding of communication channels. We study compatibility properties which express communication requirements for composed systems and we show that these properties are decidable, they are preserved in larger contexts and also by modal refinement. On this basis we propose a methodology for the specification of distributed systems in terms of modal I/O-Petri nets which supports incremental design, encapsulation of components, stepwise refinement and independent implementability.

1 Introduction

Component-based design is an important field in software engineering. Crucial tasks in the design process concern the stepwise refinement of specifications towards implementations and the formation of component assemblies by composition. Many approaches and formalisms have been proposed for rigorous component-based design supporting different communication styles and different notions of refinement. Among them particular attention has been attracted by modal transition systems introduced by Larsen and Thomsen in 1988 [15] which use distinguished may- and must-transitions to specify allowed and obligatory behaviour and thus provide a flexible basis for refinement. While refinement concerns the vertical dimension of system development, composition concerns the

This work has been partially sponsored by the EU project ASCENS, 257414.

M. Abadi and A. Lluch Lafuente (Eds.): TGC 2013, LNCS 8358, pp. 219–234, 2014.
DOI: 10.1007/978-3-319-05119-2_13,

horizontal dimension in which larger systems are built from smaller ones such that communication requirements must be respected. Communication properties are important when reasoning about distributed mechanisms, algorithms and applications (e.g. management of sockets in UNIX, maintaining unicity of a token in a ring based algorithm, guarantee of email reading, etc).

Petri nets are a natural model for the design of concurrent and distributed systems. They have received a great attention w.r.t. the composition and refinement issues including communication properties. Composition of nets has been addressed via several paradigms. The process algebra approach has been investigated by several works leading to the Petri net algebra [5]. Such an approach is closely related to synchronous composition. In [20] and [21] asynchronous composition of nets is performed via a set of places or, more generally, via a subnet modelling some medium. Then structural restrictions on the subnets are proposed in order to preserve global properties like liveness or deadlock-freeness. In [18] a general composition operator is proposed and its associativity is established. A closely related concept to composition is the one of open Petri nets which has been used in different contexts like the analysis of web services [22]. In parallel, very early works have been done for defining and analyzing refinement of nets; see [6] for a survey. Looking at more recent works, [19] (which is the closest to our contribution) studies the refinement in the context of circuit design. In [19] a notion of correct implementation is introduced which is shown to be compositional. Several works also use an abstraction/refinement paradigm to propose efficient verification methods; see e.g. [8].

In our contribution we want to combine the advantages of modal transition systems with the ability of Petri nets to represent infinite state systems, with their decidability potential and with their way how asynchronous composition is achieved. A natural candidate are *modal Petri nets* introduced in [7] (and later in [2] as a special case of Modal Process Rewrite Systems) which studies modal refinement and decidability results. Surprisingly, to the best of our knowledge, no other approaches to modal Petri nets exist yet. On the other hand, for asynchronous communication, we have recently introduced in [10] *asynchronous I/O-Petri nets*, for which we have analysed several communication properties from the compositionality and decidability point of view. Hence it is an obvious goal to combine, adjust and extend the results achieved in [7] and [10] to a rigorous design methodology that supports the vertical and the horizontal dimension of software development in a uniform and compatible way.

Concerning the vertical dimension we consider modal refinement; for the horizontal dimension we consider asynchronous composition and we focus on the *message consuming* and the *necessarily message consuming* properties which are important requirements to ensure that previously sent messages can or must necessarily be consumed by the communication partner. It turns out that the necessarily message consuming property defined in [10] is in general not preserved by modal refinement. This is due to the fact that our refinement notion supports observational abstraction. Therefore we investigate the new notion of an *observationally weakly fair run* and show that necessarily message consuming

is indeed preserved by modal refinement if we restrict the consumption requirement to all observational weakly fair runs. Due to the fairness requirement the necessarily consuming property is also preserved on the horizontal layer when components are put in compatible contexts. We also show that the new variants of the communication properties are decidable.

This paper is structured as follows: In Sect. 2, we summarise our proposal by means of an illustrating example. Section 3 presents the underlying formal defintions of modal asynchronous I/O-Petri nets (MAIOPNs) and their semantics in terms of modal asynchronous I/O-transition systems (MAIOTSs). In Sect. 4, we consider modal refinement and show that it is compositional. We also show that modal refinement is preserved by channel hiding. In Sect. 5, we study (necessarily) message consuming systems and we present the results on the preservation of the communication properties by composition and by refinement. As a consequence, our framework supports the principle of independent implementability in the sense of [1]. We finish with some concluding remarks in Sect. 6.

2 Illustrating Example

We introduce an illustrating example to motivate our notions of modal asynchronous I/O-Petri nets (MAIOPNs), their composition, hiding and refinement. For this purpose we consider a top down approach to the design of a simple compressing system (inspired by [3]) which is able to receive files for compression and outputs either zip- or jpg-files. We start with an interface specification of the system modelled by the CompressorInterface in Fig. 1a.

The interface specification is presented by a labelled Petri net with distinguished input and output labels and with modalities on the transitions. The label file suffixed with "?" indicates an input action and the labels comprJpg, comprZip suffixed with "!" indicate output actions. Following the idea of modal transition systems introduced by Larsen and Thomsen in [15] transitions are equipped with "must" or "may" modalities. A must-transition, drawn black, indicates that this transition is required for any refinement while a may-transition, drawn white, may also be removed or turned into a must-transition. Models containg only must-transitions represent implementations. In the example it is required that input files must always be received and that the option to produce zipped text files is always available while a refinement may or may not support the production of compressed jpg-files for graphical data. Our interface specification models an infinite state system since an unbounded number of files can be received.

In the next step we propose an architecture for the realisation of the compressing system as shown in Fig. 1b. It is given by an assembly of three connected components, a Controller component which delegates the compression tasks, a GifCompressor component which actually performs the compression of gif-files into jpg-files and a TxtCompressor component which produces zip-files from text files. The single components are connected by unbounded and unordered channels gif, jpg, ... for asynchronous communication.

