A Control Flow Analysis for Safe and Boxed Ambients ⋆

Francesca Levi[1] Chiara Bodei[2]

[1] DISI, Università di Genova [2] Dpt. di Informatica, Università di Pisa

Abstract. We present two main contributions: (i) an encoding of Boxed Ambients into a variant of Safe Ambients; (ii) a new Control Flow Analysis for Safe Ambients. Then, we show that the analysis, when applied to the encoded processes, permits to accurately verify Mandatory Access Control policies of the source processes.

1 Introduction

Mobile Ambients (MA) [9] is one of the most relevant linguistic answers, in literature, to the problem of modelling mobility. *Ambients* are bounded places, where multi-threaded computations happen, and represent its central notion. They are characterized by a name, and by a collection of local processes and of sub-ambients, therefore generalizing both the idea of agent and the idea of location. The ambients hierarchy can be dynamically modified, according to the three following *capabilities*: $\mathtt{in}\, n$ allows an ambient to enter into an ambient (named) n: $(m[\,\mathtt{in}\, n.\, P_1 \mid P_2\,] \mid n[\,Q\,] \longrightarrow n[\,m[\,P_1 \mid P_2\,] \mid Q\,])$; $\mathtt{out}\, n$ allows an ambient to exit from an ambient (named) n: $(n[\,m[\,\mathtt{out}\, n.\, P_1 \mid P_2\,] \mid Q\,] \longrightarrow m[\,P_1 \mid P_2\,] \mid n[\,Q\,])$; $\mathtt{open}\, n$ allows to destroy the boundary of an ambient (named) n: $(\mathtt{open}\, n.\, P \mid n[\,Q\,] \longrightarrow P \mid Q)$. The previous rules show that the affected ambient n undergoes the action and has no control on whether or not the action takes place. This is a serious drawback which is overcome in *Safe Ambients* (SA) [20, 21], a variant of MA, where a movement or an ambient dissolution can take place only when the affected ambient, named n, allows it, offering the corresponding *coaction*. This modification does not change the expressiveness of the calculus, yet makes it easier both to write programs and to formally prove their correctness, by using behavioural equivalences [20, 22]. Furthermore, the existing static techniques of MA, based on Type Systems [7, 8] and Control Flow Analysis (CFA) [24], can straightforwardly transplanted to SA, and typically give more precise results due to the coactions [20, 12, 11, 3, 15, 1, 17].

Recently, another modification of MA has been proposed, called *Boxed Ambients* (BA). In BA there is no ambient dissolution and new primitives are introduced for exchanging values across ambient boundaries (in MA/SA only processes local to the same ambient can communicate). Specifically, the *downward* actions, $\langle M \rangle^n$ and $(x)^n\, P$, indicate an output for and an input from a

⋆ Work partially supported by EU-project DEGAS (IST-2001-32072)

sub-ambient (named) n; and the *upward* actions, $\langle M\rangle^{\uparrow}$ and $(x)^{\uparrow}\, P$, indicate an output for and an input from the parent ambient.

To understand the relevant difference between the BA and the MA/SA model it is convenient to consider a simple situation. Assume that an ambient (named) a is willing to communicate a message to an ambient (named) b. The following MA and BA (using the semantics of [10]) processes show two typical protocols,

$$a[\,\mathtt{in}\,b.\,m[\,\mathtt{out}\,a.\,\langle M\rangle\,]\mid P\,]\mid b[\,\mathtt{open}\,m\mid(x)\,S\mid Q\,] \tag{1}$$

$$a[\,\mathtt{in}\,b.\,\langle M\rangle^{\uparrow}\mid P\,]\mid b[\,(x)^{a}\,S\mid Q\,] \tag{2}$$

In (1) ambient a moves inside ambient b; then an auxiliary ambient (named) m exits from ambient a, thus ending up within ambient b, where it is opened; consequently, the input $(x)\ S$ and the output $\langle M\rangle$ can locally interact. The role of the `open` capability here is crucial, but at the same time may not give security guarantees. In fact, any process contained within ambient m is unleashed inside ambient b; ambient m could for instance contain a malicious process, which may cause ambient b to move inside possibly hostile locations and to be there damaged. Further problems arise when we try to understand the opening, according to the principles of mandatory access control (MAC) policies in multi-level security. As the ambient interested by the action is dissolved, it is rather difficult to classify standard read and write operations. In (2) instead there is no need of the auxiliary ambient m, because ambient a enters inside ambient b, and can directly communicate. Moreover, the MAC properties have a natural interpretation: there is clearly a write access from ambient a to ambient b, and symmetrically a read access from ambient b to ambient a.

As this example shows, the new forms of inter-level communication offer a valid alternative to the delicate primitive of opening of MA/SA. Nevertheless, the introduction of a different model for a given language – as BA w.r.t. MA/SA - suggests the following questions: (i) is it possible to rebuild for the former model (BA) the techniques previously developed for the latter (MA/SA)? (ii) does it exist an encoding from the former into the latter model? and, consequently, (iii) which of the techniques for the latter model keep giving the expected results when applied to the encoded processes?

The first issue has been investigated: [4, 23] introduce type systems for BA, inspired from those of MA, controlling the consistency of communications and the mobility of ambients; [5] extends [4] for checking precisely the MAC properties. In this paper instead we address the other approach and we investigate whether it is possible to apply the same static techniques for both languages. We present the following contributions: – an encoding of BA (w.r.t. the semantics of [10]) into a variant of SA, in Sect. 3; – a new Control Flow Analysis for SA, in Sect. 4, which is an adaptation of the CFA of [24] for MA, obtained by profitably exploiting the presence of coactions. It computes a sound approximation of the run-time behaviour of processes, by giving information about the evolution of the ambients hierarchy, and about which capabilities may be exercised and which messages may be exchanged inside any ambient.

We finally show that our analysis can safely be applied to the encoded BA processes to verify the MAC policies, and that it is particularly adequate for this application. First, it can be more precise than the types for BA of [5] [1]. Moreover, when compared to the other static techniques for MA/SA, our analysis is a good compromise between complexity and accuracy.

Some formal statements, here omitted for lack of space, can be found in [18].

2 Syntax and Semantics of SA and BA

With the aim of compacting as much as possible the presentation, we give the common productions and rules, needed both for (the variant) of SA and for BA, and then the additional productions and semantic rules specific of any calculus.

We note that the variant of SA is a combination of recent proposals [14, 15, 22, 23, 6] where: the coaction for opening does not refer to the name of the affected ambient; the coactions for movements reside in the target ambient, and either refer to the name of the ambient authorized to move or allow any ambient to move. These modifications are necessary only for the encoding, meaning that the CFA can be easily adapted to standard SA.

Syntax. As common in CFA, we discipline the α-renaming of bound names and variables by partitioning names and variables as follows. For SA, we consider names $\widehat{\mathcal{N}} = \uplus_{n \in \mathcal{N}} \mathcal{N}_n$, where $\mathcal{N}_n = \{n_0, n_1, \ldots\}$ for the infinite set of names $\mathcal{N} = \{n, h, k, \ldots\}$; similarly for variables, given the set $\mathcal{V} = \{x, y, z \ldots\}$ and $\widehat{\mathcal{V}} = \uplus_{x \in \mathcal{V}} \mathcal{V}_x$. Analogously, for BA we consider names $\widehat{\mathcal{N}_{BA}}$ and variables $\widehat{\mathcal{V}_{BA}}$ partitioned as in SA. Hereafter, we may use the name n and the variable x for a generic element of $\mathcal{N}_n$ and demand that a name $n_i \in \mathcal{N}_n$ can only be α-converted with a name $n_j \in \mathcal{N}_n$. Moreover, to define the CFA it is convenient to label the SA ambients, using a set of labels $\mathcal{L}$ (ranged over by elements λ, μ ...), where $\top \in \mathcal{L}$ represents the outermost ambient.

The following productions define the syntax of SA and BA, processes and expressions (for simplicity we omit capability paths),

Common Part

$P ::=$	*Processes*	$M ::=$	*Expressions*
$\mathbf{0}$	nil	$\mathtt{in}\, M$	enter M
$\mid\ M.P$	prefix	$\mid\ \mathtt{out}\, M$	exit M
$\mid\ P \mid P$	parallel	$\mid\ n$	name
$\mid\ (\nu n)P$	restriction	$\mid\ x$	variable
$\mid\ !P$	replication	$\mid\ (M_1, \ldots, M_n)$	tuple
$\mid\ \langle M \rangle$	message		
$\mid\ (x_1, \ldots, x_n)\, P$	abstraction		

Additional Productions for SA

$P ::=$	*Processes*	$M ::=$	*Expressions*
$M_\lambda[P]$	ambient	$\overline{\mathtt{in}}\, M$	let M enter
		$\mid\ \overline{\mathtt{out}}\, M$	let M exit
		$\mid\ \mathtt{open}\, M$	open M
		$\mid\ \overline{\mathtt{open}}$	let open
		$\mid\ \overline{\mathtt{in}}$	let enter
		$\mid\ \overline{\mathtt{out}}$	let exit

[1] The type system is defined for a slightly different semantics of BA [4]. In this paper we refer to its easy adaptation to the semantics considered here (see [10]).

