

Modelling mobility with Petri hypernets^{*}

Marek A. Bednarczyk¹, Luca Bernardinello²
Wiesław Pawłowski¹ and Lucia Pomello²

¹ Institute of Computer Science, P.A.S., Gdańsk, Poland

² DISCO, Università degli Studi di Milano Bicocca, Milano, Italy

Abstract. *Petri hypernets*, a novel framework for modeling mobile agents based on **nets-within-nets** paradigm is presented. Hypernets employ a local and finitary character of interactions between agents, and provide means for a modular and hierarchical description. They are capable of modelling mobile agents that can dynamically change their hierarchy, and can communicate with each other and with the outside world by exchanging messages, i.e., other mobile agents.

1 Introduction

The idea of *mobility* has already attracted a lot of interest in the computing science community. Many existing formalisms devised to cope with specific application areas have been enriched by ‘mobility-related’ features. Here, examples range from extensions of Milner’s CCS, like *mobile processes* in π -calculus, cf. [10], or ‘programming-level’ notations such as *UNITY* or specification formalisms like *UML* (see e.g., [1]). Efforts to capture the essence of mobility also resulted a new, dedicated frameworks like the calculus of *mobile ambients* [6], *Join calculus* [7] or *mobile Petri nets* [2].

Here, yet another model called *Petri hypernets* is introduced. Petri nets, cf. [11], are well-known as a general and intuitive framework in which concurrent, asynchronous and distributed systems can be modeled. Our plan is to retain these strengths in an extension based on the principle that *mobile agents* should be modeled as Petri nets, and that other mobile agents should be able to manipulate them.

In contrast, e.g., *mobile nets* aim to capture the essence of mobility by allowing *names* of places of a Petri net to be sent as tokens. This requires the mechanisms of fresh variable creation and variable binding to be used in the formalism. We consider this as a departure from one of the fundamental concepts underlying Petri nets philosophy, namely that the interaction between places and transitions should be *local* and *finitary*. This sets our model apart from the formalisms like π -calculus, Join calculus, mobile nets, etc.

We intend to model mobile agents as nets and allow that nets are manipulated by other nets. Within a Petri net framework this can be realized

^{*} This research has been partially supported by the KBN grant No. 7 T11C 002 21, and by the CATNET and CATALYSIS projects within CNR/PAN and CNRS/PAN cooperation programmes, respectively.

by assuming that some nets are ‘tokens’ for other nets. This **nets-within-nets** paradigm was proposed by Valk and is being developed by his students and followers cf. [12–14, 8, 9]. Petri hypernets can be seen as yet another forking of the **nets-within-nets** paradigm. Each hypernet is a collection of mobile agents, called *open nets*, together with an assignment of mobile agents as tokens to places of other mobile agents, called *hypermarking*.

Petri hypernets support a *modular* and *hierarchical* description of reality.

There is a natural hierarchy of agents that corresponds to the assumption that any mobile agent should be higher in the hierarchy than any of the tokens it manipulates.

Modularity is imposed on mobile objects by assumption that each open net is a synchronous composition of *modules*, each one responsible for manipulation of mobile objects traveling along a fixed *channel*.

Co-operation between modules, each associated to a *different* channel within an open net, is enforced by *hand-shake synchronization* of their transitions. Namely, if the same transition t occurs in several different modules of an open net N , then in order to fire t in N all these modules must be ready to participate. It is well-known that every elementary net can be obtained as such a synchronization of state machines [4]. A *state machine* is a Petri net such that every transition has exactly one precondition and exactly one postcondition, and with only one token in exactly one place initially marked. Thus, all reachable markings are one-token markings. As a result the state machines are purely *sequential*, i.e., two transitions can never fire concurrently in a reachable marking. Something similar holds for 1-safe Petri nets. Namely, given such a net N one can construct a net N' with the same behavior as the synchronization of state machines, see [3]. These observations encourage us to restrict attention only to modules which have, essentially, the structure of sequential machines. Only the assumption that there is at most one token in exactly one place is dropped.

The features of hypernets discussed above could be found in the existing ramifications of the **nets-within-nets** paradigm. There are, however, several important novelties as well.

First, we assume that a mobile agent does not make any assumptions about the structure of other mobile agents. Following Valk’s original idea, also in our model an open net is allowed to synchronize the firing of its transition t with the firing of the transitions with the same name present in its tokens, if needed. Namely, each module in an open net has facilities to communicate locally, with a module of *its* kind in an adjacent open net. The nets *adjacent* to the given open net are those immediately *below* and *above* it in the hierarchy. However, unlike in Valk’s proposal, in hypernets such an inter-level synchronization is achieved solely by means of exchanging *messages*. This readiness to communicate with its parent net and its token nets, explains the terminology ‘open nets’ used to refer to mobile agents in our formalization.

Second, messages are also just mobile agents. This assumption implies that in Petri hypernets mobile agents can migrate between the levels of the hierarchy. Thus, the hierarchy itself may change.

Third, Valk gives two different semantics for his object nets. Consider, for instance, a transition with one input and two output places in the top level net. Then one has the problem explaining what is the outcome of firing this transition when a token net enters the input place. Valk considers two versions of the semantics, both based on the assumption that the transition does not, in fact, move the real token net, but *references* to it. Our decision to use sequential machines as moduls solves this problem. In particular, firing a transition preserves the identity of agents. We can think that a transition manipulates token nets as *values*, that it moves them, not their references, from the input place to the output place. The choice of sequential machines also entails that no agent is created, and no agent is destroyed. Thus, the hypernet has a finite state space, even if the agents it comprises are truly mobile, and can change their cooperation potential dynamically.

Finally, it is worth stressing that in case of hypernets the decompositions play an important structuring rôle. We see a hypernet as a hierarchical structure of open nets, each of them decomposed into a collection of sequential modules. From this perspective, firing of a transition t of the hypernet should be seen as a complex *transaction*, which involves firing transitions t which occur in the modules of all the open nets involved.

