A Commitment Relation for the Ambient Calculus

Luca Cardelli
Microsoft Research
luca@luca.demon.co.uk

Andrew D. Gordon
Microsoft Research
adg@microsoft.com

October 6, 2000

Abstract

We present a commitment relation, a kind of labeled transition system, for the ambient calculus. This note is an extract from an unpublished annex to our original article [2] on the ambient calculus.

1 Review of the Ambient Calculus

In this section we review the syntax of the ambient calculus, and the structural congruence and reduction relations.

1.1 Capabilities and Processes

We assume an infinite set of *names*. We let m and n range over names. Moreover, we assume there is an infinite collections of *variables* ranged over by metavariables x, y, z. The sets of *capabilities* and *processes* are defined by the grammars:

Mobility and Communication Primitives

$M ::=$	capability
x	variable
n	name
$in\ M$	can enter into M
$out\ M$	can exit out of M
$open\ M$	can open M
ϵ	null

$M.M'$	path
$P, Q, R ::=$	process
$(\nu n)P$	restriction
$\mathbf{0}$	inactivity
$P \mid Q$	composition
$!P$	replication
$M[P]$	ambient
$M.P$	action
$(x).P$	input action
$\langle M \rangle$	async output action

The general forms *in* M, *out* M and *open* M allow for the ambient to be an arbitrary capability M. The only useful cases are for M to be a name, or a variable that gets instantiated to a name. Similarly, the ambient syntax $M[P]$ allows M to be an arbitrary capability. The only useful case is for M to be a name, or a variable that gets instantiated to a name.

We identify capabilities up to the following equations:

$$\begin{aligned} \epsilon.M &= M \\ M.\epsilon &= M \\ (L.M).N &= L.(M.N) \end{aligned}$$

The following table defines the sets $\mathit{fn}(M)$ and $\mathit{fv}(M)$ of *free names* and *free variables* of a capability M, and the sets $\mathit{fn}(P)$ and $\mathit{fv}(P)$ of *free names* and *free variables* of a process P.

Free Names and Variables of Capabilities and Processes

$\mathit{fn}(x) = \varnothing$	$\mathit{fv}(x) = \{x\}$
$\mathit{fn}(n) = \{n\}$	$\mathit{fv}(n) = \varnothing$
$\mathit{fn}(\mathit{in}\ M) = \mathit{fn}(M)$	$\mathit{fv}(\mathit{in}\ M) = \mathit{fv}(M)$
$\mathit{fn}(\mathit{out}\ M) = \mathit{fn}(M)$	$\mathit{fv}(\mathit{out}\ M) = \mathit{fv}(M)$
$\mathit{fn}(\mathit{open}\ M) = \mathit{fn}(M)$	$\mathit{fv}(\mathit{open}\ M) = \mathit{fv}(M)$
$\mathit{fn}(\epsilon) = \varnothing$	$\mathit{fv}(\epsilon) = \varnothing$
$\mathit{fn}(M.M') = \mathit{fn}(M) \cup \mathit{fn}(M')$	$\mathit{fv}(M.M') = \mathit{fv}(M) \cup \mathit{fv}(M')$
$\mathit{fn}((\nu n)P) = \mathit{fn}(P) - \{n\}$	$\mathit{fv}((\nu n)P) = \mathit{fv}(P)$
$\mathit{fn}(\mathbf{0}) = \varnothing$	$\mathit{fv}(\mathbf{0}) = \varnothing$
$\mathit{fn}(P \mid Q) = \mathit{fn}(P) \cup \mathit{fn}(Q)$	$\mathit{fv}(P \mid Q) = \mathit{fv}(P) \cup \mathit{fv}(Q)$
$\mathit{fn}(!P) = \mathit{fn}(P)$	$\mathit{fv}(!P) = \mathit{fv}(P)$
$\mathit{fn}(M[P]) = \mathit{fn}(M) \cup \mathit{fn}(P)$	$\mathit{fv}(M[P]) = \mathit{fv}(M) \cup \mathit{fv}(P)$
$\mathit{fn}(M.P) = \mathit{fn}(M) \cup \mathit{fn}(P)$	$\mathit{fv}(M.P) = \mathit{fv}(M) \cup \mathit{fv}(P)$