Additional Productions for BA

$P ::=$		*Processes*	$\eta ::= M \mid \uparrow$
	$M[\,P\,]$	ambient	
	$\mid \ \langle M \rangle^{\eta}$	message up-down	
	$\mid \ (x_1,\ldots,x_n)^{\eta} P$	abstraction up-down	

For both calculi, we adopt standard syntactical conventions. We often omit the trailing $\mathbf{0}$ in processes, and we assume that parallel composition has the least syntactic precedence. As usual, the operators $(\nu n)P$, $(x_1,\ldots,x_n)P$ and $(x_1,\ldots,x_n)^{\eta}P$ act as static binders for name n and for variables $x_1,\ldots,x_n$; the free and bound names of processes and expressions are defined, accordingly. Moreover, we use the standard notation $P\{^{M}\!/\!_{x}\}$ for substitution. We use $\tilde{M}$ to denote tuples of expressions $(M_1,\ldots,M_k)$, and we assume that $(\nu\tilde{n})$ stands for $(\nu n_1)\ldots(\nu n_k)$ and $P\{^{\tilde{M}}\!/\!_{\tilde{x}}\}$ for $P\{^{M_1}\!/\!_{x_1}\}\ldots\{^{M_k}\!/\!_{x_k}\}$. We recall also that in the untyped calculi bad-formed processes may occur, such as $\mathtt{in}\, n[\,P\,]$ and $n.\,P$.

Common part $(\mathtt{Com})$ $\langle \tilde{M} \rangle \mid (\tilde{x})\, P \longrightarrow P\{^{\tilde{M}}\!/\!_{\tilde{x}}\}$

Additional axioms of SA

$(\mathtt{In})$ for $\mathtt{cap} \in \{\overline{\mathtt{in}}\,, \overline{\mathtt{in}}\, n\}$

$$n_\lambda[\,\mathtt{in}\, m.\, P_1 \mid P_2\,] \mid m_\mu[\,\mathtt{cap}.\, Q_1 \mid Q_2\,] \longrightarrow m_\mu[\,n_\lambda[\,P_1 \mid P_2\,] \mid Q_1 \mid Q_2\,]$$

$(\mathtt{Out})$ for $\mathtt{cap} \in \{\overline{\mathtt{out}}\,, \overline{\mathtt{out}}\, n\}$

$$m_\mu[\,n_\lambda[\,\mathtt{out}\, m.\, P_1 \mid P_2\,] \mid Q_2\,] \mid \mathtt{cap}.\, Q_1 \longrightarrow n_\lambda[\,P_1 \mid P_2\,] \mid m_\mu[\,Q_2\,] \mid Q_1$$

$(\mathtt{Open})$ $\mathtt{open}\, n.\, Q \mid n_\lambda[\,\overline{\mathtt{open}}\,.\, P_1 \mid P_2\,] \longrightarrow Q \mid P_1 \mid P_2$

Additional axioms of BA

$(\mathtt{Input}\ \uparrow)$ $\langle \tilde{M} \rangle^{n} \mid n[\,(\tilde{x})^{\uparrow}\, P \mid R\,] \longrightarrow n[\,P\{^{\tilde{M}}\!/\!_{\tilde{x}}\} \mid R\,]$

$(\mathtt{Output}\ \uparrow)$ $(\tilde{x})^{n}\, P \mid n[\,\langle \tilde{M} \rangle^{\uparrow} \mid R\,] \longrightarrow P\{^{\tilde{M}}\!/\!_{\tilde{x}}\} \mid n[\,R\,]$

Table 1. Reduction Axioms of BA and SA

Semantics. For lack of space, we comment only the top-level reductions (the main ones are reported in Tab. 1); the auxiliary relation of structural congruence $\equiv$ and the inference rules, which propagate the reductions, are fairly standard (for more details [9, 4]). The rules $(\mathtt{In})$, $(\mathtt{Out})$ and $(\mathtt{Open})$ of SA model the movements, in and out, and the opening of ambients. The rules of movement for BA are analogous to those for MA, outlined in the Introduction. Rule $(\mathtt{Com})$, common to both languages, models local communication, and the additional rules for BA $(\mathtt{Output}\ \uparrow)$ and $(\mathtt{Input}\ \uparrow)$ model the inter-level communications.

In the following, $\Longrightarrow$ stands for the transitive and reflexive closure of $\longrightarrow$. Moreover, to simplify the encoding and the CFA, we consider SA and BA processes where all the bound names and variables are distinct one from each other and from the free names and variables, respectively.

3 Encoding Boxed Ambients into Safe Ambients

We define the encoding of BA into SA in two steps: we give the translation into unlabeled processes; then we introduce *a posteriori* a suitable annotation [2].

The encoding, defined in Tab. 2, depends on the enclosing ambient (in general an expression N) and works compositionally on the structure of the process. The inter-level communications are simulated by processes which use auxiliary sets of names $\mathcal{N}_{aux}$ and variables $\mathcal{V}_{aux}$. We define $\mathcal{N}_{aux} = \mathcal{N}_{\mathcal{G}^{\uparrow,\downarrow}} \cup \mathcal{N}_e$, where $\mathcal{G}^{\uparrow,\downarrow} = \{\mathtt{r}_1^{\downarrow}, \mathtt{r}_2^{\downarrow}, \mathtt{w}^{\uparrow}, \mathtt{w}_1^{\downarrow}, \mathtt{w}_2^{\downarrow}, \mathtt{r}^{\uparrow}\}$, $\mathcal{N}_{\mathcal{G}^{\uparrow,\downarrow}} = \{n^g \mid n \in \mathcal{N}_{BA}, g \in \mathcal{G}^{\uparrow,\downarrow}\}$ and $\mathcal{N}_e = \{\mathtt{t}^{\mathtt{rd}}, \mathtt{t}^{\mathtt{wd}}, \mathtt{c}^{\mathtt{ru}}, \mathtt{c}^{\mathtt{rd}}\}$; analogously, we define $\mathcal{V}_{aux} = \mathcal{V}_{\mathcal{G}^{\uparrow,\downarrow}}$.

Expressions

$$\llbracket M \rrbracket = (M, M^{\mathtt{r}_1^{\downarrow}}, M^{\mathtt{r}_2^{\downarrow}}, M^{\mathtt{w}^{\uparrow}}, M^{\mathtt{w}_1^{\downarrow}}, M^{\mathtt{w}_2^{\downarrow}}, M^{\mathtt{r}^{\uparrow}})$$

$$(M_i)^g = (M^g)_i \text{ if } M \in \mathcal{V}_{BA} \cup \mathcal{N}_{BA} \qquad (\mathtt{cap}\ M)^g = \mathtt{cap}\ M^g \text{ if } \mathtt{cap} \in \{\mathtt{in}, \mathtt{out}\}$$

Processes

$$\llbracket \mathbf{0} \rrbracket^N = \mathbf{0} \quad \llbracket M.\,P \rrbracket^N = M.\,\llbracket P \rrbracket^N \quad \llbracket P \mid Q \rrbracket^N = \llbracket P \rrbracket^N \mid \llbracket Q \rrbracket^N \quad \llbracket !P \rrbracket^N = !\llbracket P \rrbracket^N$$

$$\llbracket (\nu n)\ P \rrbracket^N = (\nu \llbracket n \rrbracket)\ \llbracket P \rrbracket^N \quad \llbracket M[\,P\,] \rrbracket^N = M[\,\llbracket P \rrbracket^M \mid !\overline{\mathtt{out}} \mid !\overline{\mathtt{in}}\,] \mid !\overline{\mathtt{out}}$$

$$\llbracket \langle M \rangle^{\uparrow} \rrbracket^N = \mathtt{wu}(\llbracket M \rrbracket, N, N^{\mathtt{r}_1^{\downarrow}}, N^{\mathtt{r}_2^{\downarrow}}, N^{\mathtt{w}^{\uparrow}}) \quad \llbracket (\tilde{x})^{N'}\ P \rrbracket^N = \mathtt{rd}((\llbracket \tilde{x} \rrbracket)\ \llbracket P \rrbracket^N, N'^{\mathtt{r}_1^{\downarrow}}, N'^{\mathtt{r}_2^{\downarrow}}, N'^{\mathtt{w}^{\uparrow}})$$

$$\llbracket (\tilde{x})^{\uparrow}\ P \rrbracket^N = \mathtt{ru}((\llbracket \tilde{x} \rrbracket)\ \llbracket P \rrbracket^N, N, N^{\mathtt{w}_1^{\downarrow}}, N^{\mathtt{w}_2^{\downarrow}}, N^{\mathtt{r}^{\uparrow}}) \quad \llbracket (\tilde{x})\ P \rrbracket^N = (\llbracket \tilde{x} \rrbracket)\ \llbracket P \rrbracket^N$$

$$\llbracket \langle M \rangle^{N'} \rrbracket^N = \mathtt{wd}(\langle \llbracket M \rrbracket \rangle, N'^{\mathtt{w}_1^{\downarrow}}, N'^{\mathtt{w}_2^{\downarrow}}, N'^{\mathtt{r}^{\uparrow}}) \qquad \llbracket \langle M \rangle \rrbracket^N = \langle \llbracket M \rrbracket \rangle$$

Table 2. The encoding.

