

A Calculus of Mobile Ambients

Luca Cardelli

Digital Equipment Corporation

Systems Research Center

Andrew D. Gordon

University of Cambridge

Computer Laboratory

The Network is the Computer

The Internet: a single, flat, global address space?

No, since we want computing to be secure, ubiquitous and personal.

- Firewalls partition the net into regions.
- Mobile devices move from one net connection to another.
- Mobile agents move with data from one host to another.

A notion of **administrative domain** underlies these situations.

Our goal: a calculus to describe secure and mobile administrative domains, and to suggest flexible ways of programming them.

Design Principles

Some goals:

- a small, expressive calculus of primitive ambients
- an ambient has a boundary—no action at a distance
- an ambient may be nested inside another
- an ambient may move as a whole

Some non-goals:

- primitive global channels (but may be encoded)
- a design optimised for distribution or including node failure
- types

So what's the formalism?

Milner, Parrow and Walker's π -calculus is our starting point:

- **centralised**: contiguous group of communicating processes
- **nominal**: has names and name generation
- communication achieved via **shared names**

Instead, we want a nominal calculus of nested, distributed processes, with no action at a distance, and some account of access control.

A Calculus with Mobility Primitives

$P, Q, R ::=$	process
$(\nu n)P$	restriction
0	inactivity
$P \mid Q$	composition
$!P$	replication
$n[P]$	ambient
$M.P$	action
$M ::=$	term
$in\ n$	can enter n
$out\ n$	can exit n
$open\ n$	can open n

General Form of an Ambient

Nested ambients induce a tree-structure. Each node takes the form:

$$n[\overbrace{P_1 \mid \cdots \mid P_q}^{\text{local processes}} \mid \overbrace{m_1[\cdots] \mid \cdots \mid m_p[\cdots]}^{\text{child ambients}}] \quad P_j \neq p_j[\cdots]$$

Local processes **govern** ambient behaviour.

Via suitable capabilities, a local process:

- may move the whole of ambient n into a sibling
- may move the whole of ambient n out of its parent ambient
- dissolve the boundary of one of the children of ambient n

Reduction I: Mobility

Entering an ambient:

$$n[in\ m.P \mid Q] \mid m[R] \rightarrow m[n[P \mid Q] \mid R] \quad (\text{Red In})$$

Exiting an ambient:

$$m[n[out\ m.P \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R] \quad (\text{Red Out})$$

Possession of a capability *in m* or *out m* confers a right of passage across the boundary of **ambient m**.

Passage preserves the boundary of **ambient n**.

Reduction II: Open

Opening an ambient:

$$\textit{open } n.P \mid n[Q] \rightarrow P \mid Q \quad (\text{Red Open})$$

Capability *open n* confers the right to dissolve the boundary of ambient *n*.

Reduction III: CHAM-style Rules

The following rules complete the definition of $P \rightarrow Q$:

$$P \rightarrow Q \Rightarrow P \mid R \rightarrow Q \mid R \quad (\text{Red Par})$$

$$P \rightarrow Q \Rightarrow (\nu n)P \rightarrow (\nu n)Q \quad (\text{Red Res})$$

$$P \rightarrow Q \Rightarrow n[P] \rightarrow n[Q] \quad (\text{Red Amb})$$

$$P' \equiv P, P \rightarrow Q, Q \equiv Q' \Rightarrow P' \rightarrow Q' \quad (\text{Red } \equiv)$$

Structural Equivalence I

$P \equiv Q$ is an equivalence relation satisfying:

$$P \equiv Q \Rightarrow (\nu n)P \equiv (\nu n)Q \quad (\text{Struct Res})$$

$$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R \quad (\text{Struct Par})$$

$$P \mid Q \equiv Q \mid P \quad (\text{Struct Par Comm})$$

$$(P \mid Q) \mid R \equiv P \mid (Q \mid R) \quad (\text{Struct Par Assoc})$$

$$!P \equiv P \mid !P \quad (\text{Struct Repl Par})$$

$$(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P \quad (\text{Struct Res Res})$$

$$n \notin \text{fn}(P) \Rightarrow (\nu n)(P \mid Q) \equiv P \mid (\nu n)Q \quad (\text{Struct Res Par})$$