The reason for writing the paper is to present the simple ideas, together with their slightly less simple formalization. The main result states that the formalization is well-founded. Namely, firing a transaction preserves the forest-like hierarchy of open nets within a hypernet.

2 Petri Hypernets — a gentle introduction

2.1 Example: air travels

To introduce the model and illustrate the features of Petri hypernets let us recall a simple air travel case study considered e.g., in [1]. The case study requires the modeler to cope with mobile agents of at least 3 different kinds: *airports*, *planes* and *travelers*. The task is to describe the most basic aspects of the air travel involving possible collections of mobile agents of these kinds.

2.2 Kinds and Modules in the Airport

An agent of each kind exhibits dynamic behavior which may involve manipulation of objects of other kinds. For instance, while at an airport, travelers and planes are under the rules set forth by the airport.

One of the assumptions underlying the definition of hypernets is that agents cannot directly manipulate other agents of their kind. In fact, the separation of agents into *kinds* is one of the basic features of the model. Most of the

time kinds will be called *channels* to stress the communication aspect played by them within hypernets.

An airport perspective of the activities involved in handling of airplanes is presented on Fig. 1. This is a purely sequential view. The planes *land* at a

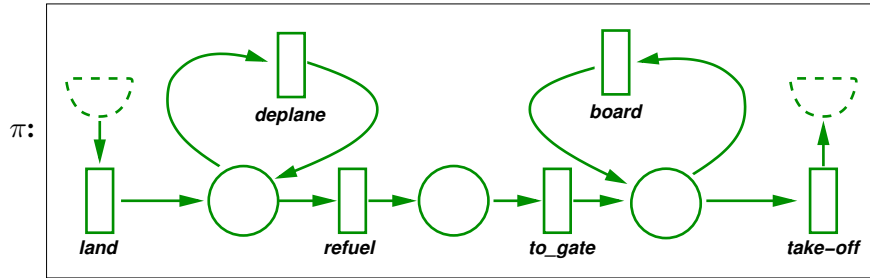


Fig. 1. Airport — plane manipulation module

gate. While at the gate they can *deplane* the travelers, one after another. Action *refuel* then takes planes to the refueling place. Then, *to_gate* moves planes to another gate where the plane may accept new travelers, and finally *take-off*.

The graphical description of the plane handling activities in Fig. 1 is essentially standard. The exception concerns the precondition of *land* and postcondition of *take-off* transitions. Here, the open world assumption comes into play. The airport is expecting that the planes are landing in co-operation with some *higher-level* traffic control authority. Similarly, the airport authority should be able to safely assume that upon *take-off* somebody above the hierarchy ladder will take care of the plane with its passengers.

The dashed half circles which provide the input for *land* and the output for *take-off* on Fig. 1 are intended to capture the above intuition. Drawing half of a place is meant to indicate that co-operation with higher level is required to successfully conclude the operation, here *land* and *take-off*. The dashes highlight the *virtual* character of the half-places. Namely, they are a means of synchronization of transitions between adjacent levels rather than real places that can store tokens. This evident link with the concept of *zero-places* proposed by Bruni and Montanari, see [5], remains to be investigated.

Fig. 1 describes the π -module of an airport agent only. We use the prefix π to indicate that the tokens manipulated within the module are ‘planes’. A τ -module for manipulating travelers at the airport is defined in Fig. 2. Recall that the π -module presented on Fig. 1 describes co-operation with the higher level only. Thus, the hypothetical higher level air traffic control agent should contain a π -module which can deliver the planes to airports for landing, and handle the planes in the air upon their take-off. This mechanism for co-operation with lower level agents is present in the τ -module on Fig. 2.

the π -module, which is a local place. Such well-formed synchronized modules are called *open nets*. The terminology intends to highlight the communication capabilities of the agents.

In our simplified view of the airport travelers appear at the airport either as a result of a plane landing, or they are already there in the initial marking. However, one could easily refine the model. For instance, adding a new module to the airport could amount to adding a new means of transportation of the passengers. Clearly, τ -module should be redefined to take care of the new ways of handling travelers. In principle, though, π -module could be left unchanged, unless some ways of direct interaction between the two means of transportation need to be introduced. This seems to indicate that the modular structure of mobile agents may increase re-usability of components.

2.4 Plane

A very simple model of a plane is presented on Fig. 4. It consists of just one

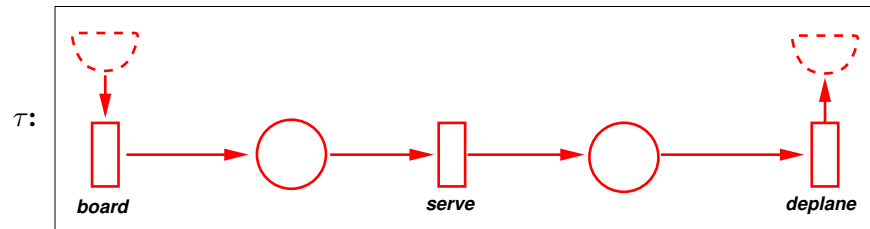


Fig. 4. An airplane.

module, a τ -module to handle travelers. The passengers may *board* the plane as a result of co-operation with a higher level agent and, after being *served* they may *deplane*, again in co-operation with the higher level mobile agent.

2.5 Petri Hypernet

An open net is, essentially, a synchronous product of sequential nets, whose transitions are capable also of sending and receiving tokens to and from other open nets. In fact, the tokens themselves are also open nets.

Accordingly, a *Petri hypernet* is a set of open nets \mathcal{N} plus a *hypermarking*:

$$m : \mathcal{N} \rightarrow \bigcup_{N \in \mathcal{N}} P_N$$

which describes the distribution of the elements of \mathcal{N} as *token nets* in places of other open nets in \mathcal{N} . The assumption that m is a partial function captures the idea that each open net has at most one *level up* net. To ensure that the

net N' such that $m(N) \in P_{N'}$ is indeed 'higher' we require that the hierarchy, i.e., the transitive closure of this 'one level up' relation, is irreflexive. If it is, we call the hypermarking *well-founded*.