Names and variables are translated into the corresponding tuple in $\mathcal{N}_{\mathcal{G}^{\uparrow,\downarrow}}$ and $\mathcal{V}_{\mathcal{G}^{\uparrow,\downarrow}}$; the other expressions are translated accordingly. The encoding of prefix, parallel composition, bang, restriction and local communications are fairly standard. An ambient is translated into a corresponding one, with the same name; generic coactions $\overline{\mathtt{in}}$ and $\overline{\mathtt{out}}$ are introduced to properly preserve the feature that a BA ambient can be traversed by any other ambient. More interesting and difficult is the encoding of inter-level communications, rendered by the four basic protocols, defined in Tab. 3, using the auxiliary ambients. More in details, *upward* and *downward* messages and abstractions, located inside an ambient N, are modelled as follows: – $\mathtt{wu}(\langle \llbracket M \rrbracket \rangle, N, N^{\mathtt{r}_1^{\downarrow}}, N^{\mathtt{r}_2^{\downarrow}}, N^{\mathtt{w}^{\uparrow}})$ and $\mathtt{ru}((\llbracket x \rrbracket)\ \llbracket P \rrbracket^N, N, N^{\mathtt{w}_1^{\downarrow}}, N^{\mathtt{w}_2^{\downarrow}}, N^{\mathtt{r}^{\uparrow}})$ realise $\langle M \rangle^{\uparrow}$ and $(x)^{\uparrow}\ P$, respectively; – and, $\mathtt{wd}(\langle \llbracket M \rrbracket \rangle, N'^{\mathtt{w}_1^{\downarrow}}, N'^{\mathtt{w}_2^{\downarrow}}, N'^{\mathtt{r}^{\uparrow}})$ and $\mathtt{rd}((\llbracket x \rrbracket)\ \llbracket P \rrbracket^N, N'^{\mathtt{r}_1^{\downarrow}}, N'^{\mathtt{r}_2^{\downarrow}}, N'^{\mathtt{w}^{\uparrow}})$ realise $\langle M \rangle^{N'}$ and $(x)^{N'}\ P$ for sub-ambients (named) N', respectively.

We now explain how the protocols of Tab. 3 suitably model the inter-level communications corresponding to ($\mathtt{Output}\ \uparrow$) and ($\mathtt{Input}\ \uparrow$). We mean that their execution lead to the expected communication and also that their steps cannot be interfered by the interactions with the external context.

[2] Notice that the labels are used only by the CFA and do not affect at all the semantics.

$$
\begin{aligned}
&\mathtt{wu}(\langle M\rangle, N, N^{\mathtt{r}_1^{\downarrow}}, N^{\mathtt{r}_2^{\downarrow}}, N^{\mathtt{w}^{\uparrow}}) = \mathsf{in}\, N^{\mathtt{r}_1^{\downarrow}} .\, \mathsf{in}\, N^{\mathtt{r}_2^{\downarrow}} .\, (N^{\mathtt{w}^{\uparrow}}[\, \mathsf{out}\, N .\, \overline{\mathtt{open}} .\, \langle M\rangle \,] \mid \mathsf{out}\, N^{\mathtt{r}_2^{\downarrow}})\\
&\mathtt{rd}((\tilde{x})\, P, N^{\mathtt{r}_1^{\downarrow}}, N^{\mathtt{r}_2^{\downarrow}}, N^{\mathtt{w}^{\uparrow}}) = (\nu \mathtt{t}^{\mathtt{rd}})\, (\nu \mathtt{c}^{\mathtt{rd}}) \Big(\mathtt{t}^{\mathtt{rd}}[\, \overline{\mathtt{in}}\, N^{\mathtt{r}_1^{\downarrow}} .\, \mathsf{open}\, N^{\mathtt{r}_1^{\downarrow}} .\, \overline{\mathtt{open}} \,] \mid\\
&\mathsf{open}\, \mathtt{t}^{\mathtt{rd}} .\, \mathsf{open}\, \mathtt{c}^{\mathtt{rd}} \mid N^{\mathtt{r}_1^{\downarrow}} [\overline{\mathtt{in}} .\, (N^{\mathtt{r}_2^{\downarrow}}[\, \overline{\mathtt{in}} .\, \overline{\mathtt{out}}\, N^{\mathtt{w}^{\uparrow}} .\, \mathsf{open}\, N^{\mathtt{w}^{\uparrow}} .\, (\tilde{x})\, \mathtt{c}^{\mathtt{rd}}[\, \mathsf{out}\, N^{\mathtt{r}_2^{\downarrow}} .\, \overline{\mathtt{open}} .\, P \,]] \mid\\
&\qquad\qquad\qquad \overline{\mathtt{out}}\, \mathtt{c}^{\mathtt{rd}} .\, \overline{\mathtt{out}} .\, \mathsf{in}\, \mathtt{t}^{\mathtt{rd}} .\, \overline{\mathtt{open}} \,)] \Big)\\
&\mathtt{ru}((\tilde{x})\, P, N, N^{\mathtt{w}_1^{\downarrow}}, N^{\mathtt{w}_2^{\downarrow}}, N^{\mathtt{r}^{\uparrow}}) = (\nu \mathtt{c}^{\mathtt{ru}})\\
&\Big(\mathsf{in}\, N^{\mathtt{w}_1^{\downarrow}} .\, \mathsf{in}\, N^{\mathtt{w}_2^{\downarrow}} .\, (N^{\mathtt{r}^{\uparrow}}[\mathsf{out}\, N .\, \overline{\mathtt{open}} .\, (\tilde{x})\, \mathtt{c}^{\mathtt{ru}}[\, \mathsf{in}\, N .\, \overline{\mathtt{open}} .\, P \,]] \mid \mathsf{open}\, \mathtt{c}^{\mathtt{ru}} .\, \mathsf{out}\, N^{\mathtt{w}_2^{\downarrow}}) \Big)\\
&\mathtt{wd}(\langle M\rangle, N^{\mathtt{w}_1^{\downarrow}}, N^{\mathtt{w}_2^{\downarrow}}, N^{\mathtt{r}^{\uparrow}}) = (\nu \mathtt{t}^{\mathtt{wd}}) \Big(\mathtt{t}^{\mathtt{wd}}[\, \overline{\mathtt{in}}\, N^{\mathtt{w}_1^{\downarrow}} .\, \mathsf{open}\, N^{\mathtt{w}_1^{\downarrow}} .\, \overline{\mathtt{open}} \,] \mid \mathsf{open}\, \mathtt{t}^{\mathtt{wd}} \mid\\
&\qquad\qquad N^{\mathtt{w}_1^{\downarrow}} [\overline{\mathtt{in}} .\, (N^{\mathtt{w}_2^{\downarrow}}[\, \overline{\mathtt{in}} .\, \overline{\mathtt{out}}\, N^{\mathtt{r}^{\uparrow}} .\, \mathsf{open}\, N^{\mathtt{r}^{\uparrow}} .\, \langle M\rangle \,] \mid \mathsf{in}\, \mathtt{t}^{\mathtt{wd}} .\, \overline{\mathtt{out}} .\, \overline{\mathtt{open}} \,)] \Big)
\end{aligned}
$$

Table 3. Protocols for inter-level communication

To illustrate (`Output` $\uparrow$) we consider a process $m[\, n[\, \langle M\rangle^{\uparrow}\,] \mid (x)^n\, P\,]$. The communication is realised by the processes $\mathtt{rd}((\llbracket x\rrbracket)\, \llbracket P\rrbracket^m, n^{\mathtt{r}_1^{\downarrow}}, n^{\mathtt{r}_2^{\downarrow}}, n^{\mathtt{w}^{\uparrow}})$, running inside m, and $\mathtt{wu}(\langle\llbracket M\rrbracket\rangle, n, n^{\mathtt{r}_1^{\downarrow}}, n^{\mathtt{r}_2^{\downarrow}}, n^{\mathtt{w}^{\uparrow}})$, running inside the sub-ambient n. Ambient $n^{\mathtt{r}_1^{\downarrow}}$, located within m, is a sort of isolating box, which protects the interaction between the ambients $n^{\mathtt{w}^{\uparrow}}$, containing the output $\langle\llbracket M\rrbracket\rangle$, and $n^{\mathtt{r}_2^{\downarrow}}$, containing the input $(\llbracket x\rrbracket)\, \llbracket P\rrbracket^m$. The protocol is started by the sub-ambient n that moves inside $n^{\mathtt{r}_1^{\downarrow}}$, and then inside $n^{\mathtt{r}_2^{\downarrow}}$. At this point, ambient $n^{\mathtt{w}^{\uparrow}}$ goes out of n, and therefore ends up inside $n^{\mathtt{r}_2^{\downarrow}}$, where it is opened; consequently, the message $\langle\llbracket M\rrbracket\rangle$ is unleashed and can be consumed. After the communication has been realised, both the continuation of the abstraction (contained inside $\mathtt{c}^{\mathtt{rd}}$) and ambient n go out from ambient $n^{\mathtt{r}_2^{\downarrow}}$. Finally, the boundary of $n^{\mathtt{r}_1^{\downarrow}}$ is dissolved, thus liberating the ambients n and $\mathtt{c}^{\mathtt{rd}}$ within ambient m. To avoid the observation of the opening of $n^{\mathtt{r}_1^{\downarrow}}$ from the external context, its name is changed into a fresh name $\mathtt{t}^{\mathtt{rd}}$ by using a well-known renaming protocol [9].

The interaction (`Input` $\uparrow$) is realised by a similar protocol using the symmetric processes and auxiliary ambients. Only slight modifications are necessary, since here the continuation of the abstraction (that is $\mathtt{c}^{\mathtt{ru}}$) has to end up inside the sub-ambient rather than inside the enclosing ambient.

To annotate the BA and auxiliary ambients, resp., we adopt sets of labels $\mathcal{L}_{BA}$ and $\mathcal{L}_{aux}$, where $\mathcal{L}_{aux} = \{(g_\lambda, i) \mid g \in \mathcal{G}^{\uparrow,\downarrow} \cup \mathcal{N}_e,\ \lambda \in \mathcal{L}_{BA}, i \in N\}$. Formally, we require that, for any ambient $M_\chi[\, Q\,]$, occurring in the labelled process: (i) if M is a BA expression, then $\chi \in \mathcal{L}_{BA}$; (ii) if M is either a name $g \in \mathcal{N}_e$ or an expression N^g for $g \in \mathcal{G}^{\uparrow,\downarrow}$, then χ is (g_λ, i), where λ is the label of the surrounding BA ambient (possibly $\top$).