$fn((x).P) = fn(P)$	$fv((x).P) = fv(P) - \{x\}$
$fn(\langle M \rangle) = fn(M)$	$fv(\langle M \rangle) = fv(M)$

We say a capability M is *closed* if and only if $fv(M) = \varnothing$; similarly, a process P is *closed* if and only if $fv(M) = \varnothing$.

If phrase ϕ is a capability or a process, we write $\phi\{x \leftarrow M\}$ for the outcome of a capture-avoiding substitution of the capability M for each free occurrence of the variable x in ϕ. We identify processes up to renaming of bound names and variables: $(\nu n)P = (\nu m)(P\{n \leftarrow m\})$ if $m \notin fn(P)$; $(x).P = (y).(P\{x \leftarrow y\})$ if $y \notin fn(P)$.

1.2 Structural Congruence

We let *structural congruence*, $\equiv$, be the least relation on processes that satisfies the following equations and rules:

Structural Congruence

$P \equiv P$	(Struct Refl)
$Q \equiv P \Rightarrow P \equiv Q$	(Struct Symm)
$P \equiv Q, Q \equiv R \Rightarrow P \equiv R$	(Struct Trans)
$P \equiv Q \Rightarrow (\nu n)P \equiv (\nu n)Q$	(Struct Res)
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$	(Struct Par)
$P \equiv Q \Rightarrow !P \equiv !Q$	(Struct Repl)
$P \equiv Q \Rightarrow M[P] \equiv M[Q]$	(Struct Amb)
$P \equiv Q \Rightarrow M.P \equiv M.Q$	(Struct Action)
$P \equiv Q \Rightarrow (x).P \equiv (x).Q$	(Struct Input)
$P \mid Q \equiv Q \mid P$	(Struct Par Comm)
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Struct Par Assoc)
$!P \equiv P \mid !P$	(Struct Repl Par)
$(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$	(Struct Res Res)
$n \notin fn(P) \Rightarrow (\nu n)(P \mid Q) \equiv P \mid (\nu n)Q$	(Struct Res Par)
$n \neq m \Rightarrow (\nu n)m[P] \equiv m[(\nu n)P]$	(Struct Res Amb)
$P \mid \mathbf{0} \equiv P$	(Struct Zero Par)
$(\nu n)\mathbf{0} \equiv \mathbf{0}$	(Struct Zero Res)
$!\mathbf{0} \equiv \mathbf{0}$	(Struct Zero Repl)
$\epsilon.P \equiv P$	(Struct ϵ)
$(M.M').P \equiv M.M'.P$	(Struct .)
$\rightarrow^*$	reflexive, transitive closure of $\rightarrow$

1.3 Reduction

We let the *reduction relation*, $\rightarrow$, be the least relation on processes to satisfy the following rules:

Reduction

$n[\mathit{in}\ m.P \mid Q] \mid m[R] \rightarrow m[n[P \mid Q] \mid R]$	(Red In)
$m[n[\mathit{out}\ m.P \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R]$	(Red Out)
$\mathit{open}\ n.P \mid n[Q] \rightarrow P \mid Q$	(Red Open)
$\langle M \rangle \mid (x).P \rightarrow P\{x \leftarrow M\}$	(Red I/O)
$P \rightarrow Q \Rightarrow P \mid R \rightarrow Q \mid R$	(Red Par)
$P \rightarrow Q \Rightarrow (\nu n)P \rightarrow (\nu n)Q$	(Red Res)
$P \rightarrow Q \Rightarrow n[P] \rightarrow n[Q]$	(Red Amb)
$P' \equiv P, P \rightarrow Q, Q \equiv Q' \Rightarrow P' \rightarrow Q'$	(Red $\equiv$)

2 Commitment

This section introduces a labeled transition system, the *commitment relation*, for the ambient calculus. The use of structural congruence to define reduction results in an elegant and compact definition, but also complicates proofs about reduction. As in the π-calculus [1, 4, 5], one purpose of a labeled transition system is to characterise reduction independently of structural congruence.