Fig. 5 presents an open Petri hypernet which consists of one top level net, the airport, one plane and two (unspecified) traveler nets.

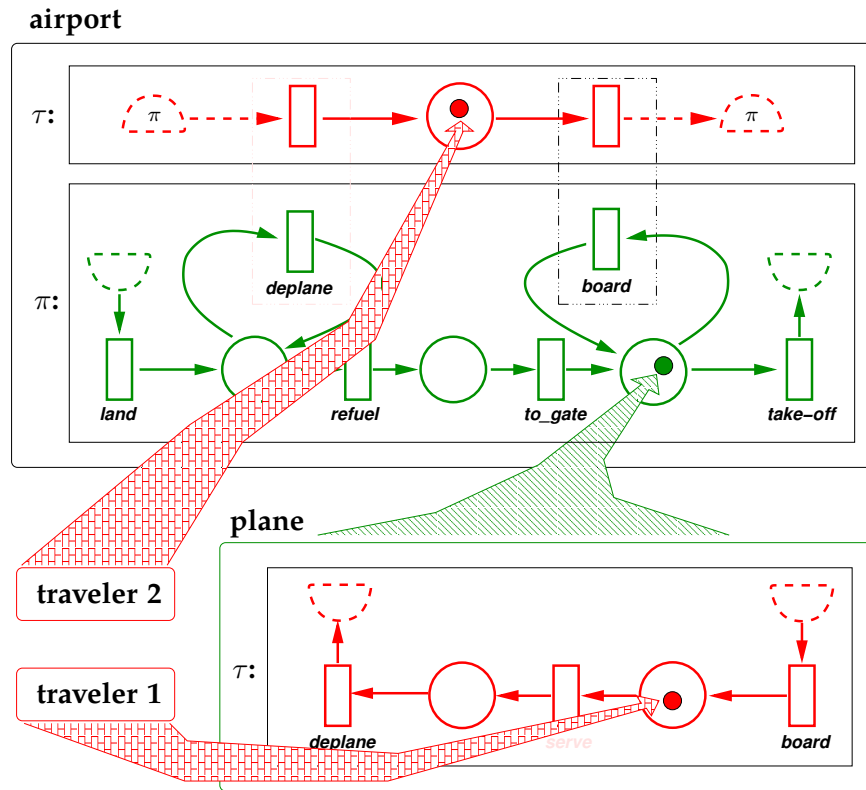


Fig. 5. An open Petri hypernet.

The hypernet is called *open*, since the top level net has a transition ready to receive and another ready to send tokens to a higher level.

Clearly, the hierarchy described by the hypermarking is well-founded.

2.6 Transactions — Firing a Transition in a Petri Hypernet

In the hypermarking presented on Fig. 5 one traveler can *board* the plane. Result of firing this transition is described on Fig. 6.

The 'traveler 2' net is sent through a 'virtual channel' connecting the airport and the airplane open nets. As we already mentioned, tokens while traveling

along *virtual channels* cannot be observed. Hence, the middle picture on Fig. 6 does not really describe a valid hypernet and serves as an intuitive explanation of the firing process only.

As the result of boarding the traveler net involved disappears from the horizon of the airport net, and is moved to the plane net postcondition of *board*. Thus, the hierarchy changes.

This single *action* involves *board* transitions on two different levels: two instances of *board* in the airport net, and one instance of *board* in the plane net. Boarding also involves manipulation of two token nets: the plane and a traveler. From this perspective firing looks like a complex *transaction* which involves synchronization of activities on many different levels in the hypernet.

This transactional view of transition firing provides insight into preservation of the well-foundedness of hypermarkings. The point to notice is this. The transaction involves making connections between complementary and well-connected virtual places. In our example there are two such places: the virtual output of *board* in the τ -module of the airport net, and the virtual input of *board* in the τ -module of the plane net. Gluing such connections together establishes temporary channels connecting *proper* places. This follows from the assumption that the modules have the structure of state machines. Thus, each transition has exactly one precondition and exactly one postcondition, either virtual or proper. It is also important to notice that these channels involve modules of the same kind, τ in our example.

The following section of the paper provides a formalization of the ideas informally introduced above.

3 Petri Hypernets — a formalization

Petri hypernets which we are about to define consist of individual nets, called *open nets*. A hypermarking is just a way of saying which open net is used as a token in another open net.

In the formalism presented the open nets will neither be created nor destroyed. Cloning of nets, which we consider as a particular form of creation, is also disallowed. To enforce this policy we define the open nets as synchronous products of *single channel components*, each having a structure of a state machine.

3.1 Channels and Components — Formalization of Kinds and Modules

Let Σ, Δ range over finite sets of *channels*, taken from some fixed countable vocabulary of channel names. We let α, β , etc., range over channel names.