Notice that, by condition (ii), the labels of the auxiliary ambients (used in Tab. 3) depend on the label of the BA ambient, which is ready to engage into the inter-level communication action (see e. g. Sect. 5). Moreover, different equivalent labellings can be given, we therefore use $\llbracket P\rrbracket^N$ for the canonical representative. As common in CFA, in order to obtain a more precise analysis it is convenient to assign distinct labels to the ambients.

The encoding is operationally correct, meaning that any reduction of a BA process is properly simulated by its SA encoding, and that the steps of the encoding simulating any BA reduction cannot be interfered. We omit here the formalization of this property, as it requires to introduce additional notions (including behavioural equivalence), and we refer the reader to [18]. We only mention that coactions are necessary to control the interferences, which may damage the protocols, similarly as in the coalescing encoding of π-calculus [20].

4 Control Flow Analysis for *SA*

The analysis approximates the run-time behaviour of a process, by predicting for each ambient, its possible sub-ambients, and which capabilities may be exercised and which messages may be exchanged locally. Our CFA does not distinguish among different elements of the same equivalence class $\mathcal{N}_n$ and $\mathcal{V}_x$: it only considers their canonical representatives n and x, respectively.

Solutions. A *local solution* I is either on the form $\langle L, U, C\rangle$ or on the form $\langle L, U \cup \{\overline{\mathtt{open}}\}, C\rangle; \langle L', U', C'\rangle$, for $L, L' \subseteq \mathcal{L}$, $C, C' \subseteq \mathcal{CE}$ and $U, U' \subseteq \mathcal{L} \cup \mathcal{CE}$, where $\mathcal{CE}$ is the set of closed SA expressions. A *solution* of the analysis is a triple (ρ, ϕ, σ), where : $\rho : \mathcal{L} \to \mathcal{CE}$, describes the relation between an ambient label and its possible names; $\phi : \mathcal{L} \to \mathcal{I}$ assigns a local solution to any ambient label (here $\mathcal{I}$ is the set of local solutions); and $\sigma : \mathcal{V} \to \mathcal{CE}$, predicts which expressions may be bound to any variable.

The local solution $\phi(\lambda)$ reports the information about the ambients with label $\lambda \in L$; more in details, it describes the possible behaviour both of the processes running inside, and of the processes which may be unleashed, when they are opened. It may be either (a) $\langle L, U, C\rangle$ or (b) $\langle L, U, C\rangle; \langle L', U', C'\rangle$, provided that $\lambda \in L$. In both cases, the sets U and C approximate the behaviour of the internal processes: U reporting the labels of the possible sub-ambients and the expressions (in particular the capabilities), which may be exercised; C reporting the expressions which may be communicated. Local solutions (a) and (b) differ in the information they give about the opening: (a) says that the ambients may be opened (if $\overline{\mathtt{open}} \in U$) inside the ambients with label $\mu \in L$, and that the process unleashed is described by U and C; while (b) says that the ambients may be opened (indeed $\overline{\mathtt{open}} \in U$) within the ambients with label $\mu \in L'$, and that the process unleashed is described by U' and C'.

Technically, to distinguish, inside a local solution I, between what happens before and after the possible opening, we use the following functions $\mathtt{pre}(I)$ and $\mathtt{post}(I)$. We let $\mathtt{pre}(\langle L, U, C\rangle) = \mathtt{pre}(\langle L, U, C\rangle; \langle L', U', C'\rangle) = \langle L, U, C\rangle$, $\mathtt{post}(\langle L, U, C\rangle; \langle L', U', C'\rangle) = \langle L', U', C'\rangle$ and $\mathtt{post}(\langle L, U, C\rangle) = \langle L, U, C\rangle$, if $\overline{\mathtt{open}} \in U$, $\mathtt{post}(\langle L, U, C\rangle) = \emptyset$, otherwise.

The following example is intended to illustrate both the different meaning of the two forms of local solutions and the role of the coaction $\overline{\mathtt{open}}$, which is exploited, in our CFA, to have a more precise treatment of the critical opening action. This is the main relevant difference with the analysis of [24] for MA, and follows the lines of the type systems for SA [20, 12, 15, 1, 17], in particular the proposal of [15] adding more flexibility to the single-threaded types of [20].

Example 1. The following process is similar to the process (1) of the Introduction: $a_\lambda[\,\mathtt{in}\,b.\,m_\mu[\,(\mathtt{out}\,a.\,\overline{\mathtt{open}}\,.\,\langle\mathtt{in}\,h\rangle) \mid \mathtt{out}\,k\,]\,] \mid b_\chi[\,\overline{\mathtt{in}}\,a.\,\overline{\mathtt{out}}\,m.\,\mathtt{open}\,m.\,(x)\,x\,]$. The only difference is that, when ambient m is opened inside ambient b, both the message $\langle\mathtt{in}\,h\rangle$ and the capability $\mathtt{out}\,k$ are unleashed. A valid solution for this process is (ρ, ϕ, σ), where $\rho(\lambda) = a,\ \rho(\mu) = m,\ \rho(\chi) = b,\ \sigma(x) = \{\mathtt{in}\,h\}$,

$\phi(\top) = \langle\{\top\}, \{\lambda, \mu, \chi\}, \emptyset\rangle \quad \phi(\chi) = \langle\{\chi\}, \{\overline{\mathtt{in}}\,a, \overline{\mathtt{out}}\,m, \mathtt{open}\,m, \mathtt{in}\,h, \mathtt{out}\,k, \mu, \lambda\}, \{\mathtt{in}\,h\}\rangle$
$\phi(\mu) = \langle\{\mu\}, \{\mathtt{out}\,a, \overline{\mathtt{open}}\,, \mathtt{out}\,k\}, \emptyset\rangle; \langle\{\chi\}, \{\mathtt{out}\,k\}, \{\mathtt{in}\,h\}\rangle \quad \phi(\lambda) = \langle\{\lambda\}, \{\mu, \mathtt{in}\,b\}, \emptyset\rangle$

Here, $\rho(\lambda) = a$ says that a may be the name of the ambient with label λ (similarly $\rho(\mu) = m, \ldots$); $\sigma(x)$ contains the expression $\mathtt{in}\,h$, meaning that variable x may be bound to $\mathtt{in}\,h$; the function ϕ describes the possible contents of any ambient. For instance,

– $\phi(\mu)$ says that the capabilities $\mathtt{out}\,a$, $\overline{\mathtt{open}}$ and $\mathtt{out}\,k$ may run inside m; moreover, ambient m may be opened inside ambient b, and, when opened, it may unleash the capability $\mathtt{out}\,k$ and the message $\langle\mathtt{in}\,h\rangle$.
– $\phi(\chi)$ says that inside ambient b: the ambients m and a may appear; the capabilities $\overline{\mathtt{in}}\,a, \overline{\mathtt{out}}\,m, \mathtt{open}\,m, \mathtt{in}\,h$ and $\mathtt{out}\,k$ may be exercised; and the expression $\mathtt{in}\,h$ may be communicated.

Note that in the local solution for ambient m (that is $\phi(\mu)$) there is a clear distinction between what may end up *before* and *after* the capability $\overline{\mathtt{open}}$. This information is exploited by the CFA to predict that ambient b, when opening ambient m, acquires *only* the capability $\mathtt{out}\,k$ and the message $\langle\mathtt{in}\,h\rangle$ (see $\phi(\chi)$). We could also adopt a local solution (as in the CFA of [24]) $\phi(\mu) = \langle\{\mu, \chi\}, \{\mathtt{out}\,a, \overline{\mathtt{open}}\,, \mathtt{out}\,k\}, \{\mathtt{in}\,h\}\rangle$. In this case, there is no distinction between what may run inside the ambient and what may be unleashed, when the ambient is opened (inside m or b). This would therefore lead to a less precise result. In particular, $\phi(\chi)$ should be modified taking into account that also the capabilities $\mathtt{out}\,a$ and $\overline{\mathtt{open}}$ may end up within ambient b, as a consequence to the opening of ambient m.

Validation. The validation of a given solution, as we will see in a while, is established by a set of clauses, that bring the validation of a compound process back to the validations of its sub-processes. To capture the distinction between what happens before and after the opening, it is necessary to keep the local solutions of the sub-processes and to combine them in a proper way. This is not a trivial task and can be intuitively introduced by considering Ex. 1 and the validation of the internal process of ambient m, w.r.t. (ρ, ϕ, σ) and $\phi(\mu)$.

The local solution $I_1 = \langle\{\mu\}, \{\mathtt{out}\,a, \overline{\mathtt{open}}\,\}, \emptyset\rangle; \langle\{\chi\}, \emptyset, \{\mathtt{in}\,h\}\rangle$ approximates the behaviour of $\mathtt{out}\,a.\,\overline{\mathtt{open}}\,.\,\langle\mathtt{in}\,h\rangle$, and is obtained by combining the following local solutions: $\langle\{\mu\}, \{\mathtt{out}\,a\}, \emptyset\rangle$ describing the presence of $\mathtt{out}\,a$ inside m; $\langle\{\mu\}, \{\overline{\mathtt{open}}\,\}, \emptyset\rangle; \langle\{\chi\}, \emptyset, \emptyset\rangle$ saying that ambient m may be opened inside b; $\langle\{\chi\}, \emptyset, \{\mathtt{in}\,h\}\rangle$ describing the presence of message $\langle\mathtt{in}\,h\rangle$ inside b. The position of the processes establishes the ordering of composition and therefore what may end up before $\overline{\texttt{open}}$ (that is inside m) and after $\overline{\texttt{open}}$ (that is inside the ambient opening m, named b).

