# Communication Interference in Mobile Boxed Ambients*

Michele Bugliesi[1], Silvia Crafa[1], Massimo Merro[2], and Vladimiro Sassone[3]

**Abstract.** Boxed Ambients (BA) replace Mobile Ambients' open capability with communication primitives acting across ambient boundaries. Expressiveness is achieved at the price of communication interferences on message reception whose resolution requires synchronisation of activities at multiple, distributed locations. We study a variant of BA aimed at controlling communication interferences as well as mobility ones. Our calculus draws inspiration from Safe Ambients (SA) (with passwords) and modifies the communication mechanism of BA. Expressiveness is maintained through a new form of co-capability that at the same time registers incoming agents with the receiver ambient and performs access control.

## Introduction

The calculus of Mobile Ambients [3] (MA) introduced the notion of ambient acting at the same time as administrative domain and computational environment. Processes live inside ambients, and inside ambients compute and interact. Ambients relocate themselves, carrying along all their contents: their migration, triggered by the processes they enclose, models mobility of entire domains and active computational loci. Two capabilities control ambient movements: in and out. These are performed by processes wishing their enclosing ambient to move to a sibling and, respectively, out of its parent, as show below, where $P$, $Q$ and $R$ are processes, $m$ and $n$ ambient names, $\mid$ is parallel composition, and square brackets delimit ambients' contents.

$$n[\mathsf{in}\ m.P \mid Q] \mid m[R] \longrightarrow m[n[P \mid Q] \mid R], \quad m[n[\mathsf{out}\ m.P \mid Q] \mid R] \longrightarrow n[P \mid Q] \mid m[R]$$

A third capability, open, can be used to dissolve ambients, as expressed by the reduction $\mathsf{open}\ n.P \mid n[Q] \longrightarrow P \mid Q$. Process interaction is by anonymous message exchanges confined inside ambients, as in

$$n[\,\langle M\rangle.P \mid (x).Q\,] \longrightarrow n[P \mid Q\{x := M\}\,],$$

where brackets represent outputs, curly brackets substitutions, and round parentheses bind input variables.

These ideas have given rise to an innovative calculus capturing several aspects of current real-world distributed systems, and have posed some new hard problems. Paper [6] unveiled a set of so-called grave interferences, i.e. situations where the inherent nondeterminism of movement goes wild. For instance, in

$$k[n[\mathsf{in}\ m.P \mid \mathsf{out}\ k.R] \mid m[Q]]$$

* Research supported by 'MyThS: Models and Types for Security in Mobile Distributed Systems', EU FET-GC IST-2001-32617. (1) Università "Ca' Foscari", Venezia; (2) École Polytechnique Fédérale de Lausanne; (3) University of Sussex

although $n$'s next hop is beyond $n$'s control, the difference that such a choice brings about is so big that it is difficult to see how such a situation could have been purposely programmed. Levi and Sangiorgi's proposal of Safe Ambients (SA) [6] counters it by using 'co-actions' to grant ambients a form of control over other ambients' access. A process willing to be entered will manifest that explicitly, as e.g. in

$$n[\mathsf{in}\; m.P \mid Q] \mid m[\overline{\mathsf{in}}\; m.R \mid S] \longrightarrow m[n[P \mid Q] \mid R \mid S],$$

and similarly for $\mathsf{out}$ and $\mathsf{open}$. Building on such infrastructure, a type-system enforced notion of *single-threadedness* ensures that at any time ambients are willing to engage in at most one activity across boundaries that may lead to grave interferences.

Recently, Merro and Hennessy [7] found useful to work with a version of SA called SAP, where incoming ambients must be able to present a suitable password in order to cross ambients' boundaries. Paper [7] develops a treatable semantic theory for SAP in the form of a labelled transition system (LTS) based characterisation of its (reduction) barbed congruence. We will find use for some of these ideas in the present paper too.

Another source of potential problems is $\mathsf{open}$ in its own nature ambient dissolver. A process exercising such a capability will embody all the contents of the dissolved ambient, including its capabilities and migration strategies. Of course there is nothing inherently wrong with that and, as a matter of fact, it is where a large part of the MA's expressiveness resides. For instance, without it the tree structure of a system could never shrink, but only be translated about (or grow). As a consequence, processes would belong to the same ambient for their entire lifespan. Despite its usefulness, from the system designer's point of view $\mathsf{open}$ must be handled with the greatest care.

The calculus of Boxed Ambients [2] (BA) was born out of the observation that, after all, there is an alternative way to yield expressiveness. Namely, direct communication across boundaries as in the Seal calculus [11]. As shown below, in BA it is possible to draw an input from a sub-ambient $n$'s local channel (viz. $(x)^n$) as well as from the parent's local channel (viz. $(x)^\uparrow$), and dually with the roles of input and output swapped.

$$\begin{array}{l}(x)^n.P \mid n[\,\langle M\rangle.Q \mid R\,] \longrightarrow P\{x := M\} \mid n[\,Q \mid R\,]\\ \langle M\rangle.P \mid n[\,(x)^\uparrow.Q \mid R\,] \longrightarrow P \mid n[\,Q\{x := M\} \mid R\,].\end{array}$$

Such design choices, although remarkable in many respects, have the drawback of introducing a great amount of non-local nondeterminism and communication interference. This is exemplified perfectly by the term below, where a single message issued in $n$ unleashes a nondeterministic race among three potential receivers located in three different ambients, possibly belonging to three different administrative domains.

$$m[\,(x)^n.P \mid n[\,\langle M\rangle \mid (x).Q \mid k[\,(x)^\uparrow.R\,]\,]\,]$$

This raises difficulties for a distributed implementation of BA, as there is a hidden, non trivial distributed consensus problem to address at each communication. These forms of interference are as grave as those that led to the definition of SA, as they too should be regarded as programming errors.

In this paper we propose a variant of BA aimed at addressing such issue and at providing a fresh foundation for the ideas behind BA. Our proposal, NBA, takes inspiration from SA(P), and is based on the idea that each ambient comes equipped with two mutually non-interfering channels, respectively for local and upward communications.

$$(x)^n.P \mid n[\langle M\rangle^{\hat{}}.Q \mid R] \longrightarrow P\{x:=M\} \mid n[Q \mid R]$$
$$\langle M\rangle^n.P \mid n[(x)^{\hat{}}.Q \mid R] \longrightarrow P \mid n[Q\{x:=M\} \mid R]$$

Hierarchical communication, whose rules are shown above, is indicated by a pair of distinct constructors, simultaneously on input and output, so that no communication interference is possible. Observe that the upward channel can be thought of as a gateway between parent and child, located at the child's and travelling with it, and poses no particular implementation challenges.