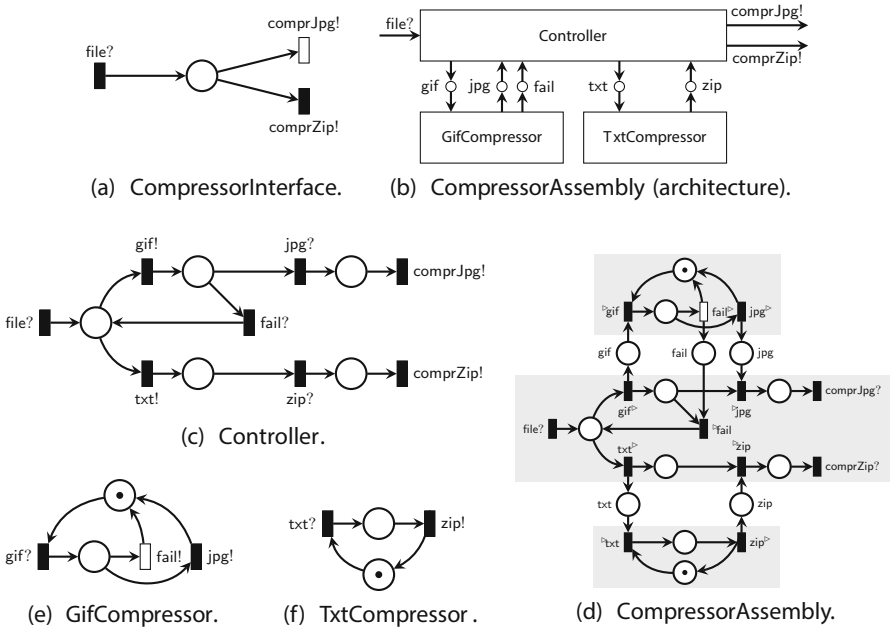


Fig. 1. a CompressorInterface, **b** Compressor Assembly (architecture), **c** Controller, **d** Compressor Assembly, **e** GifCompressor, **f** TxtCompressor.

The behaviour of the single components is modelled by the MAIOPNs shown in Fig. 1c, e, f. The behaviour of the CompressorAssembly is given by the asynchronous composition of the single Petri nets shown in Fig. 1d. For each pair of shared input and output actions a new place is introduced, called communication channel. Transitions with a shared output label a (of a given component) are connected to the new place a and the transition label is renamed to $a^{\triangleright}$ in the composition. Similarly the place a is connected to transitions with the corresponding input label a which is then renamed to ${}^{\triangleright}a$ in the composition. The result of our composition is very similar to the composition of open Petri nets, see e.g. [16], which relies on matching of interface places. But our approach is methodologically different since we introduce the communication places only when the composition is constructed. In that way our basic components are not biased to asynchronous composition but could be used for synchronous composition as well. In this work we focus on asynchronous composition and we are particularly interested in the analysis of generic communication properties ensuring that messages pending on communication channels are eventually consumed. Therefore our notion of modal asynchronous I/O-Petri net will comprise an explicit discrimination of channel places and, additionally to input/output labels we use, for each channel a, distinguished communication labels $a^{\triangleright}$ for putting messages on the channel and ${}^{\triangleright}a$ for consuming messages from the channel. If the set of channels

is empty a MAIOPN models an interface or a primitive component from which larger systems can be constructed by asynchronous composition.

In our example the behaviour of the CompressorAssembly is given by the asynchronous composition Controller $\otimes_{pn}$ TxtCompressor $\otimes_{pn}$ GifCompressor shown in Fig. 1d. It models a highly parallel system such that compressing of files can be executed concurrently and new files can be obtained at the same time. Each single compressing tool, however, is working sequentially. Its behaviour should be clear from the specifications. The GifCompressor in Fig. 1e has an optional behaviour modelled by a may-transition to indicate a compressing failure and then the controller will submit the file again.

After the assembly has been established we are interested in whether communication works properly in the sense that pending messages on communication channels will be consumed. We will distinguish between two variants of consumption requirements (see Sect. 5) expressing that for non-empty channels there must be a possibility for consumption or, more strongly, that consumption must always happen on each (observationally weakly fair) run. The fairness assumption is essential to support incremental design (Theorem 9); for instance we can first check that Controller $\otimes_{pn}$ TxtCompressor has the desired communication properties for the channels $\{\mathsf{txt}, \mathsf{zip}\}$, then we check that the full assembly has the desired communication properties for its channel subset $\{\mathsf{gif}, \mathsf{jpg}, \mathsf{fail}\}$ and from this we can automatically derive that the assembly has the properties for *all* its channels.

It remains to show that the CompressorAssembly is indeed a realisation of the CompressorInterface. For this purpose we consider the black-box behaviour of the assembly obtained by hiding the communication channels. This is done by applying our hiding operator to the CompressorAssembly denoted by CompressorAssembly$\backslash_{pn}\{\mathsf{gif}, \mathsf{jpg}, \mathsf{fail}, \mathsf{txt}, \mathsf{zip}\}$; see Fig. 2. Hiding moves all communication labels $a^{\triangleright}$ and ${}^{\triangleright}a$ for the hidden channels a to the invisible action τ. In this way producing and consuming messages from hidden channels become silent transitions. Now we have to establish a refinement relation between CompressorAssembly$\backslash_{pn}\{\mathsf{gif}, \mathsf{jpg}, \mathsf{fail}, \mathsf{txt}, \mathsf{zip}\}$ and the CompressorInterface by taking into account the modalities on the transitions such that must-transitions of the abstract specification must be available in the refinement and all transitions of the refinement must be allowed by corresponding may-transitions of the abstract specification. In our example the assembly has implemented the optional jpg compression of the interface by a must-transition. Obviously we must also deal with silent transitions which, in our example, occur in CompressorAssembly$\backslash_{pn}\{\mathsf{gif}, \mathsf{jpg}, \mathsf{fail}, \mathsf{txt}, \mathsf{zip}\}$. For this purpose we use a modal refinement relation, denoted by $\leq^{*}_{\mathrm{m}}$, which supports observational abstraction. In our case study this is expressed by the proof obligation (1) in Fig. 2.

Figure 2 illustrates that after the assembly is proven to be a correct realisation of the interface one can still further refine the assembly by component-wise refinement of its constituent parts. For instance, we can locally refine the GifCompressor by resolving the may-modality for producing failures. There are basically two possibilities: Either the failure option is removed or it is turned into

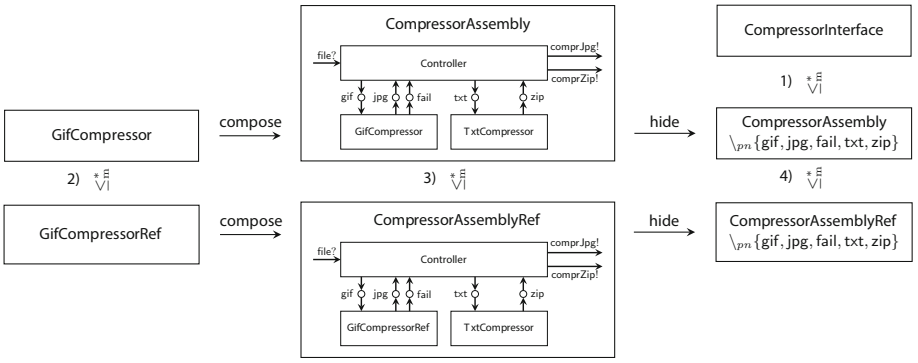


Fig. 2. System development methodology.

a must-transition. The component GifCompressorRef in Fig. 2 represents such a refinement of GifCompressor indicated by (2). We will show in Theorem 3.1 that refinement is compositional, i.e. we obtain automatically that the assembly CompressorAssemblyRef obtained by composition with the new gif compressor is a refinement of CompressorAssembly indicated by (3) in Fig. 2. As a crucial result we will also show in Theorem 10 that refinement preserves communication properties which then can be automatically derived for CompressorAssemblyRef. Finally, we must be sure that the refined assembly provides a realisation of the original interface specification CompressorInterface. This can be again automatically achieved since hiding preserves refinement (Theorem 3.2) which leads to (4) in Fig. 2 and since refinement is transitive.