Furthermore, $\phi(\mu) = \langle\{\mu\}, \{\texttt{out}\, a, \overline{\texttt{open}}, \texttt{out}\, k\}, \emptyset\rangle; \langle\{\chi\}, \{\texttt{out}\, k\}, \{\texttt{in}\, h\}\rangle$ approximates the behaviour of process $\texttt{out}\, a. \overline{\texttt{open}}. \langle \texttt{in}\, h\rangle \mid \texttt{out}\, k$, and is obtained by combining I_1 with the local solution $I_2 = \langle\{\chi, \mu\}, \{\texttt{out}\, k\}, \emptyset\rangle$ describing the process $\texttt{out}\, k$. Note that, since the capability $\texttt{out}\, k$ may be executed (at run-time) either before (inside m) or after opening (inside b), then I_2 correctly predicts that it may be enclosed in both ambients and $\phi(\mu)$ says that it may end up both before and after opening. We also note that the solution (ρ, ϕ, σ) is valid for this process w.r.t. $\phi(\mu)$, provided that the possible effects of the capabilities of movement on their enclosing ambients are properly recorded in the ambients hierarchy. For instance, $\texttt{out}\, a$ could cause the ambient m to move out of ambient a (see the discussion below about rule (**move**)).

To realise the technique explained above, following the type systems for SA [15, 1, 17], we introduce two main technical concepts. First, we define two operators of *parallel and sequential* composition $(|)$ and $(.)$ to properly combine local solutions. We order components by letting $\langle L_1, U_1, C_1\rangle \sqsubseteq \langle L_2, U_2, C_2\rangle$ iff $L_2 \subseteq L_1$, $U_1 \subseteq U_2$ and $C_1 \subseteq C_2$, and we define

$$
\begin{array}{l}
\langle L_1, U_1, C_1\rangle \sqcup \langle L_2, U_2, C_2\rangle = \langle L_1, U_1, C_1\rangle \mid \langle L_2, U_2, C_2\rangle = \langle L_1, U_1, C_1\rangle . \langle L_2, U_2, C_2\rangle \\
\langle L_1, U_1, C_1\rangle ; \langle L_2, U_2, C_2\rangle \mid I = \\
\qquad\qquad \begin{cases} (\texttt{pre}(I) \mid \langle L_1, U_1, C_1\rangle); (\texttt{pre}(I) \mid \langle L_2, U_2, C_2\rangle) & \text{if } \texttt{post}(I) = \emptyset \\ (\texttt{pre}(I) \mid \langle L_1, U_1, C_1\rangle); (\texttt{pre}(I) \mid \langle L_1, U_1, C_1\rangle \mid \langle L_2, U_2, C_2\rangle \mid \texttt{post}(I)) & \end{cases} \\
\langle L, U, C\rangle . \langle L_1, U_1, C_1\rangle ; \langle L_2, U_2, C_2\rangle = \\
\qquad\qquad \begin{cases} (\langle L, U, C\rangle . \langle L_1, U_1, C_1\rangle); \langle L_2, U_2, C_2\rangle & \text{if } \overline{\texttt{open}} \notin U \\ (\langle L, U, C\rangle . \langle L_1, U_1, C_1\rangle); (\langle L_1, U_1, C_1\rangle \sqcup \langle L_2, U_2, C_2\rangle) & \end{cases} \\
\langle L_1, U_1, C_1\rangle ; \langle L_2, U_2, C_2\rangle . I = \\
\qquad\qquad\quad \begin{cases} \langle L_1, U_1, C_1\rangle ; (\langle L_2, U_2, C_2\rangle \sqcup \texttt{pre}(I) \sqcup \texttt{post}(I)) & \text{if } \texttt{post}(I) \neq \emptyset \\ \langle L_1, U_1, C_1\rangle ; (\langle L_2, U_2, C_2\rangle \sqcup \texttt{pre}(I)) & \end{cases}
\end{array}
$$

Two local solutions of the shape $\langle L, U, C\rangle$ are composed by taking the least upper bound with respect to $\sqsubseteq$ (denoted by $\sqcup$), because we have to collect enough information to validate both processes. The parallel and sequential composition with a local solution of the shape $\langle L_1, U_1, C_1\rangle ; \langle L_2, U_2, C_2\rangle$ are based on the ideas explained above. Thus, any parallel process may end up both before and after the $\overline{\texttt{open}}$, also when two $\overline{\texttt{open}}$ may appear we don't know which of them will be executed first; in sequential composition the ordering establishes whether the process may end up either before or after the $\overline{\texttt{open}}$.

Moreover, we introduce *contexts*, which are expressions, built from local solutions and from the special expression $\diamond$ (denoting the hole) using parallel $(|)$ and sequential composition $(.)$ operators. Formally, a context is either $\diamond$, or $I.\, Z$ or $I \mid Z$, where I is a local solution and Z is a context. Contexts are needed to analyse the capabilities, appearing in a prefix; the hole actually shows where the local solution of the continuation has to be inserted (see rule **(pref)** of Tab. 4). A context Z filled with a local solution I, denoted by $Z[I]$, is evaluated to a local solution by applying the composition operators defined above.

move: $\models \mathtt{cap} : \langle L, U, C\rangle . \diamond$ iff $\mathtt{cap} \in U \wedge \mathtt{cap} \in \{\overline{\mathtt{in}}\, n, \overline{\mathtt{out}}\, n, \overline{\mathtt{out}}\, , \overline{\mathtt{in}}\, , n, \mathtt{out}\, n, \mathtt{in}\, n\}$
$\wedge\ (\mathtt{cap} = \mathtt{in}\, n \Rightarrow\ \forall \lambda \in L, \forall \mu$
$(n \in \rho(\mu) \wedge \mathcal{E}_{in}(\lambda, \mu) \Rightarrow \lambda \mathrel{\mathsf{E}} \phi(\mu)))$
$\wedge\ (\mathtt{cap} = \mathtt{out}\, n \Rightarrow \forall \lambda \in L, \forall \mu, \chi$
$(n \in \rho(\mu) \wedge \mathcal{E}_{out}(\lambda, \mu, \chi) \Rightarrow \lambda \mathrel{\mathsf{E}} \phi(\chi)))$

open: $\models \mathtt{open}\, n : \langle L, U, C\rangle . (\diamond \mid \langle L_1, U_1, C_1\rangle)$ iff $\mathtt{open}\, n \in U \wedge \forall \lambda \in L, \forall \mu$
$(n \in \rho(\mu) \wedge\ \mathcal{E}_{open}(\lambda, \mu) \Rightarrow$
$\mathtt{post}(\phi(\mu)) \sqsubseteq \langle L_1, U_1, C_1\rangle)$

co-open: $\models \overline{\mathtt{open}} : \langle L, U, C\rangle; \langle L', U', C'\rangle . \diamond$ iff $\overline{\mathtt{open}} \in U$

nil: $\models \mathbf{0} : \langle L, U, C\rangle$ iff true **res:** $\models (\nu n)P : I$ iff $\models P : I$

amb: $\models M_\lambda[\,P\,] : \langle L, U, C\rangle$ iff $\lambda \in U\ \wedge M\sigma \subseteq \rho(\lambda) \wedge \models P : \phi(\lambda) \wedge$
$\mathtt{pre}(\phi(\lambda)) = \langle L' \cup \{\lambda\}, U', C'\rangle$

msg : $\models \langle M\rangle : \langle L, U, C\rangle$ iff $M\sigma \subseteq C$ **bang :** $\models !P : I$ iff $I \mid I_1 = I \wedge \models P : I_1$

abs: $\models (\tilde{x})\, P : I$ iff $\mathtt{pre}(I) = \langle L, U, C\rangle \wedge \models P : I \wedge \forall \lambda \in L$
$(\mathtt{pre}(\phi(\lambda)) = \langle L', U', C' \cup \{\tilde{M}\}\rangle\ \Rightarrow \forall i \in \{1, k\} M_i \in \sigma(x_i))$

par : $\models P_1 \mid P_2 : I_1 \mid I_2$ iff $\models P_1 : I_1 \wedge \models P_2 : I_2$

pref: $\models M.\, P : Z[I]$ iff $\forall N \in M\sigma\ \models N : Z' \wedge Z' \preceq Z \wedge \models P : I$

Table 4. Control Flow Analysis for SA

The clauses for validation are shown in Tab. 4[3]. The judgments for: *expressions* are on the form $(\rho, \phi, \sigma) \models M : I.\, Z$, where M is a closed expression (without free variables) and $I.\, Z$ is a context; *processes* are on the form $(\rho, \phi, \sigma) \models P : I$, where P is a process and I a local solution. The rules for processes check whether (ρ, ϕ, σ) is a valid solution for P w.r.t local solution I, meaning that I contains enough information to approximate the behaviour of P, when enclosed inside any ambient with label λ, such that $\lambda \in L$ and $\mathtt{pre}(I) = \langle L, U, C\rangle$. Analogously for expressions.

The rules for *expressions* use also the following auxiliary notions: the function $\mathtt{f}$ collects, for any label λ, the labels of the possible fathers; the next three conditions check whether the capabilities of movement and opening are enabled. Let $\mathtt{f}(\lambda) = \{\mu \mid \mathtt{pre}(\phi(\mu)) = \langle L, U \cup \{\lambda\}, C\rangle\}$.