From the theoretical viewpoint, a first consequence of the elimination of unwanted interferences is a set of good, expected algebraic laws for NBA, as illustrated in §1.2. Also, the type system for BA results considerably simplified. In particular, the types of ambients and capabilities need only record upward exchanges, while processes are characterised by their local and hierarchical exchanges. The details will be discussed in §2.

Observe however that limiting ourselves to banning communication interferences as above would result in a poorly expressive calculus (although some of its good properties have been underlined in [4]). For instance, in $n[P]$ there would be no way for $P$ to communicate with its sub-ambients, unless their names were statically known. In our effort to tackle interference we seem to have killed hierarchical communication at all. Far from that, as a matter of fact in order to regain expressive power we only need to reinstate a mechanism for an ambient to learn dynamically the names of incoming ambients. Essentially, our idea is to introduce co-actions of the form $\overline{\mathsf{enter}}(x)$ that have the effect of binding such names to the variable $x$.

Observe that a purely binding mechanism such as this would not in itself be able to control access, but only to registers it. Similarly to SA, it expresses a general willingness to accept incoming ambients; in addition to that, the receiving ambient learns the incoming ambient's name. It can thus be thought as (an abstraction of) an access protocol as it would actually take place in real computational domains, where newly arrived agents would have to register themselves in order to be granted access to local resources. In order to provide ambients with a finer mechanism of access control, we add a second component to our (co-)capabilities and write rules as the one below.

$$a[\mathsf{enter}\langle b,k\rangle.P_1 \mid P_2] \mid b[\overline{\mathsf{enter}}(x,k).Q_1 \mid Q_2] \longrightarrow b[a[P_1 \mid P_2] \mid Q_1\{x:=a\} \mid Q_2]$$

In practical terms, this enhances our access protocol with a form of controlling the credentials of incoming processes ($k$ in the rule above), as a preliminary step to the registration protocol. An example for all of the practical relevance and naturality of this mechanism, is the negotiation of credential that takes place when connecting to a wireless LAN or to a LAN using DHCP or to a ISP using PPP.

Remarkably, our admission mechanism resembles quite closely the notion of passwords as developed in [7], which thus arises yet again as a natural notion to consider. As a consequence, we benefit from results similar to those in *loc. cit.* In particular, we devise a labelled transition semantics (that due to space limitations only appears in the full paper) for NBA that yield a bisimulation congruence sound with respect to (reduction) barbed congruence, and we use it to prove a number of laws. Passwords also have a relevant role in the type system, where their types keep track of the type of (upward exchanges of) incoming ambients, so contributing effectively to a clean and easy type system.

As the paper will show, besides having practical, implementation-oriented features and enjoying good theoretical properties, such as a rich and tractable algebraic theory and a simple type system, at the same time NBA remains expressive enough. In particular, by means of examples and encodings in §3 we show that the expressive power we loose with respect to BA is, as expected and planned, essentially that directly related to communication interferences.

*Structure of the paper.* §1 introduces the calculus, presents the reduction semantics and the associated notion of behavioural equivalence, and presents some of its characteristic algebraic laws. The type system of NBA is illustrated and discussed in §2, as §3 focuses on expressiveness issues in relation to BA and SA, including several examples and an encoding of BA into (an extension of) NBA.

## 1 The NBA Calculus

The syntax of processes is given in the following and is similar to that of BA, except that, as in [7], each of the original capabilities has a co-capability with an extra argument which may be looked upon as a *password*. However, unlike [7], the target ambient of a move gets to know the name of the entering ambient as a result of synchronisation.

*Names:* $a, b, \dots n, x, y, \dots \in \mathbf{N}$

*Locations:*

$$\begin{array}{llll} \eta & ::= & a & \text{nested names} \\ & \mid & \overset{\wedge}{\wedge} & \text{upward} \\ & \mid & \star & \text{local} \end{array}$$

*Messages:*

$$\begin{array}{llll} M, N & ::= & a & \text{name} \\ & \mid & \mathsf{enter}\langle M, N\rangle & \text{enter } M \text{ with } N \\ & \mid & \mathsf{exit}\langle M, N\rangle & \text{exit } M \text{ with } N \\ & & M.N & \text{path} \end{array}$$

*Prefixes:*

$$\pi ::= M \;\mid\; (x_1, \dots, x_k)^\eta \;\mid\; \langle M_1, \dots, M_k\rangle^\eta \;\mid\; \overline{\mathsf{enter}}(x, M) \;\mid\; \overline{\mathsf{exit}}(x, M)$$

*Processes:*

$$\pi ::= \mathbf{0} \;\mid\; P_1 \mid P_2 \;\mid\; (\nu n)P \;\mid\; !\pi.P \;\mid\; M[P] \;\mid\; \pi.P$$

The operators for inactivity, composition, restriction and replication are inherited from mainstream concurrent calculi, most notably the $\pi$-calculus [9]. Specific of ambient calculi are the *ambient* construct, $M[P]$, and the *prefix* via capabilities, $M.P$. Meaningless terms such as $\mathsf{enter}\langle n, p\rangle[P]$ will be ruled out by the type system in §2.

We use a number of notational conventions. Parallel composition has the lowest precedence among the operators. The process $M.N.P$ is read as $M.(N.P)$. We write $\langle\tilde{M}\rangle^\eta, (\tilde{x}), (\nu\tilde{n})$ for $\langle M_1, \dots, M_k\rangle^\eta, (x_1, \dots, x_k)$, $(\nu\, n_1, \dots, n_k)$. We omit trailing dead processes, writing $M$ for $M.\mathbf{0}$, $\langle\tilde{M}\rangle$ for $\langle\tilde{M}\rangle.\mathbf{0}$, and $n[\,]$ for $n[\mathbf{0}]$. Finally, the operators $(\nu n)P$, $\overline{\mathsf{enter}}(x, M).P$, $\overline{\mathsf{exit}}(x, M).P$, and $(\tilde{x})^\eta.P$ act as binders for names $n$, $x$, and $\tilde{x}$, respectively. The set of *free names* of $P$, $\mathrm{fn}(P)$, is defined accordingly.