In the next sections we will formally elaborate the notions discussed above. We hope that their intended meaning is already sufficiently explained such that we can keep the presentation short. An exception concerns the consideration of the message consuming properties in Sect. 5 which must still be carefully studied to ensure incremental design and preservation by refinement.

3 Modal Asynchronous I/O-Petri Nets

In this section we formalise the syntax of MAIOPNs and we define their transition system semantics. First we recall some basic definitions for modal Petri nets and modal transition systems.

3.1 Modal Petri Nets and Modal Transition Systems

In the following we consider labelled Petri nets such that transitions are equipped with labels of an alphabet Σ or with the symbol τ that models silent transitions. We write Σ_τ for $\Sigma \uplus \{\tau\}$. We assume the reader to be familiar with the basic notions of labelled Petri nets consisting of a finite set P of places, a finite set T of transitions, a set of arcs between places and transitions (transitions and

places resp.), formalised by incidence matrices W^- and W^+, an initial marking m^0 and a labelling function $\lambda : T \to \Sigma_\tau$. In [7] we have introduced *modal Petri nets* $\mathcal{N} = (P, T, T^\Box, \Sigma, W^-, W^+, \lambda, m^0)$ such that $T^\Box \subseteq T$ is a distinguished subset of must-transitions. We write $m \overset{t}{\dashrightarrow} m'$ if a transition $t \in T$ is firable from a marking m leading to a marking m'. If $t \in T^\Box$ we write $m \overset{t}{\longrightarrow} m'$. If $\lambda(t) = a$ we write $m \overset{a}{\dashrightarrow} m'$ and for $t \in T^\Box$ we write $m \overset{a}{\longrightarrow} m'$. The notation is extended as usual to firing sequences $m \overset{\sigma}{\dashrightarrow} m'$ with $\sigma \in T^*$ and to $m \overset{\sigma}{\longrightarrow} m'$ with $\sigma \in (T^\Box)^*$. A marking m is reachable if there exists a firing sequence $\sigma \in T^*$ such that $m^0 \overset{\sigma}{\dashrightarrow} m$.

Modal transition systems have been introduced in [15]. We will use them to provide semantics for modal Petri nets. A modal transition system is a tuple $\mathcal{S} = (\Sigma, Q, q^0, \dashrightarrow, \longrightarrow)$, such that Σ is a set of labels, Q is a set of states, $q^0 \in Q$ is the initial state, $\dashrightarrow \subseteq Q \times \Sigma_\tau \times Q$ is a may-transition relation, and $\longrightarrow \subseteq \dashrightarrow$ is a must-transition relation. We explicitly allow the state space Q to be infinite which is needed to give interpretations to modal Petri nets that model infinite state systems.

The semantics of a modal Petri net $\mathcal{N} = (P, T, T^\Box, \Sigma, W^-, W^+, \lambda, m^0)$ is given by the modal transition system $\mathsf{mts}(\mathcal{N}) = (\Sigma, Q, q^0, \dashrightarrow, \longrightarrow)$ representing the reachability graph of the net by taking into account modalities: $Q \subseteq \mathbb{N}^P$ is the set of reachable markings of $\mathcal{N}$, $\dashrightarrow = \{(m, a, m') \mid a \in \Sigma_\tau \text{ and } m \overset{a}{\dashrightarrow} m'\}$, $\longrightarrow = \{(m, a, m') \mid a \in \Sigma_\tau \text{ and } m \overset{a}{\longrightarrow} m'\}$, and $q^0 = m^0$.

In the sequel we will use the following notations for modal transition systems. We write may-transitions as $q \overset{a}{\dashrightarrow} q'$ for $(q, a, q') \in \dashrightarrow$ and similarly must-transitions as $q \overset{a}{\longrightarrow} q'$. The notation is extended in the usual way to finite sequences $\sigma \in \Sigma_\tau^*$ by the notations $q \overset{\sigma}{\dashrightarrow} q'$ and $q \overset{\sigma}{\longrightarrow} q'$. The set of reachable states of $\mathcal{S}$ is given by $\mathrm{Reach}(\mathcal{S}) = \{q \mid \exists \sigma \in \Sigma_\tau^* \,.\, q^0 \overset{\sigma}{\dashrightarrow} q\}$. For a sequence $\sigma \in \Sigma_\tau^*$ the observable projection $\mathrm{obs}(\sigma) \in \Sigma^*$ is obtained from σ by removing all occurrences of τ. Let $a \in \Sigma$ be a visible action and $q, q' \in Q$. We write $q \overset{a}{=\!=\!\Rightarrow} q'$ if $q \overset{\sigma}{\dashrightarrow} q'$ with $\mathrm{obs}(\sigma) = a$ and we call $q \overset{a}{=\!=\!\Rightarrow} q'$ a weak may-transition. Similarly we write $q \overset{a}{\Longrightarrow} q'$ if all transitions are must-transitions and call $q \overset{a}{\Longrightarrow} q'$ a weak must-transition. The notation is extended as expected to finite sequences $\sigma \in \Sigma^*$ of visible actions by the notations $q \overset{\sigma}{=\!=\!\Rightarrow} q'$ and $q \overset{\sigma}{\Longrightarrow} q'$. In particular, $q \overset{\epsilon}{=\!=\!\Rightarrow} q'$ means that there is a (possibly empty) sequence of silent may-transitions from q to q' and similarly $q \overset{\epsilon}{\Longrightarrow} q'$ expresses a finite sequence of silent must-transitions.

3.2 Modal Asynchronous I/O-Petri Nets, Composition and Hiding

In this paper we consider systems which may be open for communication with other systems and may be composed to larger systems. The open actions are modelled by input labels (for the reception of messages from the environment) and output labels (for sending messages to the environment) while communication inside an asynchronously composed system takes place by removing or putting messages to distinguished communication channels. Given a finite set C

of channels, an *I/O-alphabet over* C is the disjoint union $\Sigma = \mathsf{in} \uplus \mathsf{out} \uplus \mathsf{com}$ of pairwise disjoint sets of input labels, output labels and communication labels, such that $\mathsf{com} = \{{}^{\triangleright}a, a^{\triangleright} \mid a \in C\}$. For each channel $a \in C$, the label ${}^{\triangleright}a$ represents consumption of a message from a and $a^{\triangleright}$ represents putting a message on a. A *modal asynchronous I/O-Petri net* (MAIOPN) $\mathcal{N} = (C, P, T, T^{\Box}, \Sigma, W^{-}, W^{+}, \lambda, m^0)$ is a modal Petri net such that $C \subseteq P$ is a set of channel places which are initially empty, $\Sigma = \mathsf{in} \uplus \mathsf{out} \uplus \mathsf{com}$ is an I/O-alphabet over C, and for all $a \in C$ and $t \in T$, there exists an (unweighted) arc from a to t iff $\lambda(t) = {}^{\triangleright}a$ and there exists an (unweighted) arc from t to a iff $\lambda(t) = a^{\triangleright}$.