The commitment relation is a relation indexed by a set of *actions*, ranged over by α, given as follows.

Actions

$\alpha ::=$	actions
τ	internal action
$\mathit{in}\ n$	enter ambient n
$\mathit{enter}\ n$	invite sibling ambient to enter
$\overline{\mathit{enter}\ n}$	move ambient into sibling n
$\mathit{out}\ n$	exit ambient n
$\mathit{exit}\ n$	exit parent ambient
$\mathit{open}\ n$	open ambient n
$\overline{\mathit{open}\ n}$	open self
anon	anonymous input
$\overline{\mathit{anon}}$	anonymous output

To express the commitment relation, we need an auxiliary notion of *evaluation expression*, given as follows, where X drawn from an infinite collection of *process variables.*

Evaluation Expressions

$\mathcal{E} ::=$	evaluation expressions
X	process variable
$(\nu n)\mathcal{E}$	restriction
$P \mid \mathcal{E}$	left composition
$\mathcal{E} \mid Q$	right composition
$n[\mathcal{E}]$	ambient

We identify evaluation expressions up to renaming of bound names. We let $fn(\mathcal{E})$ be the set of names occurring free in evaluation expression $\mathcal{E}$. We let $fv(\mathcal{E})$ be the set of variables and process variables occurring free in evaluation expression $\mathcal{E}$. If $\mathcal{E}$ is an evaluation expression, we write $\mathcal{E}\{X \leftarrow P\}$ for the outcome of substituting the process P for each occurrence of the process variable X in evaluation expression $\mathcal{E}$. Given the grammar for evaluation expressions, $\mathcal{E}$ has either no occurrence of the process variable X, or exactly one occurrence.

The commitment relation relates a process to an *outcome*, O, which is either a process, an *abstraction* or a *concretion.* Here are the grammars for ambients, abstractions, concretions and outcomes.

Ambients, Abstractions, Concretions, Outcomes

$A, B ::= n[P]$	ambients
$F, G ::=$	abstractions
$(X)\mathcal{E}$	ambient abstraction
$(x)P$	capability abstraction
$C, D ::=$	concretions
$(\nu\vec{n})\langle A\rangle P$	ambient concretion
$(\nu\vec{n})\langle M\rangle P$	capability concretion
$O ::= P \mid F \mid C$	outcomes

In an ambient abstraction $(X)\mathcal{E}$, the process variable X is bound; its scope is the evaluation expression $\mathcal{E}$. In a capability abstraction $(x)P$, the variable x is bound; its scope is the process P. In an ambient concretion $(\nu\vec{p})\langle A\rangle P$, the names $\vec{p}$ are bound; their scope is the ambient A and the process P. In a capability concretion $(\nu\vec{p})\langle M\rangle P$, the names $\vec{p}$ are bound; their scope is the capability M and the process P.

We let ϕ range over both ambients and capabilities, so that any concretion takes the form $(\nu\vec{n})\langle\phi\rangle P$.

We extend the notions of free names, free variables and substitution to abstractions and concretions in the obvious way. We identify abstractions and concretions up to renaming of bound names and bound variables.

The commitment relation is written as follows:

The Commitment Relation

$P \xrightarrow{\alpha} O$	P α-commits to outcome O

The commitment relation is the union of ten different judgments, one for each of the different kinds of action. Judgments of the form $P \xrightarrow{\tau} Q$ indicate that process P proceeds to Q by an internal computation step. We will later prove that this judgment corresponds to the reduction relation on processes. The role of the other nine judgments making up the commitment relation is to represent intermediate stages in the derivation of a τ-commitment.