### 1.1 Reduction and Behavioural Semantics

The dynamics of the calculus is given in the form of a reduction relation $\longrightarrow$ defined as the least relation on processes closed under parallel composition, restriction, and ambient operator which satisfies the following rules.

***mobility***

$$(\textsc{enter})\ n[\mathsf{enter}\langle m,k\rangle.P_1 \mid P_2] \mid m[\overline{\mathsf{enter}}(x,k).Q_1 \mid Q_2] \longrightarrow m[n[P_1 \mid P_2] \mid Q_1\{x := n\} \mid Q_2]$$

$$(\textsc{exit})\quad n[m[\mathsf{exit}\langle n,k\rangle.P_1 \mid P_2] \mid Q] \mid \overline{\mathsf{exit}}(x,k).R \longrightarrow m[P_1 \mid P_2] \mid n[Q] \mid R\{x := m\}$$

***communication***

$$(\textsc{Local})\qquad (\tilde{x}).P \mid \langle\tilde{M}\rangle.Q \longrightarrow P\{\tilde{x} := \tilde{M}\} \mid Q$$

$$(\textsc{input}\ n)\qquad (\tilde{x})^n.P \mid n[\langle\tilde{M}\rangle^{\hat{}}.Q \mid R] \longrightarrow P\{\tilde{x} := \tilde{M}\} \mid n[Q \mid R]$$

$$(\textsc{output}\ n)\qquad \langle\tilde{M}\rangle^n P \mid n[(\tilde{x})^{\hat{}}.Q \mid R] \longrightarrow P \mid n[Q\{\tilde{x} := \tilde{M}\} \mid R]$$

***structural congruence***

$$(\textsc{str cong})\qquad P \equiv Q \quad Q \longrightarrow R \quad R \equiv S \text{ implies } P \longrightarrow S$$

As customary in process calculi, the *reduction semantics* is based on an auxiliary relation called *structural congruence*, $\equiv$, which brings the participants of a potential interaction to contiguous positions. Its formal definition is similar to that for BA [2]. The reduction rules are divided in two groups: the *mobility rules* and the *communication rules*.

As in [7], mobility requires the ambients involved to agree on a password $k$. In addition, the target of the move learns the name of the entering ambient as a result of synchronisation. Similar ideas apply to the (EXIT) rule: in particular, the co-exit capability is placed in the target ambient, as in [7]. Unlike *loc. cit.*, however, the co-action does not mention the name of the travelling ambient.

The communication rules are a simplification of those of BA. As in BA communication is *synchronous*, *polyadic*, and, most importantly, *located* and *across* boundaries. In both rules (INPUT $n$) and (OUTPUT $n$) the underlying anonymous channel is to be thought of as allocated at the sub-ambient $n$. In all communication rules we assume that tuples have the same arity, a condition that later will be enforced by the type system.

As to behavioural equivalences we focus on *reduction barbed congruence* [5], a standard equality based on the notions of reduction relation and observability. In ambients the observation predicate $P\downarrow_n$ is used to denote the possibility of process $P$ of interacting with the environment via the ambient $n$. In MA [3] this happens whenever $P \equiv (\nu\tilde{m})(n[P_1]|P_2)$, for $n \notin \{\tilde{m}\}$: since no authorisation is required to cross a boundary, the presence of an ambient $n$ at top level denotes a potential interaction between the process and the environment via $n$. In the presence of co-capabilities, however, the process $(\nu\tilde{m})(n[P_1]|P_2)$ only represents a potential interaction if $P_1$ can exercise an appropriate co-capability.

**Definition 1 (Barbs).** *We write $P\downarrow_n$ iff $P \equiv (\nu m)(n[\overline{\mathsf{enter}}(x,k).Q \mid R] \mid S)$ for $\{n,k\} \cap \{m\} = \emptyset$. We write $P\Downarrow_n$ if $P \Longrightarrow P'$ and $P'\downarrow_n$.*

**Definition 2.** *A relation $\mathscr{R}$ is: **(i)** reduction closed if $P\mathscr{R}Q$ and $P \to P'$ implies the existence of some $Q'$ such that $Q \Rightarrow Q'$ and $P'\mathscr{R}Q'$; **(ii)** barb preserving if $P\mathscr{R}Q$ and $P\downarrow_n$ implies $Q\Downarrow_n$.*

**Definition 3 (Reduction Barbed Congruence).** *Reduction barbed congruence, written $\cong$, is the largest congruence relation over processes which is reduction closed and barb preserving.*

Notice that in our setting there is at least another significant choice of barb, reflecting the other form of interaction an ambient may engage with the context, viz. via communication. In particular, we could reasonably define that $P$ has a barb at $n$ if and only if $P \equiv (\nu m)(n[\langle\cdot\rangle^{\hat{}}.P' \mid Q'] \mid Q'')$ for $n \notin \{m\}$. However, we can show that either choices of barb result in the same equivalence. Let $\downarrow_n^{\mathsf{e}}$ denote the alternative predicate we just define, and $\cong_{\mathsf{e}}$ the resulting equivalence.

**Proposition 1.** *$P \cong Q$ if and only if $P \cong_{\mathsf{e}} Q$.*

### 1.2 Algebraic Laws

In the remainder of the section we give a collections of laws which hold in NBA. All these laws can be easily proved using our notion of bisimilarity by exhibiting the appropriate bisimulation relation. In most cases the bisimulation relation contains only the pair of the processes of the law plus the identity relation. The first set of laws is about garbage collection: for $I = J = H = \emptyset$ one also derives $l[\,] = \mathbf{0}$.