Structural Equivalence II

Moreover, $P \equiv Q$ satisfies:

$$P \equiv Q \Rightarrow n[P] \equiv n[Q] \quad (\text{Struct Amb})$$

$$n \neq m \Rightarrow (\nu n)m[P] \equiv m[(\nu n)P] \quad (\text{Struct Res Amb})$$

But note that in general:

$$n[P] \mid n[Q] \not\equiv n[P \mid Q]$$

$$!n[P] \not\equiv n[!P]$$

Variation: Subjective versus Objective Moves

Subjective: “I move. I become a child ambient.”

$$\begin{aligned} n[in\ m.P \mid Q] \mid m[R] &\rightarrow m[n[P \mid Q] \mid R] \\ m[n[out\ m.P \mid Q] \mid R] &\rightarrow n[P \mid Q] \mid m[R] \end{aligned}$$

Objective: “I make you move. You become a local process.”

$$\begin{aligned} mv\ in\ n.P \mid n[Q] &\rightarrow n[P \mid Q] \\ n[mv\ out\ n.P \mid Q] &\rightarrow P \mid n[Q] \end{aligned}$$

Variation: Open versus Acid

Ambient acid: “I dissolve my own boundary.”

$$n[\mathit{acid}.P \mid Q] \rightarrow P \mid Q$$

Objective moves derivable:

$$\mathit{mv\ in}\ n.P \stackrel{\Delta}{=} (\forall q)q[\mathit{in}\ n.\mathit{acid}.P]$$

$$\mathit{mv\ out}\ n.P \stackrel{\Delta}{=} (\forall q)q[\mathit{out}\ n.\mathit{acid}.P]$$

But the risk is that acid allows ambient kidnap:

$$\mathit{entrap}\ m \stackrel{\Delta}{=} (\forall k\ q)k[] \mid q[\mathit{in}\ m.\mathit{acid}.\mathit{in}\ k.\mathbf{0}]$$

$$\mathit{entrap}\ m \mid m[P] \rightarrow^* (\forall k)k[m[P]]$$

Ambients as Administrative Domains

An ambient is like a republic with border guards:

- Local processes govern movement of the ambient.
- Disposition of capabilities governs movement across the border.
- Capabilities *in* n and *out* n are like visas.
(But acid or objective moves turn them into charters to govern.)
- Capability *open* n lets the local processes of an ambient decide who may join them.

Encoding Locks

We can use *open* to encode locks:

$$\mathit{acquire} \ n.P \triangleq \mathit{open} \ n.P$$

$$\mathit{release} \ n.P \triangleq n[] \mid P$$

Two processes may synchronise as follows:

$$\mathit{acquire} \ n.\mathit{release} \ m.P \mid \mathit{release} \ n.\mathit{acquire} \ m.Q$$

Encoding Objective Moves I

$$n^\downarrow[P] \triangleq n[P \mid \mathit{allow} \mathit{in} n]$$

$$\mathit{allow} n \triangleq !\mathit{open} n$$

$$\mathit{mv} \mathit{in} n.P \triangleq (\nu k)k[\mathit{in} n.\mathit{in}[\mathit{out} k.P]]$$

We get:

$$\begin{aligned} \mathit{mv} \mathit{in} n.P \mid n^\downarrow[Q] &\rightarrow n^\downarrow[(\nu k)k[\mathit{in}[\mathit{out} k.P]] \mid Q] \\ &\rightarrow n^\downarrow[(\nu k)k[] \mid \mathit{in}[P] \mid Q] \\ &\rightarrow n^\downarrow[(\nu k)k[] \mid P \mid Q] \\ &\approx n^\downarrow[P \mid Q] \end{aligned}$$