The *asynchronous composition* of MAIOPNs works as for asynchronous I/O-Petri nets considered in [10] by introducing new channel places and appropriate transitions for shared input/output labels. The non-shared input and output labels remain open in the composition. Moreover, we require that the must-transitions of the composition are the union of the must-transitions of the single components. The asynchronous composition of two MAIOPNs $\mathcal{N}$ and $\mathcal{M}$ is denoted by $\mathcal{N} \otimes_{pn} \mathcal{M}$. It is commutative and also associative under appropriate syntactic restrictions on the underlying alphabets. An example of a composition of three MAIOPNs is given in Fig. 1d.

We introduce a *hiding operator* on MAIOPNs which allows us to hide communication channels. In particular, it allows us to compute the black-box behaviour of an assembly when all channels are hidden. Let $\mathcal{N}$ be a MAIOPN with I/O-alphabet $\Sigma_{\mathcal{N}} = \mathsf{in}_{\mathcal{N}} \uplus \mathsf{out}_{\mathcal{N}} \uplus \mathsf{com}_{\mathcal{N}}$, and let $H \subseteq C_{\mathcal{N}}$ be a subset of its channels. The *channel hiding of* H in $\mathcal{N}$ is the MAIOPN $\mathcal{N} \setminus_{pn} H$ with channels $C = C_{\mathcal{N}} \setminus H$, with I/O-alphabet $\Sigma = \mathsf{in}_{\mathcal{N}} \uplus \mathsf{out}_{\mathcal{N}} \uplus \mathsf{com}$ such that $\mathsf{com} = \{{}^{\triangleright}a, a^{\triangleright} \mid a \in C\}$, and with the labelling function:

$$\lambda(t) = \begin{cases} \tau & \text{if } \exists a \in H \,.\, \lambda_{\mathcal{N}}(t) = {}^{\triangleright}a \text{ or } \lambda_{\mathcal{N}}(t) = a^{\triangleright}, \\ \lambda_{\mathcal{N}}(a) & \text{otherwise.} \end{cases}$$

3.3 Semantics: Modal Asynchronous I/O-Transition Systems

We extend the transition system semantics of modal Petri nets defined in Sect. 3.1 to MAIOPNs. For this purpose we introduce *modal asynchronous I/O-transition system* (MAIOTS) $\mathcal{S} = (C, \Sigma, Q, q^0, \dashrightarrow, \longrightarrow, \mathrm{val})$ which are modal transition systems such that C is a finite set of channels, $\Sigma = \mathsf{in} \uplus \mathsf{out} \uplus \mathsf{com}$ is an I/O-alphabet over C, and $\mathrm{val} : Q \longrightarrow \mathbb{N}^C$ is a channel valuation function which determines for each state $q \in Q$ how many messages are actually pending on each channel $a \in C$. Instead of $\mathrm{val}(q)(a)$ we write $\mathrm{val}(q, a)$. We require that initially all channels are empty, i.e. $\mathrm{val}(q^0, a) = 0$ for all $a \in C$, that for each $a \in C$ putting $a^{\triangleright}$ and consuming ${}^{\triangleright}a$ messages from a has the expected behaviour, and that the open input/output actions have no effect on a channel[1], i.e.

[1] $\mathrm{val}(q)[a{+}{+}]$ ($\mathrm{val}(q)[a{-}{-}]$ resp.) denotes the update of val which increments (decrements) the value of a and leaves the values of all other channels unchanged.

$$q \xrightarrow{a^{\triangleright}} q' \implies \mathrm{val}(q') = \mathrm{val}(q)[a{+}{+}],$$

$$q \xrightarrow{{}^{\triangleright}a} q' \implies \mathrm{val}(q,a) > 0 \text{ and } \mathrm{val}(q') = \mathrm{val}(q)[a--], \text{ and}$$

$$\text{for all } x \in (\mathsf{in} \cup \mathsf{out}), q \xrightarrow{x} q' \implies \mathrm{val}(q') = \mathrm{val}(q).$$

The semantics $\mathsf{maiots}(\mathcal{N})$ of a modal asynchronous I/O-Petri net $\mathcal{N}$ is given by the transition system semantics of modal Petri nets such that the reachable markings are the states. Additionally we define the associated channel valuation function $\mathrm{val} : Q \longrightarrow \mathbb{N}^C$ such that the valuation of a channel a in a current state m is given by the number of tokens on a under the marking m, i.e. $\mathrm{val}(m,a) = m(a)$. For instance, the semantics of the CompressorInterface in Sect. 2 and of the Controller leads to infinite state transition systems; the transition systems associated with the two compressor components have two reachable states and the transitions between them correspond directly to their Petri net representations in Fig. 1e, f.

The *asynchronous composition* $\mathcal{S} \otimes \mathcal{T}$ of two MAIOTSs $\mathcal{S}$ and $\mathcal{T}$ works as for asynchronous I/O-transition systems in [10] taking additionally care that must-transitions of $\mathcal{S}$ and $\mathcal{T}$ induce must-transitions of $\mathcal{S} \otimes \mathcal{T}$. The composition introduces new channels $C_{\mathcal{ST}} = \Sigma_{\mathcal{S}} \cap \Sigma_{\mathcal{T}}$ for shared input/output labels. Every transition with a shared output label a becomes a transition with the communication label $a^{\triangleright}$, and similarly transitions with input labels a become transitions with label ${}^{\triangleright}a$. The state space of the composition is (the reachable part of) the Cartesian product of the underlying state spaces of $\mathcal{S}$ and $\mathcal{T}$ together with the set $\mathbb{N}^{C_{\mathcal{ST}}}$ of valuations for the new channels such that transitions labelled by $a^{\triangleright}$ and ${}^{\triangleright}a$ have the expected effect on the new channels; for details see [10]. The asynchronous composition of two MAIOTSs is commutative and also associative under appropriate syntactic restrictions on the underlying alphabets.