**Theorem 1 (Algebraic Laws).**

**Garbage Collection Laws**

1. $l[\ \Pi_{i\in I}(\tilde{x}_i)^{n_i}.P_i \mid \Pi_{j\in J}(\tilde{x}_j).P_j \mid \Pi_{h\in H}\langle\tilde{M}\rangle^{m_h}.P_h\ ] \ \cong\ \mathbf{0}$
2. $l[\ \Pi_{i\in I}(\tilde{x}_i)^{n_i}.P_i \mid \Pi_{j\in J}\langle\tilde{M}_j\rangle.P_j \mid \Pi_{h\in H}\langle\tilde{M}_h\rangle^{m_h}.P_h\ ] \ \cong\ \mathbf{0}$

**Communication Laws** *If $|\tilde{x}|=|\tilde{M}|$ then:*

1. $l[\ \Pi_{j\in J}\langle\tilde{M}_j\rangle^{\hat{}}\ ] \ \cong\ \Pi_{j\in J}\, l[\langle\tilde{M}_j\rangle^{\hat{}}\ ]$
2. $l[(\tilde{x}).P \mid \langle\tilde{M}\rangle.Q] \ \cong\ l[P\{\tilde{x} := \tilde{M}\} \mid Q]$
3. $(\nu l)(\,(\tilde{x})^{l}.P \mid l[\langle\tilde{M}\rangle^{\hat{}}.Q]\,) \ \cong\ (\nu l)(\,P\{\tilde{x} := \tilde{M}\} \mid l[Q]\,)$
4. $m[(\tilde{x})^{l}.P \mid l[\langle\tilde{M}\rangle^{\hat{}}.Q]] \ \cong\ m[P\{\tilde{x} := \tilde{M}\} \mid l[Q]]$
5. *Dual laws of 3 and 4 resulting from exchanging input with output prefixes*

**Mobility Laws**

1. $(\nu p)(m[\mathsf{enter}\langle n,p\rangle.P] \mid n[\overline{\mathsf{enter}}(x,p).Q]) \ \cong\ (\nu p)(n[Q\{x := m\} \mid m[P]])$
2. $l[m[\mathsf{enter}\langle n,p\rangle.P] \mid n[\overline{\mathsf{enter}}(x,p).Q]] \ \cong\ l[n[Q\{x := m\} \mid m[P]]]$
3. $(\nu p)(n[m[\mathsf{exit}\langle n,p\rangle.P]] \mid \overline{\mathsf{exit}}(x,p).Q) \ \cong\ (\nu p)(m[P] \mid Q\{x := m\})$
4. $l[n[m[\mathsf{exit}\langle n,p\rangle.P]] \mid \overline{\mathsf{exit}}(x,p).Q] \ \cong\ l[m[P] \mid Q\{x := m\}]$. $\square$

The first law says how parallel composition of upward *asynchronous* messages distributes on the ambient construct. The same law holds for BA; in neither case (BA, or NBA) it extends to the case of (upward) output prefixes with non-nil continuation. Law 2 shows that NBA is free of interferences on local communications: it holds in SA but not in MA and BA. The remaining laws are peculiar of NBA and show that, unlike BA, no interference on upward communications is possible.

## 2 The Type System

The absence of interferences in communication has other interesting payoffs besides those we have discussed in the previous section. Specifically, as we illustrate below, it greatly simplifies the type system over the original definition for BA [2]. The structure of the types is as follows.

| | | | | |
|---|---|---|---|---|
| *Message Types* | $W$ | $::=$ | $\mathsf{N}[E]$ | ambient/password |
| | | $\mid$ | $\mathsf{C}[E]$ | capability |
| *Exchange Types* | $E,F$ | $::=$ | $\mathsf{shh}$ | no exchange |
| | | $\mid$ | $W_1 \times \cdots \times W_k$ | tuples ($k \geq 0$) |
| *Process Types* | $T$ | $::=$ | $[E,F]$ | composite exchange |

The main difference with respect to the type systems of [2] and [8] is in the structure of ambient types, that are defined here as one-place constructors of the form $\mathsf{N}[E]$, tracing the upward exchanges of ambients with this type. This simplification over previous systems (in which ambient types are two-place constructors) is a direct consequence of the semantics of communication, which guarantees the absence of interferences between the local exchanges of an ambient and the upward exchanges of its nested sub-ambients.

In addition, in the present system types of the form $\mathsf{N}[E]$ also serve as the types of passwords. Hence, $\mathsf{N}[E]$ is indeed the class of *name* types. When used as a password type, $\mathsf{N}[E]$ enables upward exchanges of type $E$ for any process that relies on a password with this type to cross an ambient boundary. There is no type confusion in this double role of name types, as different uses of a name have different imports in the typing rules. An alternative would be to use two different constructors for ambient and password names: this, however, would have the undesired effect of disallowing the use of the same name in the two roles, a feature that is harmless and actually rather convenient in many examples.

The remaining types have the same interpretation as in previous type systems. In particular, $\mathsf{C}[E]$ is the type of capabilities exercised within ambients with upward exchange of type $E$, and $[E,F]$ is the type of processes with local exchange of type $E$ and upward exchanges of type $F$. The role of the type $\mathsf{shh}$ also is similar to that played by the corresponding type in the type system of BA. As in that case, it indicates absence of exchanges, and can be assigned to processes that do not have any local or upward exchange. In addition, in the type systems for NBA, $\mathsf{shh}$ provides for a *silent* mode for mobility in a way similar to, but substantially simpler than, the *moded types* of [2]. Specifically, the typing rules guarantee that a boundary crossing, say by ambient $n$, via a password of type $\mathsf{N}[\mathsf{shh}]$ involves no learning of names. Thus, unless the target ambient knows the name $n$ statically, the use of a $\mathsf{N}[\mathsf{shh}]$ password guarantees that ambients can move irrespective of their upward exchanges.

We proceed with the presentation of the typing rules. Table 1 gives the rules for valid type environments and for messages. The relation $F \leqslant G$, with $F$ and $G$ exchange types, holds true if and only if $F \in \{\mathsf{shh}, G\}$. The rules (ENTER) and (EXIT) define

$$\textsc{(Env }\varnothing\textsc{)}\ \frac{}{\varnothing \vdash \diamond} \qquad \textsc{(Env name)}\ \frac{\Gamma \vdash \diamond \quad a \notin \mathrm{Dom}(\Gamma)}{\Gamma, a : W \vdash \diamond} \qquad \textsc{(Projection)}\ \frac{\Gamma, a : W, \Gamma' \vdash \diamond}{\Gamma, a : W, \Gamma' \vdash a : W} \qquad \textsc{(Path)}\ \frac{\Gamma \vdash M_1 : \mathsf{C}[E_1] \quad \Gamma \vdash M_2 : \mathsf{C}[E_2]}{\Gamma \vdash M_1.M_2 : \mathsf{C}[E_1 \sqcup E_2]}$$

$$\textsc{(Enter)}\ \frac{\Gamma \vdash M : \mathsf{N}[E] \quad \Gamma \vdash N : \mathsf{N}[F] \quad (F \leqslant G)}{\Gamma \vdash \mathsf{enter}\langle M, N\rangle : \mathsf{C}[G]} \qquad \textsc{(Exit)}\ \frac{\Gamma \vdash M : \mathsf{N}[E] \quad \Gamma \vdash N : \mathsf{N}[F] \quad (F \leqslant G)}{\Gamma \vdash \mathsf{exit}\langle M, N\rangle : \mathsf{C}[G]}$$