Definition 1. An α -component C is a triple $C = \langle P, T, F \rangle$, where

- P is a finite set of local places of C ,

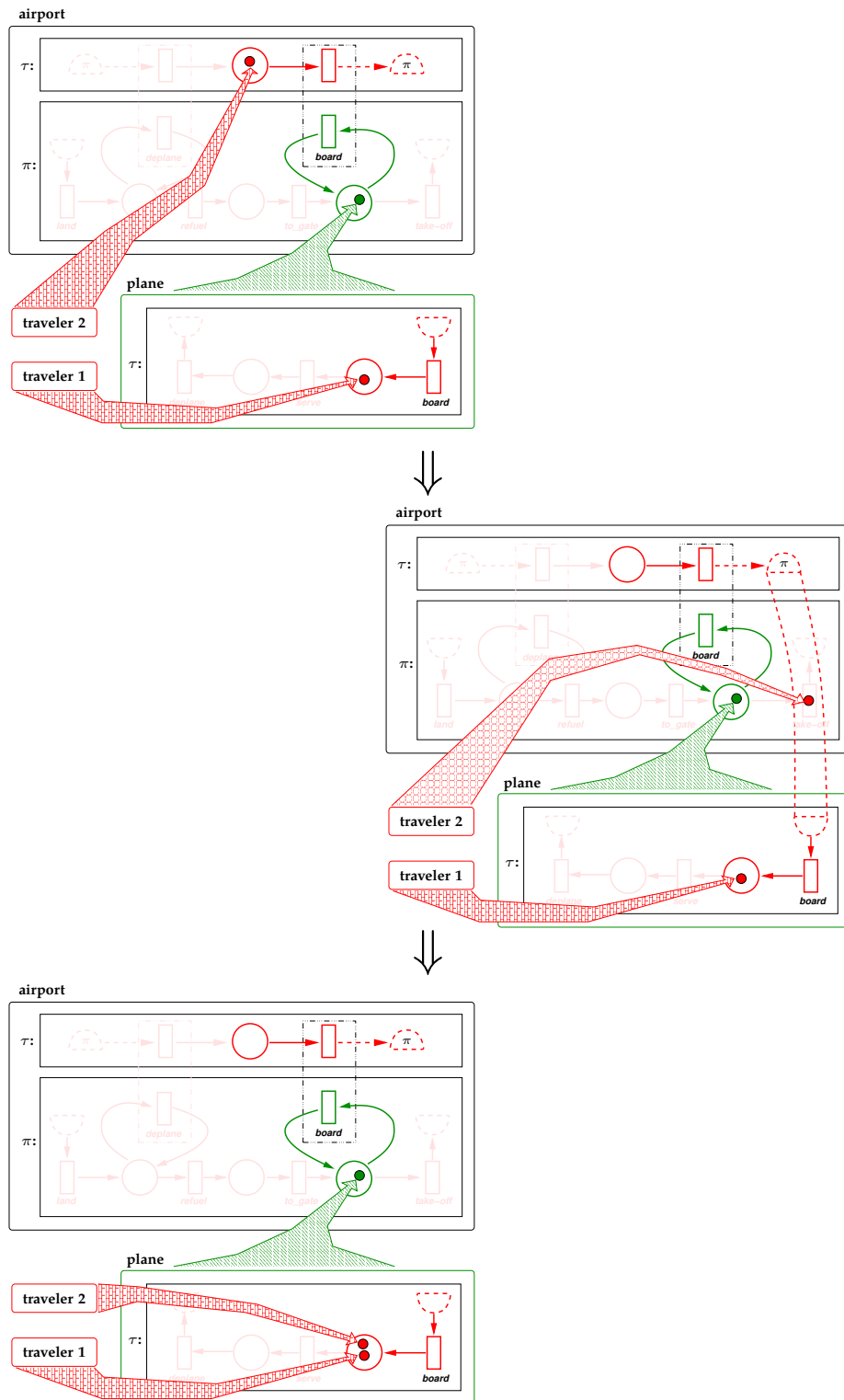


Fig. 6. Firing of a boarding transaction.

- T is a finite set of transitions of C ,
- F is a finite flow relation,

$$F \subseteq (P \cup \{?\} \cup \{?\beta \mid \beta \neq \alpha\}) \times T \cup T \times (P \cup \{!\} \cup \{!\beta \mid \beta \neq \alpha\})$$

such that for every $t \in T$ there exists a unique $p \in P \cup \{?\} \cup \{?\beta \mid \beta \neq \alpha\}$, and a unique $q \in P \cup \{!\} \cup \{!\beta \mid \beta \neq \alpha\}$ such that

$$F(p, t) \quad \text{and} \quad F(t, q) \tag{1}$$

An α -component C is of sort Σ , notation $C:\Sigma$, if $\alpha \in \Sigma$, and $\beta \in \Sigma$ whenever either $F(?\beta, t)$ or $F(t, !\beta)$ hold. Clearly, $C:\Sigma$ and $\Sigma \subseteq \Delta$ implies $C:\Delta$.

Condition (1) states that each α -component is in fact a usual *state machine*, the set of places of which is divided into three disjoint parts. The first, with local places, is mentioned explicitly in the definition. The other two are *virtual inputs* and *virtual outputs*, contained in $\{?\} \cup \{?\beta \mid \beta \neq \alpha\}$ and $\{!\} \cup \{!\beta \mid \beta \neq \alpha\}$, respectively. In accord with the intuitions put forward in the previous section, input $?$ and output $!$ refer to communication with the implicit upper level net. Inputs of the form $?\beta$ and outputs $!\beta$ indicate communications with the lower level nets.

The unique places p and q , local or otherwise, such that $F(p, t)$ and $F(t, q)$ holds are called the *precondition* and *postcondition* of t in C .

3.2 Open Nets — Formalization of Mobile Agents

Suppose that for each $\alpha \in \Sigma$ an α -component of sort Σ is given. Under suitable conditions concerning the inter-component communication, such a collection of components synchronized together forms an *open Petri net* — the counterpart of a mobile agent in our formalization.

Definition 2. An open net N is a pair $N = \langle \Sigma, \{N^\alpha\}_{\alpha \in \Sigma} \rangle$, such that

1. $N^\alpha \hat{=} \langle P^\alpha, T^\alpha, F^\alpha \rangle$ is an α -component of sort Σ , for every $\alpha \in \Sigma$
2. $F^\alpha(?\beta, t)$ implies $(\exists p \in P^\beta).F^\beta(p, t)$, for every $\alpha, \beta \in \Sigma$, $t \in T_N$
3. $F^\alpha(t, !\beta)$ implies $(\exists q \in P^\beta).F^\beta(t, q)$, for every $\alpha, \beta \in \Sigma$, $t \in T_N$
4. $\alpha \neq \beta$ implies $P^\alpha \cap P^\beta = \emptyset$

where T_N denotes the set $\bigcup_{\alpha \in \Sigma} T^\alpha$.