We introduce the rules that inductively define the commitment relation in groups, beginning with a group of congruence rules. As a convenient shorthand for writing (Comm Par 1) and (Comm Par 2), we define the *composition*, $O \parallel O'$ of outcomes O and O', at least one of which is a process, as follows:

$$\begin{array}{rcll}
P \parallel Q & \triangleq & P \mid Q & \\
((X)\mathcal{E}) \parallel Q & \triangleq & (X)(\mathcal{E} \mid Q) & \\
P \parallel ((X)\mathcal{E}) & \triangleq & (X)(P \mid \mathcal{E}) & \\
((x)P) \parallel Q & \triangleq & (x)(P \mid Q) & \text{if } x \notin \mathit{fn}(Q) \\
P \parallel ((x)Q) & \triangleq & (x)(P \mid Q) & \text{if } x \notin \mathit{fn}(P) \\
((\nu\vec{m})\langle\phi\rangle P) \parallel Q & \triangleq & (\nu\vec{m})\langle\phi\rangle(P \mid Q) & \text{if } \{\vec{m}\} \cap \mathit{fn}(Q) = \varnothing \\
P \parallel ((\nu\vec{m})\langle\phi\rangle Q) & \triangleq & (\nu\vec{m})\langle\phi\rangle(P \mid Q) & \text{if } \{\vec{m}\} \cap \mathit{fn}(P) = \varnothing
\end{array}$$

As a shorthand for writing (Comm Res), we define the *restriction*, $\underline{(\nu n)}O$ of outcome O, as follows:

$$\begin{array}{rcll}
\underline{(\nu n)}P & \triangleq & (\nu n)P & \\
\underline{(\nu n)}(X)\mathcal{E} & \triangleq & (X)(\nu n)\mathcal{E} & \\
\underline{(\nu n)}(x)P & \triangleq & (x)(\nu n)P & \\
\underline{(\nu n)}(\nu\vec{m})\langle\phi\rangle P & \triangleq & (\nu n, \vec{m})\langle\phi\rangle P & \text{if } n \in \mathit{fn}(\phi) \text{ and } n \notin \{\vec{m}\} \\
\underline{(\nu n)}(\nu\vec{m})\langle\phi\rangle P & \triangleq & (\nu\vec{m})\langle\phi\rangle(\nu n)P & \text{if } n \notin \mathit{fn}(\phi) \text{ and } n \notin \{\vec{m}\}
\end{array}$$

Congruence

(Comm Amb) $\dfrac{P \xrightarrow{\tau} Q}{n[P] \xrightarrow{\tau} n[Q]}$ (Comm Res) $\dfrac{P \xrightarrow{\alpha} O \quad n \notin \mathit{fn}(\alpha)}{(\nu n)P \xrightarrow{\alpha} \underline{(\nu n)}O}$

(Comm Par 1) $\dfrac{P \xrightarrow{\alpha} O}{P \mid Q \xrightarrow{\alpha} O \parallel Q}$ (Comm Par 2) $\dfrac{Q \xrightarrow{\alpha} O}{P \mid Q \xrightarrow{\alpha} P \parallel O}$

The next three groups of rules explain the behavior of processes of the form *in* $n.P$, *out* $n.P$ and *open* $n.P$. As a convenient shorthand for writing (Comm τ-*enter* 1) and (Comm τ-*enter* 2), we define the *interactions*, $F@C$ and $C@F$, of an ambient abstraction F and an ambient concretion C, as follows:

$$\begin{array}{lcll}
((X)\mathcal{E})@(\nu\vec{m})\langle A\rangle P & \triangleq & (\nu\vec{m})(\mathcal{E}\{X \leftarrow A\} \mid P) & \text{if } \{\vec{m}\} \cap \mathit{fn}(\mathcal{E}) = \varnothing \\
((\nu\vec{m})\langle A\rangle P)@(X)\mathcal{E} & \triangleq & (\nu\vec{m})(P \mid \mathcal{E}\{X \leftarrow A\}) & \text{if } \{\vec{m}\} \cap \mathit{fn}(\mathcal{E}) = \varnothing
\end{array}$$