**Table 1.** Good Messages: $\Gamma \vdash M : W$

$$\textsc{(Par)}\ \frac{\Gamma \vdash P : [E,F] \quad \Gamma \vdash Q : [E,F]}{\Gamma \vdash P \mid Q : [E,F]} \qquad \textsc{(Repl)}\ \frac{\Gamma \vdash P : [E,F]}{\Gamma \vdash\ !P : [E,F]} \qquad \textsc{(Dead)}\ \frac{\Gamma \vdash \diamond}{\Gamma \vdash \mathbf{0} : [E,F]} \qquad \textsc{(New)}\ \frac{\Gamma, n : \mathsf{N}[G] \vdash P : [E,F]}{\Gamma \vdash (\nu n{:}\mathsf{N}[G])P : [E,F]}$$

**Table 2.** Good processes I: $\Gamma \vdash P : [E,F]$

the types of capabilities in terms of the type of the component passwords: together with the typing rules for processes, ambients and mobility of Table 3, they construe the types of passwords as *interfaces* for mobility. In particular, if the type $F$ associated with password $N$ is a message type $W$, then $N$ constrains any ambient relying upon it for mobility to have upward exchanges of type $W$ (cf. rules (PREFIX) and (AMB) in Table 3). If instead $F = \mathsf{shh}$, then $N$ enforces no constraint, as the type $G$ in the conclusion of the rule can be any exchange type. Rule (PATH) follows the same intuition: it is applicable only when $E_1 \sqcup E_2$ exists, $\sqcup$ being the lub associated with $\leqslant$.

Tables 2 to 4 define the typing of processes. The rules in Table 2 are standard. The rules in Table 3 complement those in Table 1 in governing mobility. Rule (AMB) is standard, and construes the type $\mathsf{N}[E]$ as the interface of ambient $M$ for any process which knows name $M$: any such process may have sound $E$ exchanges with $M$, as the process enclosed within $M$ has upward exchanges of this type. The rules for the mobility co-actions provide similar guarantees for the exchanges a process may have with ambients whose name the process learns by exercising the co-capability. In this case, it is the type of the password $M$ that acts as interface: if $M$ has type $\mathsf{N}[\tilde{W}]$, as in the rules (CO-ENTER) and (CO-EXIT), we are guaranteed that $\tilde{W}$ is indeed the type of the exchanges of the incoming ambient. If instead the password type is $\mathsf{N}[\mathsf{shh}]$, then no such guarantee can be provided, as it is easily verified by an inspection of the (PREFIX) rule and of the rules for communication in Table 4. To recover soundness, rules (CO-ENTER/EXIT-SILENT) require that no use be made of the continuation process $P$ of the variable $x$ (hence of the name of the incoming ambient, unless the name was known already to $P$). An alternative sound solution would be to generalise the (CO-ENTER) and (CO-EXIT) rules by (systematically) replacing the type $\tilde{W}$ with a generic exchange type $G$. Although such

$$\frac{\Gamma \vdash M : \mathsf{N}[E] \quad \Gamma \vdash P : [F,E]}{\Gamma \vdash M[P] : [G,H]}\ (\textsc{Amb}) \qquad \frac{\Gamma \vdash M : \mathsf{C}[F] \quad \Gamma \vdash P : [E,G] \quad (F \leqslant G)}{\Gamma \vdash M.P : [E,G]}\ (\textsc{Prefix})$$

$$\frac{\Gamma \vdash M : \mathsf{N}[\tilde{W}] \quad \Gamma, x : \mathsf{N}[\tilde{W}] \vdash P : [E,F]}{\Gamma \vdash \overline{\mathsf{enter}}(x,M).P : [E,F]}\ (\textsc{Co-enter}) \qquad \frac{\Gamma \vdash M : \mathsf{N}[\tilde{W}] \quad \Gamma, x : \mathsf{N}[\tilde{W}] \vdash P : [E,F]}{\Gamma \vdash \overline{\mathsf{exit}}(x,M).P : [E,F]}\ (\textsc{Co-exit})$$

$$\frac{\Gamma \vdash M : \mathsf{N}[\mathsf{shh}] \quad \Gamma \vdash P : [E,F] \quad (x \notin \mathrm{fv}(P))}{\Gamma \vdash \overline{\mathsf{enter}}(x,M).P : [E,F]}\ (\textsc{Co-enter-silent}) \qquad \frac{\Gamma \vdash M : \mathsf{N}[\mathsf{shh}] \quad \Gamma \vdash P : [E,F] \quad (x \notin \mathrm{fv}(P))}{\Gamma \vdash \overline{\mathsf{exit}}(x,M).P : [E,F]}\ (\textsc{Co-exit-silent})$$

**Table 3.** Good Processes II (mobility)

$$\frac{\Gamma, \tilde{x}:\tilde{W} \vdash P : [\tilde{W},E]}{\Gamma \vdash (\tilde{x}:\tilde{W}).P : [\tilde{W},E]}\ (\textsc{Input}) \qquad \frac{\Gamma, \tilde{x}:\tilde{W} \vdash P : [E,\tilde{W}]}{\Gamma \vdash (\tilde{x}:\tilde{W})^{\hat{}}.P : [E,\tilde{W}]}\ (\textsc{Input}\ \hat{}\,) \qquad \frac{\Gamma \vdash M : \mathsf{N}[\tilde{W}] \quad \Gamma, \tilde{x}:\tilde{W} \vdash P : [G,H]}{\Gamma \vdash (\tilde{x}:\tilde{W})^{M}.P : [G,H]}\ (\textsc{Input}\ M)$$

$$\frac{\Gamma \vdash \tilde{M} : \tilde{W},\ \Gamma \vdash P : [\tilde{W},E]}{\Gamma \vdash \langle \tilde{M} \rangle.P : [\tilde{W},E]}\ (\textsc{Output}) \qquad \frac{\Gamma \vdash \tilde{M} : \tilde{W},\ \Gamma \vdash P : [E,\tilde{W}]}{\Gamma \vdash \langle \tilde{M} \rangle^{\hat{}}.P : [E,\tilde{W}]}\ (\textsc{Output}\ \hat{}\,) \qquad \frac{\Gamma \vdash N : \mathsf{N}[\tilde{W}],\ \Gamma \vdash \tilde{M} : \tilde{W},\ \Gamma \vdash P : [G,H]}{\Gamma \vdash \langle \tilde{M} \rangle^{N}.P : [G,H]}\ (\textsc{Output}\ N)$$

**Table 4.** Good Processes III (input/output)

an approach would allow to dispense with rules (CO-ENTER/EXIT-SILENT) and (CO-EXIT-SILENT), the resulting system would be less general than the present one, in that ambients using silent passwords (i.e. of type $\mathsf{N}[\mathsf{shh}]$) for mobility would be required to be upward silent. Notice, in fact, that we impose no such constraint: the typing rules only prevent upward exchanges with the processes enclosed in ambients reached by the use of a silent password.