An open net N consists of sequential components N^α , for $\alpha \in \Sigma$. Co-operation of the components within N is enforced by taking the (*partial*) *synchronous product*. That is, one glues the components on identical transitions while keeping their state spaces apart. The latter is ensured by Def. 2.4.

A sequential component formalizes the notion of *module* which copes with specific activities associated with its channel. From this perspective the synchronous product of the components of an open net is a means to force co-operation of different modules, cf Fig. 3. Additionally, and this is specific to

our proposal, the synchronization facilitates the communication between the open net and its tokens, cf. Fig. 6.

Thus, flow $F^\alpha(?\beta, t)$ in the α -component of N signals that the net, while executing t , intends to exchange communication with its token net traveling along channel β . Specifically, it means that transition t in the component N^α wants to read what the token net enabling t in N^β ‘sends up’ on channel α . Note that, by Def. 2.1, N has a β -component. Def. 2.2 states that in such a case the precondition of t in the component N^β has to be a *local place* from P^β . This, as discussed later, ensures the local character of firing a transition.

In terms of our running example, consider N^τ and N^π to be the sequential components of the airport net responsible for managing travelers and planes respectively, see Fig. 3. Then, Def. 2.2 ensures that disembarkment of passengers from a plane requires the presence of the local place in N^τ , which we could think of as an ‘arrival gate’.

Similarly, $F^\alpha(t, !\beta)$ means that while performing t , the component N^α will send *its* token one level down, as a token of the token net traveling along channel β . Def. 2.3 ensures that in this case the postcondition of t in N^β is also a proper place. In our example this situation is demonstrated by the *board* transition and the ‘departure gate’ in the π -component of the airport.

Note also that the above discussion relies on the assumption that the sequential components are indeed state machines, i.e., that each occurrence of transition t has exactly one input, and one output place.

3.3 Hypernets

Let $N = \langle \Sigma, \{N^\alpha\}_{\alpha \in \Sigma} \rangle$ be an open net. In the sequel the notation $P_N = \bigcup_{\alpha \in \Sigma} P^\alpha$ and $T_N = \bigcup_{\alpha \in \Sigma} T^\alpha$ is used to denote the collection of all local places and all transitions of N , respectively.

Definition 3. A Petri hypernet H is a pair $H = \langle \mathcal{N}, m \rangle$, where

- \mathcal{N} is a finite set of open nets.
- $m : \mathcal{N} \rightarrow \bigcup_{N \in \mathcal{N}} P_N$ is a partial function called the hypermarking.

such that the following conditions hold.

1. $N \neq N'$ implies $P_N \cap P_{N'} = \emptyset$, for $N, N' \in \mathcal{N}$.
2. The partial m -depth function $d_m : \mathcal{N} \rightarrow \mathbb{N}$ inductively defined by

$$d_m(N) \cong \begin{cases} 0 & \text{if } m(N) \text{ is undefined} \\ d_m(N') + 1 & \text{if } m(N) \in P_{N'} \end{cases}$$

is total.

Def. 3.1 captures the idea that different mobile agents cannot share any local places. Similarly, sequential modules of an open net have disjoint local spaces.

If the value $m(N)$ is defined, then, by Def. 3.1, there exists a unique $N' \in \mathcal{N}$ such that $m(N)$ is a local place of N' . In this case N is a *token* or *token net* of N' . Thus, the clause in Def. 3.2 indeed defines d_m as a partial function.

The condition of Def. 3.2 really states that the hypermarking of a Petri hypernet is *well-founded* in the sense that its depth function stratifies \mathcal{N} into a finite, forest-like hierarchy. A token net N in H is a *top-level* net iff $d_m(N) = 0$ iff $m(N)$ is undefined.

In the sequel usual conventions apply, e.g., P_H stands for $\bigcup_{N \in \mathcal{N}} P_N$, etc. We write $\bar{m}(N)$ for N' such that $m(N) \in P_{N'}$, and $\chi_m(N)$ for the unique channel α such that $m(N)$ is a place of the α -component of N' . In the sequel the use $\bar{m}(N)$ always implies that $m(N)$ is defined.

3.4 Consortia

In Petri hypernets firing a transition may involve nets at several levels in the current hierarchy. Moreover, as an effect of firing the transition the hierarchy itself may change. All this makes the situation more complex than usual. Yet, the central paradigm underlying Petri net theory is retained — transition firing has *local* and *finitary* character.

We have already argued that with the explicit decomposition of a Petri hypernet into open nets, and these in turn into components, it is appropriate to view the global firing of a transition t as a result of a complex *transaction* which involves these mobile agents and all their modules in which an occurrence of t takes part in reshuffling of tokens involved in the transition.

To start with we consider *consortia*. The idea behind the notion is to take account of all open nets involved in a transition. This involvement may take two forms: the net is being moved around, or it is involved in moving other tokens around. In fact, some token nets may well play both rôles at the same time, as demonstrated by the plane agent in our running example, see Fig.6.

Let $H = \langle \mathcal{N}, m \rangle$ be a hypernet. Consider $t \in T_H$, and a non-empty family \mathcal{T} of open nets containing t , i.e., $N \in \mathcal{T}$ implies $t \in T_N$. For $\mathcal{T}' \subseteq \mathcal{T}$ let us define $\text{Inp}_t(\mathcal{T}') = \{p \in P_H \mid \exists N \in \mathcal{T}'. F_N(p, t)\}$ and call it the set of *t-input-places* of \mathcal{T}' . We use the notation $\text{Inp}_t^\alpha(\mathcal{T}')$ when we restrict attention to places in α -components of the nets from \mathcal{T}' only. If \mathcal{T}' is a singleton $\{N\}$ we simply write $\text{Inp}_t(N)$ instead of $\text{Inp}_t(\{N\})$. The set $\text{Out}_t(\mathcal{T})$ of *t-output-places* for \mathcal{T} is defined analogously.