Entering an Ambient

(Comm *in*) $\dfrac{}{(\mathit{in}\ n.M).P \xrightarrow{\mathit{in}\ n} M.P}$ (Comm $\overline{\mathit{enter}}$) $\dfrac{P \xrightarrow{\mathit{in}\ n} Q}{m[P] \xrightarrow{\overline{\mathit{enter}\ n}} \langle m[Q]\rangle\mathbf{0}}$ (Comm *enter*) $\dfrac{}{n[P] \xrightarrow{\mathit{enter}\ n} (X)n[X \mid P]}$

(Comm τ-*enter* 1) $\dfrac{P \xrightarrow{\overline{\mathit{enter}\ n}} C \quad Q \xrightarrow{\mathit{enter}\ n} F}{P \mid Q \xrightarrow{\tau} C@F}$ (Comm τ-*enter* 2) $\dfrac{P \xrightarrow{\mathit{enter}\ n} F \quad Q \xrightarrow{\overline{\mathit{enter}\ n}} C}{P \mid Q \xrightarrow{\tau} F@C}$

We begin an enumeration of the nine judgments that represent intermediate stages in deriving τ-commitments, with the three used in the rules for entering an ambient:

(1) $P \xrightarrow{\mathit{in}\ n} Q$: process P instructs its enclosing ambient to move into sibling ambient n.

(2) $P \xrightarrow{\mathit{enter}\ n} (X)\mathcal{E}$: an ambient n within P is ready to accept entry of a sibling ambient, A; in this event, the residue of P is $\mathcal{E}\{X \leftarrow A\}$.

(3) $P \xrightarrow{\overline{\mathit{enter}\ n}} (\nu\vec{m})\langle A\rangle Q$: an ambient A within P is ready to enter a sibling ambient n; in this event, process Q is the residue of P.

Exiting an Ambient

(Comm *out*)

$$\frac{}{(\mathit{out}\, n.M).P \xrightarrow{\mathit{out}\, n} M.P}$$

(Comm *exit*)

$$\frac{P \xrightarrow{\mathit{out}\, n} Q}{m[P] \xrightarrow{\mathit{exit}\, n} \langle m[Q]\rangle \mathbf{0}}$$

(Comm τ-*exit*) (where $n \notin \{\vec{m}\}$)

$$\frac{P \xrightarrow{\mathit{exit}\, n} (\nu\vec{m})\langle A\rangle Q}{n[P] \xrightarrow{\tau} (\nu\vec{m})(A \mid n[Q])}$$

Apart from τ-commitments, the rules above allow the derivation of the following judgments:

(4) $P \xrightarrow{\mathit{out}\, n} Q$: process P instructs its enclosing ambient to move out of its parent ambient n.

(5) $P \xrightarrow{\mathit{exit}\, n} (\nu\vec{n})\langle A\rangle Q$: ambient A within P is ready to exit its parent ambient n; in this event, process Q is the residue of P.

Opening an Ambient

(Comm *open*)

$$\frac{}{(\mathit{open}\, n.M).P \xrightarrow{\mathit{open}\, n} M.P}$$

(Comm $\overline{\mathit{open}}$)

$$\frac{}{n[P] \xrightarrow{\overline{\mathit{open}\, n}} P}$$

(Comm τ-*open* 1)

$$\frac{P \xrightarrow{\overline{\mathit{open}\, n}} P' \quad Q \xrightarrow{\mathit{open}\, n} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

(Comm τ-*open* 2)

$$\frac{P \xrightarrow{\mathit{open}\, n} P' \quad Q \xrightarrow{\overline{\mathit{open}\, n}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

Apart from τ-commitments, the rules above allow the derivation of the following judgments:

(6) $P \xrightarrow{\mathit{open}\, n} Q$: process P opens a sibling ambient n.

(7) $P \xrightarrow{\overline{\mathit{open}\, n}} Q$: an ambient n within P is opened.