We also introduce a *hiding operator* on MAIOTSs that hides communication channels and moves all corresponding communication labels to τ. Let $\mathcal{S} = (C_{\mathcal{S}}, \Sigma_{\mathcal{S}}, Q_{\mathcal{S}}, q^0_{\mathcal{S}}, \dashrightarrow_{\mathcal{S}}, \longrightarrow_{\mathcal{S}}, \mathrm{val}_{\mathcal{S}})$ be a MAIOTS with I/O-alphabet $\Sigma_{\mathcal{S}} = \mathsf{in}_{\mathcal{S}} \uplus \mathsf{out}_{\mathcal{S}} \uplus \mathsf{com}_{\mathcal{S}}$, and let $H \subseteq C_{\mathcal{S}}$ be a subset of its channels. The *channel hiding of* H in $\mathcal{S}$ is the MAIOTS $\mathcal{S} \setminus H = (C, \Sigma, Q_{\mathcal{S}}, q^0_{\mathcal{S}}, \dashrightarrow, \longrightarrow, \mathrm{val})$, such that $C = C_{\mathcal{S}} \setminus H$, $\Sigma = \mathsf{in}_{\mathcal{S}} \uplus \mathsf{out}_{\mathcal{S}} \uplus \mathsf{com}$ with $\mathsf{com} = \{{}^{\triangleright}a, a^{\triangleright} \mid a \in C\}$, $\mathrm{val}(q,a) = \mathrm{val}_{\mathcal{S}}(q,a)$ for all $q \in Q, a \in C$, and the may-transition relation is given by:

$$\dashrightarrow = \{(q,a,q') \mid a \in (\Sigma_\tau \setminus \{{}^{\triangleright}a, a^{\triangleright} \mid a \in H\}) \text{ and } q \overset{a}{\dashrightarrow}_{\mathcal{S}} q'\} \cup$$
$$\{(q,\tau,q') \mid \exists a \in H \text{ such that either } q \overset{a^{\triangleright}}{\dashrightarrow}_{\mathcal{S}} q' \text{ or } q \overset{{}^{\triangleright}a}{\dashrightarrow}_{\mathcal{S}} q'\},$$

The must-transition relation $\longrightarrow$ is defined analogously.

For the results developed in the next sections it is important that our transition system semantics is compositional and compatible with hiding as stated in the next theorem. The proof is given in Appendix A of [11].

Theorem 1. *Let $\mathcal{N}$ and $\mathcal{M}$ be two composable MAIOPNs and let $H \subseteq C_{\mathcal{N}}$ be a subset of the channels of $\mathcal{N}$. The following holds:*

1. $\mathsf{maiots}(\mathcal{N} \otimes_{pn} \mathcal{M}) = \mathsf{maiots}(\mathcal{N}) \otimes \mathsf{maiots}(\mathcal{M})$ *(up to isomorphic state spaces),*
2. $\mathsf{maiots}(\mathcal{N} \setminus_{pn} H) = \mathsf{maiots}(\mathcal{N}) \setminus H$.

4 Modal Refinement

The refinement relation between MAIOPNs will be defined by considering their semantics, i.e. MAIOTSs. For this purpose we adapt the weak modal refinement relation for modal transition systems introduced in [13] which is based on a simulation relation in both directions. It says that must-transitions of an abstract specification must be preserved by the refinement while may-transitions of a concrete specification must be allowed by the abstract one. In any case silent transitions labelled with τ can be inserted, similarly to weak bisimulation, but respecting modalities. Observational abstraction from silent transitions is indeed important in many examples; e.g. when relating the encapsulated behaviour of an assembly to a requirements specification as discussed in Sect. 2. If all transitions of the abstract specification are must-transitions, modal refinement coincides with weak bisimulation. Obviously, the modal refinement relation defined as follows is reflexive and transitive.

Definition 2 (Modal refinement). *Let* $\mathcal{S} = (C, \Sigma, Q_\mathcal{S}, q^0_\mathcal{S}, \dashrightarrow_\mathcal{S}, \longrightarrow_\mathcal{S}, \mathrm{val}_\mathcal{S})$ *and* $\mathcal{T} = (C, \Sigma, Q_\mathcal{T}, q^0_\mathcal{T}, \dashrightarrow_\mathcal{T}, \longrightarrow_\mathcal{T}, \mathrm{val}_\mathcal{T})$ *be two MAIOTSs with the same I/O-alphabet* Σ *over channels* C*. A relation* $R \subseteq Q_\mathcal{S} \times Q_\mathcal{T}$ *is a* modal refinement relation *between* $\mathcal{S}$ *and* $\mathcal{T}$ *if for all* $(q_\mathcal{S}, q_\mathcal{T}) \in R$ *and* $a \in \Sigma$*:*

1. $q_\mathcal{T} \xrightarrow{a}_\mathcal{T} q'_\mathcal{T} \implies \exists q'_\mathcal{S} \in Q_\mathcal{S} \,.\, q_\mathcal{S} \overset{a}{\Longrightarrow}_\mathcal{S} q'_\mathcal{S} \wedge (q'_\mathcal{S}, q'_\mathcal{T}) \in R,$
2. $q_\mathcal{T} \xrightarrow{\tau}_\mathcal{T} q'_\mathcal{T} \implies \exists q'_\mathcal{S} \in Q_\mathcal{S} \,.\, q_\mathcal{S} \overset{\epsilon}{\Longrightarrow}_\mathcal{S} q'_\mathcal{S} \wedge (q'_\mathcal{S}, q'_\mathcal{T}) \in R,$
3. $q_\mathcal{S} \overset{a}{\dashrightarrow}_\mathcal{S} q'_\mathcal{S} \implies \exists q'_\mathcal{T} \in Q_\mathcal{T} \,.\, q_\mathcal{T} \overset{a}{=\!=\!\Rightarrow}_\mathcal{T} q'_\mathcal{T} \wedge (q'_\mathcal{S}, q'_\mathcal{T}) \in R,$
4. $q_\mathcal{S} \overset{\tau}{\dashrightarrow}_\mathcal{S} q'_\mathcal{S} \implies \exists q'_\mathcal{T} \in Q_\mathcal{T} \,.\, q_\mathcal{T} \overset{\epsilon}{=\!=\!\Rightarrow}_\mathcal{T} q'_\mathcal{T} \wedge (q'_\mathcal{S}, q'_\mathcal{T}) \in R.$

We say that $\mathcal{S}$ *is a* modal refinement *of* $\mathcal{T}$*, written* $\mathcal{S} \leq^*_\mathrm{m} \mathcal{T}$*, if there exists a modal refinement relation* R *between* $\mathcal{S}$ *and* $\mathcal{T}$ *such that* $(q^0_\mathcal{S}, q^0_\mathcal{T}) \in R$*. We write* $q_\mathcal{S} \leq^*_\mathrm{m} q_\mathcal{T}$ *when* $(q_\mathcal{S}, q_\mathcal{T}) \in R$ *for a modal refinement relation* R*.* ♢

The next theorem shows that modal refinement is preserved by asynchronous composition and by channel hiding. The compositionality result stems in principle from [13] and is proved in Appendix B of [11] in the context of MAIOTS. The second result is also proved in Appendix B of [11] similarly to a result in [12] for hiding in synchronous systems.