A *t-consortium* selects the set of open nets \mathcal{T} involved in performing the transition, as well as an input token from every input place of the transition t in any net belonging to \mathcal{T} . This choice is modeled by the function ξ in the definition below.

The definition of a *t-consortium* lists five *structural conditions* which have to be satisfied by the nets from the set \mathcal{T} and the function ξ . First, it is required that the choice of token nets given by ξ agrees with the hypermarking of H , i.e., for any *t-input-place* in \mathcal{T} only a token assigned to this place in the hypermarking can be selected by ξ . The remaining four conditions describe the relationship between a (component of a) *token net* and its *parent net*, which make the *inter-level exchange of tokens* possible. Conditions 2 and 3 describe the situation from the token net point of view, while 4 and 5 take the other perspective into account.

Definition 4. A pair $\langle \mathcal{T}, \xi \rangle$, where $\xi : \text{Inp}_t(\mathcal{T}) \rightarrow \mathcal{N}$, is a t -consortium in H provided the following conditions hold for $N \in \mathcal{T}$.

1. $F_N(p, t)$ implies $m(\xi(p)) = p$, for $p \in \text{Inp}_t(\mathcal{T})$.
2. $F_N^\alpha(? , t)$ implies $\overline{m}(N) \in \mathcal{T} \wedge F_{\overline{m}(N)}^\alpha(t, !\overline{\alpha}) \wedge \xi(\text{Inp}_t^{\overline{\alpha}}(\overline{m}(N))) = N$
where $\overline{\alpha} = \chi_m(N)$.
3. $F_N^\alpha(t, !)$ implies $\overline{m}(N) \in \mathcal{T} \wedge F_{\overline{m}(N)}^\alpha(? \overline{\alpha}, t) \wedge \xi(\text{Inp}_t^{\overline{\alpha}}(\overline{m}(N))) = N$
where $\overline{\alpha} = \chi_m(N)$.
4. $F_N^\alpha(? \beta, t)$ implies $\xi(\text{Inp}_t^\beta(N)) \in \mathcal{T} \wedge F_{\xi(\text{Inp}_t^\beta(N))}^\alpha(t, !)$.
5. $F_N^\alpha(t, !\beta)$ implies $\xi(\text{Inp}_t^\beta(N)) \in \mathcal{T} \wedge F_{\xi(\text{Inp}_t^\beta(N))}^\alpha(? , t)$.

An important observation concerning the notion of consortium is that we do not require \mathcal{T} to contain *all* open nets which contain t among their transitions. It is easy to develop an example in which there are two disjoint t -consortia present in the hypernet. Then, clearly, their union is also a t -consortium.

From the condition 1 it follows that the function ξ is injective.

Condition 2 states that if the α -component of N expects a token from the upper level, then the upper level net $\overline{m}(N)$ exists and belongs to \mathcal{T} . Moreover, the α -component of $\overline{m}(N)$ must be ready to *send something down* to its (unique) $\overline{\alpha}$ -channel such that N is the input token for t in $\overline{m}(N)^\alpha$ chosen by ξ . Channel $\overline{\alpha}$ is the channel through which N travels in the upper level net, i.e., $\overline{\alpha} = \chi_m(N)$.

In terms of the airport example, an instance of this condition would say that in any *board-consortium* if the plane net is ready to accept a passenger, then the airport net must be ready to provide one, and for this to happen the plane has to be at the departure gate (which is a *board-input-place* of the plane-manipulation component of the airport).

Condition 3. is similar, but refers to the ability of *sending* a token *up*.

Above, the unique *upper channel* from both conditions 2. and 3. is slightly ambiguously denoted $\overline{\alpha}$. It is hoped that the precise meaning can always be deduced from the context.

Condition 4. states, that if the α -component of N expects a token to be provided by the token net selected by ξ in the β -precondition of t , then the token net selected for this precondition by ξ is in \mathcal{T} , and has an α -component in which t is ready to send something up. For example, in any *deplane-consortium* if the airport is ready to accept a passenger from a plane, the plane must be at the *deplane-input-place* (i.e., an arrival gate) and has to be ready to send the passenger up to the passenger handling τ -component of the airport.

As we know, transitions in \mathcal{T} may also involve *virtual inputs/outputs*. The set containing both input-places and virtual inputs defined as

$$\text{PreCon}_t(\mathcal{T}) = \{ \langle p, N^\alpha \rangle \mid N \in \mathcal{T} \wedge \alpha \in \Sigma_N \wedge F_N^\alpha(p, t) \}$$

will be called the set of *t-preconditions* in \mathcal{T} . The set of *t-postconditions* in \mathcal{T} is defined in an analogous way and denoted by $\text{PostCon}_t(\mathcal{T})$.

Again, when restricting attention to transitions occurring in α -components of the nets from \mathcal{T} only, we shall write $\text{PreCon}_t^\alpha(\mathcal{T})$ and $\text{PostCon}_t^\alpha(\mathcal{T})$.

3.5 Transactions, or firing a t -consortium

Let $H = \langle \mathcal{N}, m \rangle$ be a hypernet, $t \in T_H$, and \mathcal{T} be a t -consortium in H . As we know, the occurrences of t in the open nets from \mathcal{T} may have virtual inputs/outputs as their preconditions/postconditions. Informally speaking, in the process of firing the consortium the virtual places *along channels* are glued together and become invisible. Token nets are being moved from the *input places* they occupy (via ξ) to the corresponding *output places*. Both input places and output places are ‘proper’, i.e., local places from P_H .