The type system is sound, as expected.

**Proposition 2 (Subject Reduction).** *If $\Gamma \vdash P : T$ is a derivable judgement in NBA, and $P \longrightarrow Q$, then $\Gamma \vdash Q : T$ is also a derivable judgement.*

## 3 Examples

We illustrate the expressive power of NBA and its algebraic theory with several examples. The first is a standard expressiveness test: the encoding of the pi-calculus [9].

### 3.1 Encoding of Channels

Among the possible solutions, we present the most compact one, which further illustrates the power of our co-actions. We only show the encoding of channels, the remaining clauses are defined homomorphically.

$$\begin{aligned}
\{\!|\, \overline{c}\langle \tilde{M}\rangle P \,|\!\} &\triangleq (\nu a)\,(\ c[\langle \tilde{M}\rangle^{\hat{}}.c[\mathsf{exit}\langle c,a\rangle]] \mid \overline{\mathsf{exit}}(x,a).\{\!|\, P \,|\!\}\ )\\
\{\!|\, c(\tilde{x})P \,|\!\} &\triangleq (\tilde{x})^{c}.\{\!|\, P \,|\!\}
\end{aligned}$$

Given the direct nature of the encoding, its operational correctness is simple to prove. Let $\gtrsim$ denote the *expansion* relation [1], an asymmetric variant of $\approx$ such that $P \gtrsim Q$ holds if $P \approx Q$, and $P$ has at least as many $\tau$-moves as $Q$.

**Lemma 1 (Operational Correspondence).** *If $P \xrightarrow{\ \tau\ } P'$ then $\{\!|\, P \,|\!\} \xrightarrow{\ \tau\ } \gtrsim \{\!|\, P' \,|\!\}$. Conversely, if $\{\!|\, P \,|\!\} \xrightarrow{\ \tau\ } Q$, then there exists $P'$ such that $P \xrightarrow{\ \tau\ } P'$ and $Q \gtrsim \{\!|\, P' \,|\!\}$.*

*Proof.* By transition induction. Expansion relations are derived by law in Theorem 1.3.

The result extends to weak reductions. Based on that, and on the compositionality of $\{\!|\cdot|\!\}$, we can show that the encoding is sound in the following sense.

**Theorem 2 (Equational Soundness).** *If $\{\!|\, P \,|\!\} \cong \{\!|\, Q \,|\!\}$ then $P \cong Q$.*

### 3.2 NBA versus BA and SA

The next suite of examples relates NBA and its 'parent' calculi BA and SA, reinforcing our claim that NBA removes non-determinism from BA, and providing more a formal statement about their relative expressive power.

**A one-to-one communication server** Our first example is a system that represents a server for point-to-point communication.

$$\mathsf{w}(k) \ = \ k[\ \overline{\mathsf{enter}}(x,k).\overline{\mathsf{enter}}(y,k).(!(z)^{x}.\langle z\rangle^{y} \mid !(z)^{y}.\langle z\rangle^{x})\ ]$$

Here $\mathsf{w}(k)$ is a bidirectional forwarder for any pair of incoming ambients. An agent willing to participate in a point-to-point communication must know the password $k$ and should be implemented as the process $A(a,k,P,Q) = a[\mathsf{enter}\langle k,k\rangle.P \mid \mathsf{exit}\langle k,k\rangle.Q]$, with $P$ performing the expected (upward) exchanges. A complete implementation of the point-to-point server can be then defined as shown below:

$$\mathsf{o2o}(k) = (\nu r)\ (\ r[\langle\rangle^{\hat{}}\,] \mid !\,()^{r}.(\mathsf{w}(k) \mid \overline{\mathsf{exit}}(\_,k).\overline{\mathsf{exit}}(\_,k).r[\langle\rangle^{\hat{}}\,])\ )$$

The process $\mathsf{o2o}(k)$ accepts a pair of ambients within the forwarder, provides them with the necessary support of the point-to-point exchange and then lets them out before preparing a new instance of $\mathsf{w}(k)$ for a new protocol session. Given the configuration $\mathsf{o2o}(k) \mid A(k,a_1,P_1,Q_1) \mid \cdots \mid A(k,a_n,P_n,Q_n)$, we are guaranteed that at most one pair of agents can be active within $k$ at any given time ($k$ is locked until the two ambients are inside $k$). In particular, one has:

$$(\nu k)(\ \mathsf{o2o}(k) \mid A(k,a_1,\langle M\rangle^{\hat{}}.P_1,Q_1) \mid A(k,a_2,(x)^{\hat{}}.P_2\{x\},Q_2) \mid \Pi_{i\in I}A(K,a_i,R_i,S_i)\ )$$
$$\Longrightarrow\cong\ (\nu k)(\ \mathsf{o2o}(k) \mid a_1[P_1 \mid Q_1] \mid a_2[P_2\{x:=M\} \mid Q_2] \mid \Pi_{i\in I}A(K,a_i,R_i,S_i)\ )$$

This says that once (and if) the two agents have reached the forwarder, no other agent knowing the key $k$ can interfere and prevent them from completing their exchange. The equivalence above follows by the 'garbage collection' and the 'mobility' laws of Theorem 1. In particular, once the two ambients are back at top level, the currently active instance of the forwarder $k$ has the form $k[\,!(z)^{a_1}.\langle z\rangle^{a_2}|!(z)^{a_2}.\langle z\rangle^{a_1}] \cong \mathbf{0}$.