Theorem 3. *Let* $\mathcal{S}, \mathcal{T}, \mathcal{E}$ *and* $\mathcal{F}$ *be MAIOTSs such that* C *is the set of channels of* $\mathcal{S}$ *and of* $\mathcal{T}$ *and let* $H \subseteq C$*.*

1. *If* $\mathcal{S} \leq^*_\mathrm{m} \mathcal{T}$*,* $\mathcal{E} \leq^*_\mathrm{m} \mathcal{F}$ *and* $\mathcal{S}$ *and* $\mathcal{E}$ *are composable, then* $\mathcal{T}$ *and* $\mathcal{F}$ *are composable and* $\mathcal{S} \otimes \mathcal{E} \leq^*_\mathrm{m} \mathcal{T} \otimes \mathcal{F}$*.*
2. *If* $\mathcal{S} \leq^*_\mathrm{m} \mathcal{T}$ *then* $\mathcal{S} \setminus H \leq^*_\mathrm{m} \mathcal{T} \setminus H$*.*

The refinement definition and results are propagated to modal asynchronous I/O-Petri nets in the obvious way: A MAIOPN $\mathcal{M}$ is a *modal refinement of* a MAIOPN $\mathcal{N}$, also denoted by $\mathcal{M} \leq^*_\mathrm{m} \mathcal{N}$, if $\mathsf{maiots}(\mathcal{M}) \leq^*_\mathrm{m} \mathsf{maiots}(\mathcal{N})$. The counterparts of Theorem 3.1 and 3.2 for MAIOPNs are consequences of the

semantic compatibility results in Theorem 1.1. Examples for modal refinements of MAIOPNs and applications of the theorem are pointed out in Sect. 2.

The decidability status of the modal refinement problem for MAIOPNs depends on the kind of Petri nets one considers. Observing that there is a simple reduction from the bisimilarity problem to the modal refinement problem and that the former problem is undecidable for Petri nets [14], one gets that the latter problem is also undecidable; for an evolved discussion see [2]. However when one restricts Petri nets to be modally weakly deterministic, the modal refinement problem becomes decidable. The modal weak determinism of Petri nets is a behavioural property which can also be decided (see [7] for both proofs). In addition, determinism is a desirable feature for a specification (when possible). For instance modal language specification is an alternative to modal transition systems that presents such a behaviour [17].

5 Message Consuming Systems

In this section we consider generic properties concerning the asynchronous communication via channels inspired by the various channel properties studied in [10].

We focus on the *message consuming* and the *necessarily message consuming* properties and generalise them to take into account modalities and observational abstraction w.r.t. silent transitions. Our goal is that the properties scale up to larger contexts (to support incremental design) and that they are preserved by modal refinement. Moreover we consider their decidability. For the definitions we rely on the semantics of MAIOPNs, i.e. on MAIOTSs.

Let us first discuss the message consuming property (a) of Definition 4 for a MAIOTS $\mathcal{S}$ and a subset B of its channels. It requires, for each channel $a \in B$, that if in an arbitrary reachable state q of $\mathcal{S}$ there is a message on a, then $\mathcal{S}$ must be able to consume it, possibly after some delay caused by the execution of autonomous must-transitions. All transitions that do not depend on the environment, i.e. are not related to input labels, are considered to be autonomous. Our approach follows a "pessimistic" assumption taking into account arbitrary environments that can let the system go where it wants and can also stop to provide inputs at any moment. It is important that we require must-transitions since the consuming property should be preserved by modal refinement. It is inspired by the notion of "output compatibility" studied for synchronously composed transition systems in [12].

A central role when components run in parallel is played by fairness assumptions; see, e.g., [4]. First we must define what we mean by a run and then we will explain our fairness notion. A run is a finite or infinite sequence of state transitions which satisfies a maximality condition. In principle a run can only stop when a deadlock is reached. However we must be careful since (1) we are dealing with open systems whose executions depend on the input from the environment, (2) we must take into account that transitions with a may-modality can be skipped in a refinement, and (3) we must be aware that also silent must-transitions without successive mandatory visible actions can be omitted in a

refinement; cf. Definition 2, rule 2. In particular divergence in an abstract state could be implemented by a deadlock. If one of the above conditions holds in a certain state it is called a *potential deadlock state*.

Formally, let $\mathcal{S} = (C, \Sigma, Q, q^0, \dashrightarrow, \longrightarrow, \mathrm{val})$ be a MAIOTS with $\Sigma = \mathsf{in} \uplus \mathsf{out} \uplus \mathsf{com}$. A state $q \in Q$ is a potential deadlock state if for all $a \in (\Sigma \setminus \mathsf{in})$, there is no state $q' \in Q$ such that $q \stackrel{a}{\Longrightarrow} q'$. A *run* of $\mathcal{S}$ starting in a state $q_1 \in Q$ is a finite or infinite sequence $\rho = q_1 \stackrel{a_1}{\dashrightarrow} q_2 \stackrel{a_2}{\dashrightarrow} q_3 \stackrel{a_3}{\dashrightarrow} \cdots$ with $a_i \in \Sigma_\tau$ and $q_i \in Q$ such that, if the sequence is finite, its last state is a potential deadlock state. We assume that system runs are executed in a runtime infrastructure which follows a fair scheduling policy. In our context this means that any visible autonomous action a, that is always enabled by a weak must-transition from a certain state on, will infinitely often be executed. Formally, a run $\rho = q_1 \stackrel{a_1}{\dashrightarrow} q_2 \stackrel{a_2}{\dashrightarrow} \cdots$ is called *observationally weakly fair* if it is finite or if it is infinite and then for all $a \in (\Sigma \setminus \mathsf{in})$ the following holds:

$$(\exists k \geq 1 \,.\, \forall i \geq k \,.\, \exists q' \,.\, q_i \stackrel{a}{\Longrightarrow} q') \implies (\forall k \geq 1 \,.\, \exists i \geq k \,.\, a_i = a).$$

We denote the set of all observationally weakly fair runs of $\mathcal{S}$ starting from q_1 by $\mathrm{owfrun}_{\mathcal{S}}(q_1)$. For instance, for the MAIOPN $\mathcal{M}$ in Fig. 3b on page xx an infinite run which always executes $a^{\triangleright}, {}^{\triangleright}a, \ldots$ is observationally weakly fair. A (diverging) run of $\mathcal{M}$ which always executes τ from a certain state on is not observationally weakly fair since ${}^{\triangleright}a$ is then always enabled by a weak must-transition but never taken.

Note that for our results it is sufficient to use a weak fairness property instead of strong fairness. We are now ready to define also the necessarily consuming property 4 which requires that whenever a message is pending on a communication channel then the message must eventually be consumed on all observationally weakly fair runs.