- $\mathcal{E}_{in}(\lambda, \mu) \equiv (\overline{\mathtt{in}} \in U_1 \vee (\overline{\mathtt{in}}\, m \in U_1 \wedge m \in \rho(\lambda)) \wedge (\mathtt{f}(\lambda) \cap \mathtt{f}(\mu) \neq \emptyset)$ where $\mathtt{pre}(\phi(\mu)) = \langle L_1, U_1, C_1\rangle$;
- $\mathcal{E}_{out}(\lambda, \mu, \chi) \equiv (\mu \in \mathtt{f}(\lambda)) \wedge (\chi \in \mathtt{f}(\mu)) \wedge (\overline{\mathtt{out}} \in U_1 \vee (\overline{\mathtt{out}}\, m \in U_1 \wedge m \in \rho(\lambda)))$ where $\mathtt{pre}(\phi(\chi)) = \langle L_1, U_1, C_1\rangle$;
- $\mathcal{E}_{open}(\lambda, \mu) \equiv (\lambda \in \mathtt{f}(\mu)) \wedge \overline{\mathtt{open}} \in U_1$, where $\mathtt{pre}(\phi(\mu)) = \langle L_1, U_1, C_1\rangle$.

In all the clauses the expression, appearing at top-level, is properly recorded in the local solution. Rule (**move**) handles the actions and coactions of move-

[3] In the rules we write $\models M : Z$ for $(\rho, \phi, \sigma) \models M : Z$; similarly for processes.

ment; the context is simply $\langle L, U, C\rangle.\diamond$. The additional conditions verify the possible movements produced by the capabilities, $\mathtt{in}\, n$ and $\mathtt{out}\, n$, on their possible enclosing ambients (those with label $\lambda \in L$). For instance, for $\mathtt{in}\, n$ we control whether there exists an ambient (named) n and with label μ, which may be a sibling of that with label λ and which offers the right coaction. When the ambient (with label λ) may move, it has to be recorded as possible sub-ambient of the target ambient (with label μ). To this aim, we use $\lambda \mathrel{\mathsf{E}} I$, where $\lambda \mathrel{\mathsf{E}} I$ iff, either $I = \langle L, U, C\rangle$ and $\lambda \in U$, or $I = \langle L_1, U_1, C_1\rangle; \langle L_2, U_2, C_2\rangle$ and $\lambda \in U_1 \cap U_2$.

In rule (**co-open**) the context is $\langle L, U, C\rangle; \langle L', U', C'\rangle.\diamond$, where $\overline{\mathtt{open}} \in U$, because the continuation ends up after $\overline{\mathtt{open}}$. In rule (**open**) the context $\langle L, U, C\rangle.(\diamond \mid \langle L_1, U_1, C_1\rangle)$ shows that the process unleashed by the opening of an ambient n (approximated by $\langle L_1, U_1, C_1\rangle$) may end up in parallel with the continuation.

In the clauses for *processes* the following notion is also used. Given an expression M, we define $M\sigma = \{N \in \mathcal{CE} \mid N = M\eta \text{ for a substitution } \eta \text{ such that } \eta(x) \in \sigma(x)\}$. The set $M\sigma$ reports the closed expressions, that may appear in place of M at run-time. The rules (**amb**) and (**msg**) are rather standard; the ambient and the message appear at top-level, and therefore have to be properly recorded in the local solution. Rule (**abs**) says that any message, which may be communicated inside the possible enclosing ambients, has to be recorded by the function σ. The rules for parallel, sequential composition and bang, (**par**), (**pref**) and (**bang**), exploit the auxiliary composition operations to properly combine the local solutions of the sub-processes. Rule (**pref**) says that the local solution of a process $M.P$ is obtained by filling the context, modelling the capability, with the local solution of the continuation. In order to find out a common context that correctly approximates the behaviour of all the capabilities that may belong to $M\sigma$ we use a relation $\preceq$ over contexts. The relation $\preceq$ is the least transitive and reflexive relation such that: (i) $\langle L, U, C\rangle.\diamond \preceq \langle L, U, C\rangle.(\diamond \mid \langle L', U', C'\rangle)$; and (ii) $\langle L, U, C\rangle; \langle L, U, C\rangle.\diamond \preceq \langle L, U, C\rangle.\diamond$.

Properties. A solution (ρ, ϕ, σ) is *valid* for a process P iff $(\rho, \phi, \sigma) \models P : \phi(\top)$, s.t. $\mathtt{pre}(\phi(\top)) = \langle L, U, C\rangle$ and $\top \in L$. The CFA satisfies standard properties (see [24, 18]); i.e. Subject Reduction (validity of solutions is preserved under reduction) and soundness (the static behaviour is a sound approximation of the dynamic one). Furthermore, there always exists a least (in terms of precision) solution which is valid [4]. To formalise soundness we use contexts, that are standard process expressions with a hole. A context is said *enabling* whenever the hole does not appear underneath a prefix, an abstraction or a bang.

Theorem 1 (Subject Reduction and Soundness). *Let (ρ, ϕ, σ) be a valid solution for process P. For any process P', s.t. $P \Longrightarrow P'$, we have that*

1. *(ρ, ϕ, σ) is a valid solution also for P';*
2. *if $P' \equiv C[n_\lambda[\,R\,]]$, for some enabling context C, then $n \in \rho(\lambda)$. Moreover, if $\mathtt{pre}(\phi(\lambda)) = \langle L, U, C\rangle$, we have also that: (a) if $R = M.P_1 \mid P_2$ and $M \in \mathcal{CE}$, then $M \in U$; (b) if $R = m_\mu[\,P_1\,] \mid P_2$, then $\mu \in U$; (c) if $R = \langle M\rangle \mid (x)P_1 \mid P_2$ and $M \in \mathcal{CE}$, then $M \in C$ and $M \in \sigma(x)$.*

[4] We omit for lack of space the formal definition of the ordering.

5 Applying the CFA to the encoding

We show how the MAC properties of BA [5] can be verified by analysing the encoding. A MAC policy is specified by a boolean predicate $\mathcal{P} \subseteq \mathcal{N}_{BA} \times \mathcal{N}_{BA} \times \mathcal{AM}$, where $\mathcal{AM} = \{r, w, rw\}$ defines the access modes (read, write, read-write).

Definition 1 (MAC Properties). *A BA process P* satisfies *the property $\mathcal{P}$ iff for any $P \Longrightarrow P'$, s.t. $P' \equiv C[m[\, R_1 \mid n[\, R_2 \mid Q_2\,] \mid Q_3\,]]$ for an enabling context C, we have that: (i) if $R_1 = (x)^n\, Q_1$ and $R_2 = \langle M \rangle^{\uparrow}$, then $\mathcal{P}(n, m, \alpha)$ and $\mathcal{P}(m, n, \beta)$, for $\alpha \in \{w, rw\}$ and $\beta \in \{r, rw\}$; (ii) if $R_1 = \langle M \rangle^n$ and $R_2 = (x)^{\uparrow}\, Q_1$, then $\mathcal{P}(n, m, \alpha)$ and $\mathcal{P}(m, n, \beta)$, for $\alpha \in \{r, rw\}$ and $\beta \in \{w, rw\}$.*

We introduce a *static property* on the solutions of the analysis that is *safe*, i.e. : if a solution for the encoding passes the test, then the source BA process satisfies the security property of Def. 1. The basic idea is to translate the conditions of Def. 1 into equivalent conditions on the behaviour of the auxiliary ambients that realise the inter-level communication actions. We examine, as an example, case (ii) corresponding to (`Input` $\uparrow$); (`Output` $\uparrow$) is analogous.

To this aim, we focus on the following process $Q = c[\, \langle M_2 \rangle^a\,] \mid a[\, \mathtt{in}\, c \mid \mathtt{in}\, b \mid (x)^{\uparrow}\, P \mid b[\, \langle M_1 \rangle^a\,]\,]$, and we consider its encoding (w.r.t k)

$$\llbracket Q \rrbracket^k \equiv !\overline{\mathtt{out}} \mid c_c[\, !\overline{\mathtt{out}} \mid !\overline{\mathtt{in}} \mid \mathtt{wd}_c(\langle \llbracket M_2 \rrbracket \rangle, a^{\mathtt{w}_1^{\downarrow}}, a^{\mathtt{w}_2^{\downarrow}}, a^{\mathtt{r}^{\uparrow}})\,] \mid a_a[\, !\overline{\mathtt{out}} \mid !\overline{\mathtt{in}} \mid \mathtt{in}\, c \mid \mathtt{in}\, b \mid \mathtt{ru}_a((\llbracket x \rrbracket)\, \llbracket P \rrbracket^a, a, a^{\mathtt{w}_1^{\downarrow}}, a^{\mathtt{w}_2^{\downarrow}}, a^{\mathtt{r}^{\uparrow}}) \mid b_b[\, !\overline{\mathtt{out}} \mid !\overline{\mathtt{in}} \mid \mathtt{wd}_b(\langle \llbracket M_1 \rrbracket \rangle, a^{\mathtt{w}_1^{\downarrow}}, a^{\mathtt{w}_2^{\downarrow}}, a^{\mathtt{r}^{\uparrow}})\,]\,]$$

For simplicity, we adopt in the encoding the following labels: for the BA ambients their names; for the auxiliary ambients the derived labels (see Sect. 3), where indexes are omitted (they usually are necessary to have distinct labels, but here are not relevant.). For instance, the ambients $a^{\mathtt{w}_1^{\downarrow}}$ and $a^{\mathtt{w}_2^{\downarrow}}$ appearing in $\mathtt{wd}_c(\langle \llbracket M_2 \rrbracket \rangle, a^{\mathtt{w}_1^{\downarrow}}, a^{\mathtt{w}_2^{\downarrow}}, a^{\mathtt{r}^{\uparrow}})$ have labels $\mathtt{w}^{\downarrow}_{1\,c}$ and $\mathtt{w}^{\downarrow}_{2\,c}$, resp.; while those occurring in $\mathtt{wd}_b(\langle \llbracket M_1 \rrbracket \rangle, a^{\mathtt{w}_1^{\downarrow}}, a^{\mathtt{w}_2^{\downarrow}}, a^{\mathtt{r}^{\uparrow}})$ have labels $\mathtt{w}^{\downarrow}_{1\,b}$ and $\mathtt{w}^{\downarrow}_{2\,b}$, resp.. Similarly, in $\mathtt{ru}_a((\llbracket x \rrbracket)\, \llbracket P \rrbracket^a, a, a^{\mathtt{w}_1^{\downarrow}}, a^{\mathtt{w}_2^{\downarrow}}, a^{\mathtt{r}^{\uparrow}})$ the ambient $a^{\mathtt{r}^{\uparrow}}$ has label $\mathtt{r}^{\uparrow}_a$.