The use of the forwarder to implement a point-to-point communication protocol may at first appear artificial, for it would seem that two ambients wishing to communicate are likely to know their partner's name, and could then interact without intervention of a third party. Indeed, in NBA the example can be simplified with these assumptions. In BA, instead, the knowledge of names still poses a challenge due to possible communication interferences. To illustrate the point, let us consider the following BA coding of the protocol.

$$a[\mathsf{in}\ b.\mathsf{in}\ k.P] \mid b[k[!(x).\langle x\rangle^a \mid !(x)^a.\langle x\rangle] \mid Q]$$

Here $Q$ can read from and write to $k$ to exchange values with $P$ inside $a$, but it is not obvious what $P$ should do. Clearly, with $k$ as above, it must use local communication to talk with $k$, and hence with $Q$. However, this would not avoid interference with its own local exchanges. Not is it clear how to ban such unwanted effect, as there seem to be similar problems with several different implementations of $k$. For instance, using $k[!(x).\langle x\rangle]$ $a$ and $b$ may end up re-reading their own messages; while with $k[!(x).\langle x\rangle \mid (x)^{\uparrow}.\langle x\rangle^a]$ the upward read by $k$ may mistakenly synchronise with local output in $b$.

**A print server** Our next example implements a print server to print jobs arriving off the network in the order of arrival. We give the implementation in steps. First consider the following process that assigns a progressive number to each incoming job.

$$\mathsf{enqueue}(k) =\ (\nu c)\ (\ c[\langle 1\rangle^{\hat{}}] \mid !(n)^c.\overline{\mathsf{enter}}(x,k).\langle n\rangle^x.c[\langle n+1\rangle^{\hat{}}])$$

The (private) ambient $c$ holds the current value of the counter. The process accepts a job and delivers it the current number. Then, it updates the counter and prepares for the next round. Now, we can show how this can be turned into a print server ($\mathsf{enqueue}(k)$ is defined as above):

$$\begin{aligned}
\mathsf{prtsrv}(k) &= k[\ \mathsf{enqueue}(k) \mid \mathsf{print}\ ]\\
\mathsf{print} &=\ (\nu c)\ (\ c[\langle 1\rangle^{\hat{}}] \mid !(n)^c.\overline{\mathsf{exit}}(x,n).(data)^x.(P\{data\} \mid c[\langle n+1\rangle^{\hat{}}])\\
\mathsf{job}(M,s) &= (\nu p)p[\ \mathsf{enter}\langle s,s\rangle.(n)^{\hat{}}.(\nu q)q[\mathsf{exit}\langle p,n\rangle.\langle M\rangle^{\hat{}}\,]\,]
\end{aligned}$$

Process $\mathsf{job}(M,s)$ enters the server $\mathsf{prtsrv}(s)$, it is assigned a number to be used as a password for carrying the job $M$ to the printer process $P$ (note that the use of passwords is critical here).

This situation appears hard to implement with SA(P) or BA. In SA(P) because one would need to know the names of the incoming jobs to be able to assign them their numbers. In BA because dequeuing the jobs (according to the intended FIFO policy) requires a test of the number a job has been assigned, and an atomic implementation of such test appears problematic.

**BA $\lesssim$ NBA + Guarded Choice.** In order to relate BA and NBA more formally, and support our claim that the only loss of expressiveness in the passage is very directly related the removal of grave communication interferences, we present an encoding of BA into an extended version of NBA. Precisely, we enrich NBA with a limited, focused form of nondeterminism, viz. a sum operator, that we use in the encoding to circumscribed the nondeterminism in communication typical of BA. Besides showing the relationship between the two calculi, this has the further advantage of localising their differences in a single construct.

The encoding is defined parametrically over four names $n, \mathsf{mv}, \mathsf{pr}, \mathsf{pw}$: $n$ is the name of the ambient (if any) that encloses the process that we are encoding, while the remaining three names are used as passwords. To ease the notation, we use the following shorthands: $\overline{\mathsf{cross}} = !\overline{\mathsf{enter}}(x, \mathsf{mv}) \mid !\overline{\mathsf{exit}}(x, \mathsf{mv})$, $\mathsf{in}\ n = \mathsf{enter}\langle n, \mathsf{mv}\rangle$, and $\mathsf{out}\ n = \mathsf{exit}\langle n, \mathsf{mv}\rangle$. The encoding is defined in terms of two mutually recursive translations, $\{\!|\cdot|\!\}_n$ and $\langle\!\langle \cdot \rangle\!\rangle_n$, with $\{\!|\cdot|\!\}_n$ leading. The interesting cases are given below, where we assume the implicit side condition that $p, y \notin \mathrm{fn}(P)$.

$$
\begin{aligned}
\{\!| P |\!\}_n &= \overline{\mathsf{cross}} \mid \langle\!\langle P \rangle\!\rangle_n \\
\langle\!\langle m[P] \rangle\!\rangle_n &= m[\{\!| P |\!\}_m] \\
\langle\!\langle (x)^a P \rangle\!\rangle_n &= (x)^a \langle\!\langle P \rangle\!\rangle_n \\
\langle\!\langle (x) P \rangle\!\rangle_n &= (x) \langle\!\langle P \rangle\!\rangle_n + (x)^{\hat{}} \langle\!\langle P \rangle\!\rangle_n + \overline{\mathsf{exit}}(y, \mathsf{pw})(x)^y \langle\!\langle P \rangle\!\rangle_n \\
\langle\!\langle (x)^{\uparrow} P \rangle\!\rangle_n &= (\nu p) p[\mathsf{exit}\langle n, \mathsf{pr}\rangle.(x)^{\hat{}}.\mathsf{enter}\langle n, p\rangle.\langle x\rangle^{\hat{}}\,] \mid \overline{\mathsf{enter}}(y, p)(x)^y \langle\!\langle P \rangle\!\rangle_n \\
\langle\!\langle \langle M\rangle^a P \rangle\!\rangle_n &= \langle M\rangle^a \langle\!\langle P \rangle\!\rangle_n \\
\langle\!\langle \langle M\rangle P \rangle\!\rangle_n &= \langle M\rangle \langle\!\langle P \rangle\!\rangle_n + \langle M\rangle^{\hat{}} \langle\!\langle P \rangle\!\rangle_n + \overline{\mathsf{exit}}(y, \mathsf{pr})\langle M\rangle^y \langle\!\langle P \rangle\!\rangle_n \\
\langle\!\langle \langle M\rangle^{\uparrow} P \rangle\!\rangle_n &= (\nu p) p[\mathsf{exit}\langle n, \mathsf{pw}\rangle.\langle M\rangle^{\hat{}}.\mathsf{enter}\langle n, p\rangle.(\,)^{\hat{}}\,] \mid \overline{\mathsf{enter}}(y, p)(\,)^y \langle\!\langle P \rangle\!\rangle_n
\end{aligned}
$$

The remaining cases are defined homomorphically.