Encoding Objective Moves II

Deriving objective exit:

$$\begin{aligned} n^\uparrow[P] &\triangleq n[P] \mid \text{allow out} \\ \text{mv out } n.P &\triangleq (\nu k)k[\text{out } n.\text{out}[\text{out } k.P]] \end{aligned}$$

We get:

$$n^\uparrow[\text{mv out } n.P \mid Q] \rightarrow^* \approx P \mid n^\uparrow[Q]$$

Ambient allowing both objective entry and exit:

$$n^\uparrow\downarrow[P] \triangleq n[P \mid \text{allow in}] \mid \text{allow out}$$

Synchronisation on Named Channels

We represent a named CCS channel by the ambient,

$$n^{\uparrow\downarrow} \square$$

and synchronisation on complementary ports $n?$ and $n!$ by:

$$n?.P \triangleq mv \text{ in } n.\textit{acquire} \text{ rd.}\textit{release} \text{ wr.}\textit{mv out} \text{ n.P}$$

$$n!.P \triangleq mv \text{ in } n.\textit{release} \text{ rd.}\textit{acquire} \text{ wr.}\textit{mv out} \text{ n.P}$$

Encoding Boolean Flags

$$\mathit{flag} \ n \stackrel{\Delta}{=} n \mid$$

$$\mathit{if} \ \mathit{tt} \ P, \ \mathit{if} \ \mathit{ff} \ Q \stackrel{\Delta}{=}$$

$$(\nu k)(k \mid$$

$$\mathit{open} \ \mathit{tt} . \mathit{open} \ k . (\nu t)(\mathit{ff}[t \mid \mid \mathit{open} \ t . P) \mid$$

$$\mathit{open} \ \mathit{ff} . \mathit{open} \ k . (\nu f)(\mathit{tt}[f \mid \mid \mathit{open} \ f . Q))$$

We have:

$$\mathit{flag} \ \mathit{tt} \mid \ \mathit{if} \ \mathit{tt} \ P, \ \mathit{if} \ \mathit{ff} \ Q \ \rightarrow^* \approx P$$

$$\mathit{flag} \ \mathit{ff} \mid \ \mathit{if} \ \mathit{tt} \ P, \ \mathit{if} \ \mathit{ff} \ Q \ \rightarrow^* \approx Q$$

Turing Machines

Idea: tape looks like $\text{end}^{\uparrow\downarrow}[\text{ff}[] \mid \text{sq}^{\uparrow\downarrow}[\text{ff}[] \mid \text{sq}^{\uparrow\downarrow}[\text{ff}[] \mid \text{sq}^{\uparrow\downarrow}[\dots]]]]$.

$\text{head} \triangleq \text{head}^{\uparrow\downarrow}[\text{!open } S_1.\text{mv out head}.$

$\text{if tt}(\text{ff}[] \mid \text{mv in head.in sq.S}_2[]),$

$\text{if ff}(\text{tt}[] \mid \text{mv in head.out sq.S}_3[])$ |

... |

$S_1[]]$

$\text{stretchRht} \rightarrow^* \text{sq}^{\uparrow\downarrow}[\text{ff}[] \mid \text{stretchRht}]$

$\text{machine} \triangleq \text{end}^{\uparrow\downarrow}[\text{ff}[] \mid \text{head} \mid \text{stretchRht}]$

Extensions for I/O and Paths

We include **variables**, x , and **names**, n , **null**, ϵ , and **paths**, $M.M'$, as terms.