Process Q obviously satisfies the MAC policy, specified only by $\mathcal{P}(a, c, r)$ and $\mathcal{P}(c, a, w)$ (meaning that a can read from c and c can write to a). In fact, ambient a may move inside ambient c, and there communicate; while it does not enter inside and communicate with ambient b (b is a sub-ambient of a).

The condition on the encoded process has to capture whether ambient a may communicate with ambient c (or analogously with ambient b). The most relevant auxiliary ambients used in the simulation of (`Input` $\uparrow$) are: the ambient $a^{\mathtt{r}^{\uparrow}}$, containing the upward abstraction $(\llbracket x \rrbracket)\, \llbracket P \rrbracket^a$; the ambients $a^{\mathtt{w}_2^{\downarrow}}$, one for b and one for c, containing the downward outputs $\langle \llbracket M_1 \rrbracket \rangle$ and $\langle \llbracket M_2 \rrbracket \rangle$ resp.. The communication between a and c takes place whenever $a^{\mathtt{r}^{\uparrow}}$ ends up (by exiting from a) inside the ambient $a^{\mathtt{w}_2^{\downarrow}}$, local to c (as shown by the label $\mathtt{w}^{\downarrow}_{2\,c}$), and there is opened; similarly, for the communication between a and b. Therefore, one way to capture these interactions is to detect inside which ambients $a^{\mathtt{w}_2^{\downarrow}}$ the ambient $a^{\mathtt{r}^{\uparrow}}$ may end up. This is formalised by the following definition, according to the labelling of the auxiliary ambients, explained in Sect. 3.

Definition 2 (Static Property). *Let (ρ, ϕ, σ) be a solution. We say that (ρ, ϕ, σ)* satisfies *the property $\mathcal{P}$ when the following conditions hold:*
(i) if $\mathtt{pre}(\phi((\mathtt{r}^{\downarrow}_{2\mu}, j))) = \langle L, U, C\rangle$, $(\mathtt{w}^{\uparrow}{}_{\lambda}, i) \in U$, $n^{\mathtt{r}^{\downarrow}_2} \in \rho((\mathtt{r}^{\downarrow}_{2\mu}, j))$, $n^{\mathtt{w}^{\uparrow}} \in \rho((\mathtt{w}^{\uparrow}{}_{\lambda}, i))$ and $n \in \rho(\lambda)$, then for any $m \in \rho(\mu)$, we have $\mathcal{P}(n, m, \alpha)$ and $\mathcal{P}(m, n, \beta)$, where $\alpha \in \{w, rw\}$ and $\beta \in \{r, rw\}$;
(ii) if $\mathtt{pre}(\phi((\mathtt{w}^{\downarrow}_{2\mu}, j))) = \langle L, U, C\rangle$, $(\mathtt{r}^{\uparrow}{}_{\lambda}, i) \in U$, $n^{\mathtt{w}^{\downarrow}_2} \in \rho((\mathtt{w}^{\downarrow}_{2\mu}, j))$, $n^{\mathtt{r}^{\uparrow}} \in \rho((\mathtt{r}^{\uparrow}{}_{\lambda}, i))$ and $n \in \rho(\lambda)$, then for any $m \in \rho(\mu)$, we have $\mathcal{P}(n, m, \alpha)$ and $\mathcal{P}(m, n, \beta)$, where $\alpha \in \{r, rw\}$ and $\beta \in \{w, rw\}$.

Theorem 2 (Safeness). *Let (ρ, ϕ, σ) be a valid solution for $[\![P]\!]^N$. If (ρ, ϕ, σ) satisfies the property $\mathcal{P}$, then the BA process P satisfies the property $\mathcal{P}$.*

We apply this approach to the example above, by analysing the encoding of Q. The following is a valid solution [5], where (ρ, ϕ, σ), and $\sigma(x) = \{M_2\}$, $\rho(\mathtt{r}^{\uparrow}_a) = a^{\mathtt{r}^{\uparrow}}$, $\rho(\mathtt{c}^{\mathtt{ru}}_a) = \mathtt{c}^{\mathtt{ru}}$, $\rho(g_b) = \rho(g_c) = a^g$ for $g \in \{\mathtt{w}^{\downarrow}_1, \mathtt{w}^{\downarrow}_2\}$, $\rho(\mathtt{t}^{\mathtt{wd}}{}_b) = \rho(\mathtt{t}^{\mathtt{wd}}{}_c) = \mathtt{t}^{\mathtt{wd}}$,

$$
\begin{array}{l}
\phi(\top) = \langle\{\top\}, \{\overline{\mathtt{out}}, a, c, \mathtt{r}^{\uparrow}_a\}, \emptyset\rangle \quad \phi(b) = \langle\{b\}, \{\overline{\mathtt{out}}, \overline{\mathtt{in}}, \mathtt{w}^{\downarrow}_{1b}, \mathtt{w}^{\downarrow}_{2b}, \mathtt{t}^{\mathtt{wd}}_b, \mathtt{open}\,\mathtt{t}^{\mathtt{wd}}\}, \emptyset\rangle \\
\phi(a) = \langle\{a\}, \{\overline{\mathtt{out}}, \overline{\mathtt{in}}, \mathtt{in}\,b, \mathtt{in}\,c, b, \mathtt{in}\,a^{\mathtt{w}^{\downarrow}_1}, \mathtt{in}\,a^{\mathtt{w}^{\downarrow}_2}, \mathtt{r}^{\uparrow}_a, \mathtt{open}\,\mathtt{c}^{\mathtt{ru}}, \mathtt{out}\,a^{\mathtt{w}^{\downarrow}_2}, \mathtt{c}^{\mathtt{ru}}_a\}, \emptyset\rangle \\
\phi(c) = \langle\{c\}, \{\overline{\mathtt{out}}, \overline{\mathtt{in}}, a, \mathtt{r}^{\uparrow}_a, \mathtt{w}^{\downarrow}_{1c}, \mathtt{w}^{\downarrow}_{2c}, \mathtt{t}^{\mathtt{wd}}_c, \mathtt{open}\,\mathtt{t}^{\mathtt{wd}}\}, \emptyset\rangle \\
\phi(\mathtt{r}^{\uparrow}_a) = \langle\{\mathtt{r}^{\uparrow}_a\}, \{\mathtt{out}\,a, \overline{\mathtt{open}}\}, \emptyset\rangle; \langle\{\mathtt{w}^{\downarrow}_{2c}\}, \{\mathtt{c}^{\mathtt{ru}}_a\}, \emptyset\rangle \\
\phi(\mathtt{c}^{\mathtt{ru}}_a) = \langle\{\mathtt{c}^{\mathtt{ru}}_a\}, \{\mathtt{in}\,a, \overline{\mathtt{open}}\}, \emptyset\rangle; \langle\{a\}, \emptyset, \emptyset\rangle \\
\phi(\mathtt{w}^{\downarrow}_{1b}) = \langle\{\mathtt{w}^{\downarrow}_{1b}\}, \{\overline{\mathtt{in}}, \overline{\mathtt{out}}, \mathtt{in}\,\mathtt{t}^{\mathtt{wd}}, \overline{\mathtt{open}}, \mathtt{w}^{\downarrow}_{2b}\}, \emptyset\rangle; \langle\{\mathtt{t}^{\mathtt{wd}}{}_b\}, \{\mathtt{w}^{\downarrow}_{2b}\}, \emptyset\rangle \\
\phi(\mathtt{w}^{\downarrow}_{2b}) = \langle\{\mathtt{w}^{\downarrow}_{2b},\}, \{\overline{\mathtt{in}}, \overline{\mathtt{out}}\,a^{\mathtt{r}^{\uparrow}}, \mathtt{open}\,a^{\mathtt{r}^{\uparrow}}\}\rangle\{M_1\} \\
\phi(\mathtt{t}^{\mathtt{wd}}{}_b) = \langle\{\mathtt{t}^{\mathtt{wd}}{}_b\}, \{\overline{\mathtt{in}}\,a^{\mathtt{w}^{\downarrow}_1}, \mathtt{open}\,a^{\mathtt{w}^{\downarrow}_1}, \overline{\mathtt{open}}, \mathtt{w}^{\downarrow}_{1b}, \mathtt{w}^{\downarrow}_{2b}\}, \emptyset\rangle; \langle\{b\}, \{\mathtt{w}^{\downarrow}_{1b}, \mathtt{w}^{\downarrow}_{2b}\}, \emptyset\rangle \\
\phi(\mathtt{w}^{\downarrow}_{1c}) = \langle\{\mathtt{w}^{\downarrow}_{1c}\}, \{\overline{\mathtt{in}}, \overline{\mathtt{out}}, \mathtt{in}\,\mathtt{t}^{\mathtt{wd}}, \overline{\mathtt{open}}, \mathtt{w}^{\downarrow}_{2c}, \mathtt{r}^{\uparrow}_a, a\}, \emptyset\rangle; \langle\{\mathtt{t}^{\mathtt{wd}}{}_c\}, \{\mathtt{w}^{\downarrow}_{2c}, \mathtt{r}^{\uparrow}_a, a\}, \emptyset\rangle \\
\phi(\mathtt{w}^{\downarrow}_{2c}) = \langle\{\mathtt{w}^{\downarrow}_{2c}\}, \{\overline{\mathtt{in}}, \overline{\mathtt{out}}\,a^{\mathtt{r}^{\uparrow}}, \mathtt{open}\,a^{\mathtt{r}^{\uparrow}}, \mathtt{r}^{\uparrow}_a, a, \mathtt{c}^{\mathtt{ru}}_a\}, \{M_2\}\rangle \\
\phi(\mathtt{t}^{\mathtt{wd}}{}_c) = \langle\{\mathtt{t}^{\mathtt{wd}}{}_c\}, \{\overline{\mathtt{in}}\,a^{\mathtt{w}^{\downarrow}_1}, \mathtt{open}\,a^{\mathtt{w}^{\downarrow}_1}, \overline{\mathtt{open}}, \mathtt{w}^{\downarrow}_{1c}, \mathtt{w}^{\downarrow}_{2c}, \mathtt{r}^{\uparrow}_a, a\}, \emptyset\rangle; \langle\{c\}, \{\mathtt{w}^{\downarrow}_{1c}, \mathtt{w}^{\downarrow}_{2c}, \mathtt{r}^{\uparrow}_a, a\}, \emptyset\rangle
\end{array}
$$