Definition 4 (Message consuming). *Let $\mathcal{S} = (C, \Sigma, Q, q^0, \dashrightarrow, \longrightarrow, \mathrm{val})$ be a MAIOTS with I/O-alphabet $\Sigma = \mathsf{in} \uplus \mathsf{out} \uplus \mathsf{com}$ and let $B \subseteq C$ be a subset of its channels.*

(a) $\mathcal{S}$ is message consuming *w.r.t. B if for all $a \in B$ and all $q \in \mathrm{Reach}(\mathcal{S})$,*

$$\mathrm{val}(q, a) > 0 \implies \exists q', q'' \in Q \,.\, \exists \sigma \in (\Sigma \setminus \mathsf{in})^* \,.\, q \stackrel{\sigma}{\Longrightarrow} q' \stackrel{{}^{\triangleright}a}{\longrightarrow} q''.$$

(b) $\mathcal{S}$ is necessarily message consuming *w.r.t. B if for all $a \in B, q \in \mathrm{Reach}(\mathcal{S})$,*

$$\mathrm{val}(q, a) > 0 \implies \forall \rho \in \mathrm{owfrun}_{\mathcal{S}}(q) \,.\, {}^{\triangleright}a \in \rho \,.^{2}$$

$\mathcal{S}$ is (necessarily) message consuming *if $\mathcal{S}$ is (necessarily) message consuming w.r.t. C.* ♢

In the special case, in which all may-transitions are must-transitions and no silent transitions occur observationally weakly fair runs coincide with weakly fair runs and the two consuming properties coincide with the corresponding properties in [10].

[2] i.e. there is a transition in ρ labelled by ${}^{\triangleright}a$.

Proposition 5. *Let $\mathcal{S}$ be a MAIOTS. If $\mathcal{S}$ is necessarily message consuming w.r.t B then $\mathcal{S}$ is message consuming w.r.t B.*

The proof can found in Appendix C of [11]. It is an adoptation of the one in [9].

The definitions and the proposition are propagated to modal asynchronous I/O-Petri nets in the obvious way. For instance, a MAIOPN $\mathcal{N}$ is *(necessarily) message consuming* if $\mathsf{maiots}(\mathcal{N})$ is (necessarily) message consuming. All MAIOPNs considered in Sect. 2 are necessarily message consuming.

As stated in the introduction, Petri nets are a useful model since (1) they model infinite state systems and (2) several relevant properties of transition systems are decidable. The following proposition whose proof is an adaption of the one in [10] establishes that one can decide both consuming properties. For sake of completeness, its proof can be found in Appendix C of [11].

Proposition 6. *Let $\mathcal{N}$ be a MAIOPN and let $B \subseteq C$ be a subset of its channels. The satisfaction by $\mathcal{N}$ of the message consuming and the necessarily message consuming properties w.r.t. B are decidable.*

Both message consuming properties are compositional; they are preserved when systems are put into larger contexts. The proof of the compositionality of the message consuming property 4 relies on the fact that autonomous executions of constituent parts (not involving inputs) can be lifted to executions of compositions. To prove compositionality of the necessarily consuming property 4 one shows that projections of observationally weakly fair runs to constituent parts of a composition are again observationally weakly fair runs. Both facts and the following consequences are proved in Appendix C of [11]. The proof has the same shape as the proof of Proposition 15 in [10] which is given in [9].

Proposition 7. *Let $\mathcal{S}$ and $\mathcal{T}$ be two composable MAIOTSs such that $C_{\mathcal{S}}$ is the set of channels of $\mathcal{S}$. Let $B \subseteq C_{\mathcal{S}}$. If $\mathcal{S}$ is (necessarily) consuming w.r.t. B, then $\mathcal{S} \otimes \mathcal{T}$ is (necessarily) consuming w.r.t. B.*

Proposition 7 leads directly to the desired modular verification result which allows us to check consuming properties in an incremental manner: To show that a composed system $\mathcal{S} \otimes \mathcal{T}$ is (necessarily) message consuming it is sufficient to know that both constituent parts $\mathcal{S}$ and $\mathcal{T}$ have this property and to check that $\mathcal{S} \otimes \mathcal{T}$ is (necessarily) message consuming w.r.t. the new channels established for the communication between $\mathcal{S}$ and $\mathcal{T}$, i.e. that $\mathcal{S}$ and $\mathcal{T}$ are compatible.

Definition 8 (Compatibility). *Two composable MAIOTSs $\mathcal{S}$ and $\mathcal{T}$ with shared labels $\Sigma_{\mathcal{S}} \cap \Sigma_{\mathcal{T}}$ are* (necessarily) message consuming compatible *if $\mathcal{S} \otimes \mathcal{T}$ is (necessarily) message consuming w.r.t. $\Sigma_{\mathcal{N}} \cap \Sigma_{\mathcal{M}}$.* ♢

Theorem 9 (Incremental Design). *Let $\mathcal{S}$ and $\mathcal{T}$ be (necessarily) message consuming compatible. If both $\mathcal{S}$ and $\mathcal{T}$ are (necessarily) message consuming, then $\mathcal{S} \otimes \mathcal{T}$ is (necessarily) message consuming.*

All results hold analogously for asynchronous I/O-Petri nets due to the compositional semantics of MAIOPNs; see Theorem 1.1 An application of incremental design has been discussed in Sect. 2.

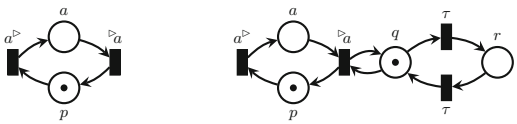
An important issue concerns the preservation of the message consuming properties by refinement. We can show that this holds for modal refinement which is not considered in [10]. The preservation of the message consuming property relies on the fact that for any "concrete" reachable state there is a related "abstract" state with the same number of messages on each channel. To prove the preservation of the necessarily consuming property the essential point is to show that for any observationally weakly fair run of a concrete MAIOTS there is a corresponding observationally weakly fair run of the abstract MAIOTS with the same visible actions. Both facts and the following consequences are proved in Appendix C of [11].

Theorem 10. *Let $\mathcal{S}$, $\mathcal{T}$ be two MAIOTSs with channels C and let $\mathcal{S} \leq^*_{\mathrm{m}} \mathcal{T}$. Let $B \subseteq C$. If $\mathcal{T}$ is (necessarily) message consuming w.r.t. B, then $\mathcal{S}$ is (necessarily) message consuming w.r.t. B. By definition, the theorem propagates to MAIOPNs.*

Example 11. The nets in Fig. 3 show an abstract MAIOPN $\mathcal{N}$ and a concrete MAIOPN $\mathcal{M}$ with silent τ-transitions. Both nets have a single channel place a. Obviously, $\mathcal{M} \leq^*_{\mathrm{m}} \mathcal{N}$ is a modal refinement. It is also clear that $\mathcal{N}$ is necessarily message consuming. By Theorem 10, $\mathcal{M}$ is necessarily message consuming as well. Indeed, as pointed out above, a diverging run of $\mathcal{M}$ which always executes τ from a certain state on is not observationally weakly fair and therefore needs not to be considered. This shows also why weakly fair runs are not appropriate here since a diverging run of $\mathcal{M}$ is weakly fair (it always visits a state in which $\triangleright a$ is not immediately enabled) but does not consume.