To describe the effect of firing the transition t in H let us first recursively define a family of partial functions:

- $\text{src}_t^\alpha : \text{PreCon}_t^\alpha(\mathcal{T}) \cup \text{PostCon}_t^\alpha(\mathcal{T}) \rightarrow \text{Inp}_t^\alpha(\mathcal{T})$
- $\text{trg}_t^\alpha : \text{PreCon}_t^\alpha(\mathcal{T}) \cup \text{PostCon}_t^\alpha(\mathcal{T}) \rightarrow \text{Out}_t^\alpha(\mathcal{T})$

for all α in the alphabet of \mathcal{T} (i.e. $\alpha \in \bigcup \{ \Sigma_N \mid N \in \mathcal{T} \}$).

For an arbitrary t -precondition/ t -postcondition in \mathcal{T} the functions src_t^α and trg_t^α assign the ‘source input-place’ and the ‘target output-place’ for the particular occurrence of t respectively (both are elements of P_H).

$$\text{src}_t^\alpha(\langle p, N^\alpha \rangle) \hat{=} \begin{cases} p & p \in P_N^\alpha \wedge F_N^\alpha(p, t) \\ \text{src}_t^\alpha(\langle p', N^\alpha \rangle) & F_N^\alpha(t, p) \wedge F_N^\alpha(p', t) \wedge p \neq p' \\ \text{src}_t^\alpha(\langle !\bar{\alpha}, \bar{m}(N)^\alpha \rangle) & p = \text{“?”} \\ \text{src}_t^\alpha(\langle !, \xi(q)^\alpha \rangle) & p = \text{“?}\beta\text{”} \end{cases}$$

$$\text{where } q = \text{Inp}_t^\beta(N)$$

$$\text{trg}_t^\alpha(\langle p, N^\alpha \rangle) \hat{=} \begin{cases} p & p \in P_N^\alpha \wedge F_N^\alpha(t, p) \\ \text{trg}_t^\alpha(\langle p', N^\alpha \rangle) & F_N^\alpha(p, t) \wedge F_N^\alpha(t, p') \wedge p \neq p' \\ \text{trg}_t^\alpha(\langle ?\bar{\alpha}, \bar{m}(N)^\alpha \rangle) & p = \text{“!”} \\ \text{trg}_t^\alpha(\langle ?, \xi(q)^\alpha \rangle) & p = \text{“!}\beta\text{”} \end{cases}$$

$$\text{where } q = \text{src}_t^\beta(\langle \text{Out}_t^\beta(N), N^\beta \rangle)$$

Let us briefly comment on the above definition. The first two clauses in the definition of src_t^α and the definition of trg_t^α are self-explanatory. The third clause in both cases refers to $\bar{m}(N)^\alpha$ i.e., the α -component of the ‘parent net’ of N (wrt. to the hypermarking m). Such a parent net exists by conditions 2 and 3 of the Definition 4 respectively.

The third clause of the definition of src_t^α simply says that to find the *source input-place* of an occurrence of t which *reads* something ‘from above’ we have to look for the source input-place of t in the corresponding component of the parent net. The third clause of the definition of trg_t^α has an analogous interpretation.

Up to this point definitions of src_t^α and trg_t^α were symmetric. The last clauses brake this symmetry for obvious reasons. In the case of src_t^α the situation is simple. We have to search for a matching !-virtual place in the token

provided to t in the α -component of N . This token is in the unique local place q such that $F_N^\beta(q, t)$ holds, i.e., $q = \text{Inp}_t^\beta(N)$.

A symmetric argument in case of trg_t^α would lead to a local place $r = \text{Out}_t^\beta(N)$ which is a *postcondition* of t , so ξ cannot be directly applied to it. To cope with this we have to first compute the local place which is going to end up in r , and only then apply ξ to it.

The following result is crucial to guarantee correctness of our last definition.

Lemma 1. *Let $H = \langle \mathcal{N}, m \rangle$ be a hypernet, $t \in T_H$, and \mathcal{T} be a t -consortium in H .*

1. $\text{src}_t^\alpha \langle p, N^\alpha \rangle$ is defined for any local $p \in \text{Out}_t^\alpha(\mathcal{T})$.
2. $\text{trg}_t^\alpha \langle p, N^\alpha \rangle$ is defined for any local $p \in \text{Inp}_t^\alpha(\mathcal{T})$.

Note that the definition of src_t^α and trg_t^α does not directly refer to $\text{src}_t^\beta / \text{trg}_t^\beta$ for $\beta \neq \alpha$. Intuitively it stresses the fact that the flow of tokens within a hypernet takes place ‘along channels’, i.e., a token can move from an α -component of one open net to a β -component of another open net within H only if β and α are equal (of course other conditions have to be satisfied as well).

Definition 5. *The result of firing a consortium $\langle \mathcal{T}, \xi \rangle$ in a hypernet $\langle \mathcal{N}, m \rangle$ is a hypernet $\langle \mathcal{N}, m' \rangle$ such that:*

$$m'(N) = \begin{cases} \text{trg}_t^\alpha(\langle m(N), \overline{m}(N)^\alpha \rangle) & N \in \xi[\text{Inp}_t^\alpha(\mathcal{T})], m(N) \in \overline{m}(N)^\alpha \\ m(N) & \text{otherwise} \end{cases}$$

i.e., the new hypermarking m' is obtained from the original one by moving all the input tokens designated by ξ to their target output-places given by the appropriate instance of the function trg_t .

Finally, let us formulate the main result. We state it without proof which is quite involved and therefore omitted due to space limitations.

Proposition 1. *Definition 5 is correct. In particular, the new assignment of open nets as tokens which result from the firing of a t -consortium in a hypernet is a well-founded hypermarking.*

4 Conclusion

Petri hypernets, an extension of elementary Petri nets, have been introduced as a means to represent systems of interacting mobile agents. Some basic properties of the model have also been established. The main features of Petri hypernets are the following.