The following group of rules explains the behavior of processes of the form $\langle M\rangle$ and $(x).P$. As a convenient shorthand for writing (Comm τ-*anon* 1) and (Comm τ-*anon* 2), we define the *interactions*, $F@C$ and $C@F$, of a capability abstraction F and a capability concretion C, as follows:

$$\begin{aligned} ((x)P)@(\nu\vec{m})\langle M\rangle Q &\triangleq (\nu\vec{m})(P\{x \leftarrow M\} \mid Q) && \text{if } \{\vec{m}\} \cap \mathit{fn}(P) = \varnothing \\ ((\nu\vec{m})\langle A\rangle P)@(x)Q &\triangleq (\nu\vec{m})(P \mid Q\{x \leftarrow M\}) && \text{if } \{\vec{m}\} \cap \mathit{fn}(Q) = \varnothing \end{aligned}$$

Anonymous I/O

(Comm *anon*) $\quad$ (Comm $\overline{anon}$)

$$\frac{}{(x).P \xrightarrow{anon\,M} (x)P} \qquad \frac{}{\langle M\rangle \xrightarrow{\overline{anon}} \langle M\rangle\mathbf{0}}$$

(Comm τ-*anon* 1) $\quad$ (Comm τ-*anon* 2)

$$\frac{P \xrightarrow{\overline{anon}} C \quad Q \xrightarrow{anon} F}{P \mid Q \xrightarrow{\tau} C@F} \qquad \frac{P \xrightarrow{anon} F \quad Q \xrightarrow{\overline{anon}} C}{P \mid Q \xrightarrow{\tau} F@C}$$

We can now conclude our enumeration of the nine judgments that represent intermediate stages in the derivation of a τ-commitment:

(8) $P \xrightarrow{anon} (x)Q$: a process within P is ready to input a capability, M; in this event, the residue of P is $Q\{x \leftarrow M\}$.

(9) $P \xrightarrow{\overline{anon}} (\nu\vec{m})\langle M\rangle Q$: a process within P is ready to output capability M; in this event, process Q is the residue of P.

Finally, we complete the definition of the commitment relation with two groups of commitment rules that explain the behavior of processes of the form $\epsilon.P$ and $!P$:

Capability Sequencing

(Comm ϵ)

$$\frac{P \xrightarrow{\alpha} O}{\epsilon.P \xrightarrow{\alpha} O}$$

Replication

(Comm Repl)

$$\frac{P \mid !P \xrightarrow{\alpha} O}{!P \xrightarrow{\alpha} O}$$

By standard techniques, we can relate the reduction semantics of the previous section and the commitment semantics of this section as follows:

Theorem 1 *$P \to Q$ if and only if there is R such that $P \xrightarrow{\tau} R$ and $R \equiv Q$.*

3 Discussion

Our original motivation for defining this commitment relation was to use it as the basis of labeled bisimilarity for the ambient calculus. We encountered various difficulties, in part due to the multitude of different labels used in the definition of commitment. Subsequently, in our paper on testing equivalence for the ambient calculus [3], we employed a different labeled transition system, with rather fewer labels, defined in terms of an auxiliary relation called the hardening relation.

References

[1] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. Technical Report 414, University of Cambridge Computer Laboratory, January 1997. (This version contains detailed proofs omitted from the journal version of the paper.).

[2] L. Cardelli and A. D. Gordon. Mobile ambients. *Theoretical Computer Science*, 240:177–213, 2000.

[3] A. D. Gordon and L. Cardelli. Equational properties of mobile ambients. In *Foundations of Software Science and Computation Structures*, Lecture Notes in Computer Science, pages 212–226. Springer-Verlag, 1999. An extended version appears Microsoft Research Technical Report MSR–TR–99–11.

[4] R. Milner. *Communicating and Mobile Systems: the π-Calculus.* 1999.

[5] D. Sangiorgi. *Expressing Mobility in Process Algebras: First-Order and Higher-Order Paradigms.* PhD thesis, University of Edinburgh, 1992.