The encoding $\{\!|\cdot|\!\}_n$ provides unboundedly many co-capabilities, at all nesting levels, so that ambient mobility in BA is rendered faithfully. As for the translation of the communication primitives, the intuition is the following. The upward exchanges of a BA term are dealt with by the taxi ambients that exit the enclosing ambient $n$ to deliver output (or collect input) and then return to $n$ to unlock the continuation $P$. The use of restricted names as passwords is essential here for the continuation $P$ to be able to identify its helper taxi ambient without risk of confusion. As for the translation of a local input/output, the three branches of the choice reflect the three possible synchronisations: local, from upward, from a nested ambient. Note, in particular, that the right-most branch of these choices may only match upward requests that are encodings of upward requests: this is guaranteed by the use of the passwords $\mathsf{pr}$ and $\mathsf{pw}$ that regulate the movements of the read/write taxi ambients. The use of two different passwords ensures that they do not interfere with each other and with other BA ambients' moves, as the latter use $\mathsf{mv}$.

Using the algebraic laws in §1.2 we can show that the encoding is operationally correct in the sense of Lemma 1 for *single-threaded* terms. The single-threadedness hypothesis here, although morally identical to SA's, is adapted to NBA to record that engaging in inter-ambient communications is an activity across ambient boundaries that may as well create grave interferences. For instance, $a[\,\langle x\rangle^{\hat{}} \mid \mathsf{exit}\langle n,k\rangle.P\,]$ cannot be considered single-threaded, as illustrated by, say, the context $\overline{\mathsf{exit}}(x,k).R \mid n[\,(x)^a Q \mid -\,]$. The technical definition is not particularly enlightening, and we omit it in this presentation.

**Theorem 3.** *If $P$ and $Q$ are single-threaded , then $\{\!|\, P \,|\!\}_n \cong \{\!|\, Q \,|\!\}_n$ implies $P \cong Q$.*

The proof follows the same pattern as the one outlined for the encoding of pi-calculus channels. The additional hypothesis on the source terms $P$ and $Q$ is needed to guarantee the atomicity of the protocol that implements an upward exchange (once the taxi ambient leaves $n$, we need to make sure that no process causes $n$ to move).

## 4 Conclusion and Future Work

We presented NBA, a new foundation for the calculus of Boxed Ambients that eliminates grave communication interferences without renouncing to mobility and with no further lost of expressiveness. NBA greatly enhances the distributed nature of BA in the precise sense of removing some obvious challenges to implementation in a distributed scenario. The basic mechanism of NBA is to promote co-capabilities to the role of binding constructs that inform ambients of the incoming ambient's name. Together with a system of password control which verifies the visitor's credentials, this yields an interesting way to learn names dynamically that makes up for most of the expressiveness lost by removing from BA the interference-prone forms of across-boundary communication. Furthermore, the access protocol of NBA has a practical, realistic flavour, at all analogous to the negotiation needed for new computational agents to join existing agent communities, such as networks. We are currently investigating a variant of our calculus where co-capabilities are not binding but they rather perform a renaming of the incoming ambient, as in the Seal calculus [11], giving the recipient full control of the incoming ambients. This variant would be a perfect candidate to compare ambient calculi and Seal. Actually, passwords in NBA plays the same role as channels in Seal: they allow synchronisation for code mobility.

From the theoretical viewpoint, NBA enjoys a rich algebraic theory. Its barbed congruence admits a sound coinductive characterisation built on a LTS semantics. We strongly conjecture that such characterisation is also complete. We presented a type system for NBA whose simplicity and generality relies also on passwords. The mobility types of [8] can be applied in NBA at no additional cost.

Finally, we have characterised the loss of expressiveness by means of several small examples and an encoding of BA to a version of NBA enriched with guarded choice. We plan in future work to investigate encodings without sum, following the lines of, e.g., [10]. We expect only very weak forms of encoding to hold, so as to confirm the present characterisation of the expressiveness gap between BA and NBA.

## References


1. S. Arun-Kumar and M. Hennessy. An Efficiency Preorder for Processes. *Acta Informatica*, 29:737–760, 1992.
2. M. Bugliesi, G. Castagna, and S. Crafa. Boxed Ambients. In *TACS'01 Proc. of the 4th Int. Conference on Theoretical Aspects of Computer Science*, number 2215 in Lecture Notes in Computer Science, pages 38–63. Springer-Verlag, 2001.
3. L. Cardelli and A. Gordon. Mobile Ambients. In *Proceedings of F0SSaCS'98*, number 1378 in Lecture Notes in Computer Science, pages 140–155. Springer, 1998.
4. S. Crafa, M. Bugliesi, and G. Castagna. Information Flow Security for Boxed Ambients. In *F-WAN: Int. Workshop on Foundations of Wide Area Network Computing*, number 66.3 in ENTCS. Elsevier, 2002.
5. K. Honda and N. Yoshida. On Reduction-based Process Semantics. *Theoretical Computer Science*, 152(2):437–486, 1995.
6. F. Levi and D. Sangiorgi. Controlling Interference in Ambients. In *Proceedins of POPL'00*, pages 352–364. ACM Press, 2000.
7. M. Merro and M. Hennessy. Bisimulation Congruences in Safe Ambients. In *POPL'02 Proc. 29th ACM Symposium on Principles of Programming Languages*, pages 71–80. ACM Press, 2002.
8. M. Merro and V. Sassone. Typing and Subtyping Mobility in Boxed Ambients. In *Proceedings of Concur'02*, Lecture Notes in Computer Science. Springer, 2002. To appear.
9. R. Milner, J. Parrow, and D. Walker. A Calculus of Mobile Processes, Parts I and II. *Information and Computation*, 100:1–77, September 1992.
10. B.C. Pierce and U. Nestmann. Decoding Choice Encodings. *Information and Computation*, 163:1–59, 2000.
11. J. Vitek and G. Castagna. Seal: A framework for Secure Mobile Computations. In *Internet Programming Languages*, 1999.