As a consequence of Theorems 3.1 and 10 our theory supports the principle of independent implementability in the sense of [1]. This fact is applied in Sect. 2 to obtain the global refinement (3) in Fig. 2 from the local refinement (2) preserving the necessarily consuming property.

Corollary 12 (Independent Implementability). *Let $\mathcal{S}, \mathcal{T}, \mathcal{E}$ and $\mathcal{F}$ be MAIOTSs. If $\mathcal{T}$ and $\mathcal{F}$ are (necessarily) message consuming compatible and $\mathcal{S} \leq^*_{\mathrm{m}} \mathcal{T}$ and $\mathcal{E} \leq^*_{\mathrm{m}} \mathcal{F}$, then $\mathcal{S}$ and $\mathcal{E}$ are (necessarily) message consuming compatible and $\mathcal{S} \otimes \mathcal{E} \leq^*_{\mathrm{m}} \mathcal{T} \otimes \mathcal{F}$. This holds analogously for MAIOPNs.*



(a) Abstract MAIOPN $\mathcal{N}$ (b) Concrete MAIOPN $\mathcal{M}$

Fig. 3. Necessarily consuming nets and modal refinement

6 Conclusion and Future Work

We have developed a fully integrated approach for the design of asynchronously composed component systems based on the formalism of MAIOPNs. Our approach ensures that the communication properties are preserved by asynchronous composition and by modal refinement, the basic ingredients of the design process. Several continuations of this work are possible. First, it would be interesting to see how our approach works in larger case studies and concrete applications. The "Assume/Guarantee" approach is a standard way to substitute a component by a behavioural interface in order to make easier the compositional verification. We plan to investigate how this approach can be integrated in our framework. Also it would be interesting to consider further operators on specifications like quotients. For the latter we would be faced with the problem to find mild conditions for the existence of quotients in the context of modal refinement which supports observational abstraction. Finally, broadcasting is an appropriate communication mechanism in the asynchronous environment. So it would be interesting to investigate how our approach can be adapted to this communication operator.

References

1. de Alfaro, L., Henzinger, T.A.: Interface-based design. Engineering Theories of Software-intensive Systems, NATO Science Series: Mathematics, Physics, and Chemistry, vol. 195. Springer, pp. 83–104 (2005)
2. Beneš, N., Křetínský, J.: Modal process rewrite systems. In: Proceedings of the International Conference on Theoretical Aspects of Computing (ICTAC 2012), vol. LNCS 7521, pp 120–135, Springer (2012)
3. Bernardo, M., Ciancarini, P., Donatiello, L.: Architecting families of software systems with process algebras. ACM Trans. Softw. Eng. Meth. **11**(4), 386–426 (2002)
4. Bérard, B., Bidoit, M., Finkel, A., Laroussinie, F., Petit, A., Petrucci, L., Schnoebelen, P.: Systems and Software Verification: Model-Cheking Techniques and Tools. Springer, Heidelberg (2001)
5. Best, E., Devillers, R., Koutny, M.: Petri Net Algebra. Springer Monographs in Theoretical Computer Science (2001)
6. Brauer, W., Gold, R., Vogler, W.: A survey of behaviour and equivalence preserving refinements of Petri nets. Applications and Theory of Petri Nets, pp. 1–46 (1989)
7. Elhog-Benzina, D., Haddad, S., Hennicker, R.: Refinement and asynchronous composition of modal Petri Nets. In: Jensen, K., Donatelli, S., Kleijn, J. (eds.) Transactions on Petri Nets and Other Models of Concurrency V. LNCS, vol. 6900, pp. 96–120. Springer, Heidelberg (2012)
8. Ganty, P., Raskin, J.-F., Van Begin, L.: From many places to few: automatic abstraction refinement for Petri Nets, ATPN 2007. LNCS **4546**, 124–143 (2007)
9. Haddad, S., Hennicker, R., Møller, M.H.: Channel properties of asynchronously composed Petri Nets. In: Colom, J.-M., Desel, J. (eds.) Petri Nets 2013. LNCS, vol. 7927, pp. 369–388. Springer, Heidelberg (2013)
10. Haddad, S., Hennicker, R., Møller, M.H.: Channel properties of asynchronously composed Petri Nets. Research Report LSV-13-05, Laboratoire Spécification et Vérification, ENS Cachan, France (2013)

11. Haddad, S., Hennicker, R., Møller, M.H.: Specification of asynchronous component systems with modal I/O-Petri Nets. Research Report LSV-13-16, Laboratoire Spécification et Vérification. ENS Cachan, France (2013)
12. 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)
13. Hüttel, H., Larsen, K.G.: The use of static constructs in a modal process logic. In: Logic at Botik 1989, pp. 163–180 (1989)
14. Jancar, P.: Undecidability of bisimilarity for Petri Nets and related problems. Theor. Comput. Sci. **148**, 281–301 (1995)
15. Larsen, K.G., Thomsen, B.; A modal process logic. In: 3rd Annual Symposium on Logic in Computer Science (LICS), IEEE Computer Society, pp. 203–210 (1988)
16. Lohmann, N., Massuthe, P., Wolf, K.: Operating guidelines for finite-state services. In: Kleijn, J., Yakovlev, A. (eds.) ICATPN 2007. LNCS, vol. 4546, pp. 321–341. Springer, Heidelberg (2007)
17. Raclet, J.-B.: Residual for component specifications. In: Proceedings of the 4th International Workshop on Formal Aspects of Component Software (FACS07), Sophia-Antipolis, France (2007)
18. Reisig, W.: Simple composition of nets. In: Franceschinis, G., Wolf, K. (eds.) Petri Nets 2009. LNCS, vol. 5606, pp. 23–42. Springer, Heidelberg (2009)
19. Schäfer, M., Vogler, W.: Component refinement and CSC-solving for STG decomposition. Theor. Comput. Sci. **388**(1–3), 243–266 (2007)
20. Souissi, Y.: On liveness preservation by composition of nets via a set of places. In: 11th International Conference on Applications and Theory of Petri Nets, LNCS 524, pp. 277–295 (1990)
21. Souissi, Y., Memmi, G.: Composition of nets via a communication medium. In: 10th International Conference on Applications and Theory of Petri Nets, LNCS 483, pp. 457–470 (1989)
22. Stahl, C., Wolf, K.: Deciding service composition and substitutability using extended operating guidelines. Data Knowl. Eng. **68**(9), 819–833 (2009)