- Local and finitary character of interactions between mobile agents, retained from Petri nets.

This allows to reuse the usual notions from Petri net theory. For instance, two consortia can be fired concurrently, provided their resources are disjoint, etc. In fact, this seems to apply to all ramifications of Valk’s **nets-within-nets** paradigm.

- Limited expressive power.
Clearly, the state space of a finite closed hypernet is finite. Thus, the model promises to capture some essential features without resort to powerful semantical concepts, like free variable generation and binding.
- The *channels* and the *hierarchy* as a support for modularization.
Channels and modules allow separation of concerns with regard to agents of different kinds. The hierarchy helps structure the control flow of mobile agents.
- Flexibility of the hierarchy structure, and inter-level migration.
The ability of open nets to move up and down in the hierarchy is one of the main differences between hypernets and the other ramifications of Valk's **nets-within-nets** paradigm.
- Semantics based on the principle of preservation of mobile agents identity.

The comparable alternative approaches to the problem of specifying systems of mobile agents can be very roughly divided in two classes: the π -calculus and the calculi inspired by it, or akin to it; the net-based models following the paradigm of **nets-within-nets**, proposed by Valk. We have chosen the latter paradigm in order to adhere to the key principles of Petri's net theory: locality of states and transitions, and finiteness of the basic model. This choice sets bounds to the expressive power of the model. In particular, each finite hypernet has a finite state space if it is closed or no interactions with the outside world are made.

A more detailed comparison of our proposal with respect to both trends has been made in the introduction. Let us finish by discussing plans for further work.

The next step we intend to undertake is to develop sound reasoning techniques for verification of properties of Petri hypernets. This should involve defining suitable logics to match the behavioral notions of the model. In particular, the logics have to take the individual character of tokens into account.

Petri hypernets provide a natural model in cases similar to the airport case study. Here, it is natural to think of places as physical locations and of tokens as *physical objects*. However, even in simple generalizations, manipulation of references seems unavoidable. For instance, if passengers were allowed to carry luggage, then the task of collecting the luggage after landing by its owner could be realized by references. The luggage should have a reference to its owner, and the owner could have a reference to its luggage. We plan to investigate if hypernets could be extended to handle also references to agents. Perhaps one of Valk's referential semantics could be adapted here. Within this context it might also be natural to consider the non-well-founded hierarchies.

References

1. P. Andrade and P. Baldan, H. Baumeister, et al. AGILE: Software architecture for mobility. In M. Wirsing, D. Pattinson, and R. Hennicker, editors, *Recent Trends in Algebraic Development Techniques. 16th International Workshop WADT 2002, Frauenchiem-*

- see, Germany, September 24-27, 2002. *Revised Selected Papers*, volume 2755 of LNCS, pages 48–70. Springer-Verlag, 2003.
2. A. Asperti and N. Busi. Mobile Petri nets. Technical Report UBLCS-96-10, Lab. for Computer Science, University of Bologna, Italy, 1996.
 3. M. A. Bednarczyk and A. M. Borzyszkowski. On concurrent realizations of reactive systems and their morphisms. In H. Ehrig, G. Juhas, J. Padberg, and G. Rozenberg, editors, *Unifying Petri nets, Advances in Petri nets*, volume 2128 of LNCS, pages 346–379. Springer-Verlag, 2001.
 4. L. Bernardinello. Synthesis of net systems. In M. A. Marsan, editor, *Application and Theory of Petri Nets 1993, 14th International Conference, Chicago, Illinois, USA, June 21-25, 1993, Proceedings*, volume 691 of LNCS, pages 89–105. Springer-Verlag, 1993.
 5. R. Bruni and U. Montanari. Transactions and zero-safe nets. In H. Ehrig, G. Juhas, J. Padberg, and G. Rozenberg, editors, *Unifying Petri nets, Advances in Petri nets*, volume 2128 of LNCS, pages 346–379. Springer-Verlag, 2001.
 6. L. Cardelli, G. Ghelli, and A. D. Gordon. Mobility types for mobile ambients. In J. Wiedermann, P. van Emde Boas, and M. Nielsen, editors, *Automata, Languages and Programming: 26th International Colloquium, ICALP'99, Proceedings*, volume 1644 of LNCS, pages 230–239. Springer-Verlag, 1999.
 7. C. Fournet and G. Gonthier. The reflexive chemical abstract machine and the Join calculus. In *Proceedings of POPL'96*, pages 372–385, 1996.
 8. M. Köhler, D. Moldt, and H. Rölke. Modelling mobility and mobile agents using nets within nets. In W. van der Aalst and E. Best, editors, *Applications and Theory of Petri Nets 2003, Proceedings*, volume 2679 of LNCS, pages 121–139. Springer-Verlag, 2003.
 9. I. A. Lomazova and P. Schnoebelen. Some decidability results for nested Petri nets. In D. Bjørner, M. Broy, and A. Zamulin, editors, *Perspectives of System Informatics (PSI'99)*, volume 1755 of LNCS, pages 208–220. Springer-Verlag, 2000.
 10. R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, parts 1-2. *Information and Computation*, 100(1):1–77, 1992.
 11. W. Reisig. *Petri Nets*, volume 4 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1985.
 12. R. Valk. On processes of object Petri nets. Technical Report FB-185, Fachbereich Informatik, Universität Hamburg, 1996.
 13. R. Valk. Petri nets as token objects: An introduction to elementary object nets. In W. van der Aalst and E. Best, editors, *Applications and Theory of Petri Nets 1998, Proceedings*, volume 1420 of LNCS, pages 1–25. Springer-Verlag, 1998.
 14. R. Valk. Concurrency in communicating object Petri nets. In G. Agha, F. De Cindio, and G. Rozenberg, editors, *Concurrent Object-Oriented Programming and Petri Nets: Advances in Petri Nets*, volume 2001 of LNCS, pages 164–195. Springer-Verlag, 2001.