The solution says that the ambient $a^{\mathtt{r}^{\uparrow}}$ (with label $\mathtt{r}^{\uparrow}_a$) can enter only inside the ambient $a^{\mathtt{w}^{\downarrow}_2}$ with label $\mathtt{w}^{\downarrow}_{2c}$, as shown by $\phi(\mathtt{w}^{\downarrow}_{2c})$ and $\phi(\mathtt{w}^{\downarrow}_{2b})$. Since this kind of access among a and c is authorised, the policy is satisfied according to Def. 2.

Note that the crucial information to achieve this result is that the capability $\mathtt{in}\,b$ cannot be exercised (since ambients a and b cannot be siblings); and therefore ambient a cannot enter inside ambient b (see $\phi(b)$). Moreover, the local solutions are essential $\langle L_1, U_1, C_1\rangle; \langle L_2, U_2, C_2\rangle$ for the auxiliary ambients to accurately distinguish what happens before and after their dissolution. A weaker analysis as [24] (see Ex. 1) cannot argue that $\mathtt{in}\,a$ has been consumed before $\mathtt{c}^{\mathtt{ru}}$ (carrying the continuation of the abstraction) is opened inside ambient a (see $\phi(\mathtt{c}^{\mathtt{ru}}_a)$ and $\phi(a)$). As a consequence, it would report that ambient a may enter inside itself, by exercising $\mathtt{in}\,a$, and that the MAC property may be violated (as $\mathtt{in}\,b$ may run ambient a inside its possible sibling b). Also the types of [5] for BA cannot prove this property, precisely because they do not capture that $\mathtt{in}\,b$ cannot be exercised inside a.

[5] We assume for simplicity that $P = \mathbf{0}$ and that M_1 and M_2 are two closed expressions. Also, we omit ρ for the BA ambients (it is the identity).

6 Conclusions

We have presented: – an encoding of BA into a variant of SA; – a CFA for SA that refines the proposal of [24] with a finer treatment of the critical opening action, along the lines of the type systems for SA [20, 12, 15, 1, 17] (see Ex. 1). The CFA is sufficiently informative to capture the typical security properties of SA [7, 12, 3, 11, 2] controlling which boundaries an ambient may cross and which ambients it may open.

These results are interesting by themselves, but also when combined together, given that, by means of the translation, the CFA can be applied to verify security properties also of BA, in particular the MAC properties. Other properties can probably be verified this way, e.g. the mobility property of [23]. The relevance of our CFA for this application is demonstrated by the example, discussed in Sect. 5, which cannot be proved using either the types of [5] for BA or the adaptation of CFA of [24] to SA. Several refinements [16, 25, 11, 19, 13] of [24] have been proposed. The analyses of [11, 13, 19] for MA/SA further restrict the space of possible interactions, by exploiting more contextual information, and better handle replication. It is not clear whether their treatment of the opening is sufficiently accurate for analysing the encoding. The CFA for MA of [16, 25] are more complex than [24] (and also of our CFA), as they exploit more detailed information about the possible run-time shape of processes, and could be profitably applied to the encoding to derive more accurate predictions. The CFA in [2] verifies an information flow property, which does not seem adequate to capture the MAC properties. The type systems of MA/SA [7, 8, 20, 3, 1, 12, 15, 17] are typically simpler and more elegant than the analyses. They cannot however distinguish the auxiliary ambients with the same name and therefore cannot give better results than the types for BA [5], when applied to the encoded processes (see e.g. the use of labels in the example of Sect. 5).

Finally, we mention that in [4, 5, 23] a different semantics for the BA inter-level communications is used, where downward-upward communication actions interact with local prefixes rather than with each other. In [18] we have adapted the approach of this paper to the BA version of [4], which seems more expressive, and we have obtained similar results w.r.t [5]. Although the encoding of [4] is much more complex, it can be derived along the lines of Sect. 3. It is enough to modify the translation of local and inter-level communication actions, by using different auxiliary ambients in the protocols of Tab. 3 and by introducing sum between non-prefixed terms in SA, reflecting the intrinsic non-determinism on local prefixes of this model. We intend to study the encoding also of another version of BA [6], which extends [10] with special coactions using passwords.

References

1. T. Amtoft and A. J. Kfoury and S. M. Pericas-Geertsen. *What are Polymorphically-Typed Ambients?* Proc. of ESOP'01, LNCS 2028, 2001.
2. C. Braghin, A. Cortesi and R. Focardi. *Control Flow Analysis for information flow security in mobile ambients.* Proc. of FMOODS'02, pp. 197-212, 2002.

3. M. Bugliesi and G. Castagna. *Secure safe ambients.* Proc. of POPL '01, pp. 222-235, 2001.
4. M. Bugliesi, G. Castagna, S. Crafa. *Boxed Ambients.* Proc. of TACS'01, LNCS 2225, pp. 36-61, 2001.
5. — *Reasoning about Security in Mobile Ambients.* Proc. of CONCUR'01, LNCS 2154, pp. 102-120, 2001.
6. M. Bugliesi, S. Crafa, M. Merro and V. Sassone. *Communication Interference in Boxed Ambients.* Proc. of FSTTCS'02, LNCS 2556, pp. 71–84, 2002.
7. L. Cardelli, G. Ghelli, and A.D. Gordon. *Mobility types for mobile ambients.* Proc. of ICALP'99, LNCS 1644, pp. 230-239, 1999.
8. — *Types for the ambient calculus.* Information and Computation, 177(2), 2002.
9. L. Cardelli and A.D. Gordon. *Mobile ambients.* Proc. of FoSSaCS '98, LNCS 1378, pp. 140-155, 1998.
10. S. Crafa, M. Bugliesi and G. Castagna. *Information Flow Security in Boxed Ambients.* Proc. of FWAN '02, ENTCS, 66(3), 2002.
11. P. Degano, F. Levi and C. Bodei. *Safe Ambients: Control Flow Analysis and Security.* Proc. of ASIAN '00, LNCS 1961, pp. 199-214, 2000.
12. M. Dezani-Ciancaglini and I. Salvo. *Security Types for Mobile Safe Ambients.* Proc. of ASIAN '00, LNCS 1961, pp. 215-236, 2000.
13. J. Feret. *Abstract Interpretation-Based Static Analysis of Mobile Ambients.* Proc. of SAS'01, LNCS 2126, pp. 412-430, 2001.
14. X. Guan, Y. Yang, and J. You. *Making Ambients More Robust.* Proc. of the International Conference on Software: Theory and Practise, 377–384, 2000.
15. — *Typing Evolving Ambients.* Inf. Processing Letters, 80(5), pp. 265–270, 2001.
16. R. R. Hansen and J. G. Jensen and F. Nielson and H. R.Nielson. *Abstract Interpretation of Mobile Ambients.* Proc. of SAS'99, LNCS 1694, pp. 135-148, 1999.
17. F. Levi. *Types for Evolving Communication in Safe Ambients.* Proc. of VMCAI '03, LNCS 2575, pp. 102-115, 2003.
18. F. Levi and C. Bodei *A Control Flow Analysis for Safe and Boxed Ambients (Extended Version).* Available at `http://www.di.unipi.it/~ levifran/papers.html`
19. F. Levi and S. Maffeis. *An Abstract Interpretation Framework for Analysing Mobile Ambients.* Proc. of SAS '01, LNCS 2126, pp. 395-411, 2001.
20. F. Levi and D. Sangiorgi. *Controlling Interference in Ambients.* Proc. of POPL '00, pp. 352-364, 2000.
21. — *Mobile Safe Ambients.* TOPLAS, 25(1), 1–69, 2003.
22. M. Merro and M. Hennessy. *Bisimulation congruences in Safe Ambients.* Proc. of POPL '02, 2002.
23. M. Merro and V. Sassone. *Typing and Subtyping Mobility in Boxed Ambients.* Proc. of CONCUR'02, LNCS 2421, pp. 304-320, 2002.
24. F. Nielson, H.R. Nielson, R.R. Hansen. *Validating firewalls using flow logics.* Theoretical Computer Science, 283(2), 381-418, 2002. Also (joint work also with J.G. Jensen) appeared in the Proc. of CONCUR'99, LNCS 1664, 1999.
25. H. R. Nielson and F. Nielson. *Shape Analysis for Mobile Ambients.* Proc. of POPL' 00, pp. 135-148, 2000.