We include **inputs**, $(x).P$, and **outputs**, $\langle M \rangle$, as processes.

$$\epsilon.P \equiv P \quad (\text{Struct } \epsilon)$$

$$(M.M').P \equiv M.M'.P \quad (\text{Struct } .)$$

$$\langle M \rangle \mid (x).P \rightarrow P\{x \leftarrow M\} \quad (\text{Red I/O})$$

Modelling Mobile Processes

$$\mathit{tourist} \triangleq (x).\mathit{fred}[x.\mathit{enjoyNY}]$$

$$\mathit{kiosk} \triangleq !\langle \mathit{in} \mathit{QE2UK}.\mathit{out} \mathit{QE2USA} \rangle$$

Southampton[*kiosk* | *tourist* | *QE2UK*[*route*]]

→* Southampton[*kiosk* |
 fred[*in* *QE2UK*.*out* *QE2USA*.*enjoyNY*] |
 QE2UK[*route*]]

→* Southampton[*kiosk* |
 QE2UK[*route* |
 fred[*out* *QE2USA*.*enjoyNY*]]]

Gaining Entry Behind a Firewall

Authentication of the incoming agent depends on a shared secret, k :

$$\mathit{Firewall} \triangleq (\nu w)(k[\mathit{in} k.\mathit{in} w] \mid w[\mathit{open} k.P])$$

$$\mathit{Agent} \triangleq k[\mathit{open} k.C]$$

Any agent gaining entry behind the firewall either knows k , or is helped by someone who does.

$$\begin{aligned} \mathit{Agent} \mid \mathit{Firewall} &\equiv (\nu w)(k[\mathit{open} k.C] \mid k[\mathit{in} k.\mathit{in} w] \mid w[\mathit{open} k.P]) \\ &\rightarrow (\nu w)(k[\mathit{open} k.C \mid k[\mathit{in} w]] \mid w[\mathit{open} k.P]) \\ &\rightarrow (\nu w)(k[C \mid \mathit{in} w] \mid w[\mathit{open} k.P]) \\ &\rightarrow (\nu w)w[k[C] \mid \mathit{open} k.P] \\ &\rightarrow (\nu w)w[C \mid P] \end{aligned}$$

Communication on Named Channels

Encoding the choice-free asynchronous π -calculus:

$$(\mathit{new} \ n)P \triangleq (\nu n)(n \uparrow \square \mid P)$$

$$\bar{n}\langle M \rangle \triangleq \mathit{mv} \ \mathit{in} \ n.\langle M \rangle$$

$$n(x).P \triangleq \mathit{mv} \ \mathit{in} \ n.(x).\mathit{mv} \ \mathit{out} \ n.P$$

Hence (and also directly) we can encode the λ -calculus.

Encoding the Spi Calculus

Abadi and Gordon's spi calculus is a value-passing π -calculus.

Names play the role of encryption keys; restriction models secrecy.

A value, V , either an S-expression or a ciphertext, may be coded as an ambient \bar{V} :

$$\begin{aligned} \overline{\#n} &\triangleq \langle n \rangle \\ \overline{(V_1, V_2)} &\triangleq \text{fst}^\downarrow[\bar{V}_1] \mid \text{snd}^\downarrow[\bar{V}_2] \\ \overline{\{V\}_K} &\triangleq K[\bar{V}] \end{aligned}$$

Encoding destructors (projection, decryption) is easy.

Basic Theory of Ambients

Reduction characterised by an LTS: $P \rightarrow Q$ iff $P \xrightarrow{\tau} \equiv Q$.

Morris-style **contextual equivalence**:

$$P \downarrow n \stackrel{\Delta}{=} \exists P_1, P_2, \vec{m} (P \equiv (\nu \vec{m})(P_1 \mid n[P_2]) \ \& \ n \notin \{\vec{m}\})$$

$$P \Downarrow \stackrel{\Delta}{=} \exists Q, n (P \rightarrow^* Q \ \& \ Q \downarrow n)$$

$$P \approx Q \stackrel{\Delta}{=} \forall \text{contexts } \mathcal{C} (\mathcal{C}[P] \Downarrow \Leftrightarrow \mathcal{C}[Q] \Downarrow)$$

The **Perfect Firewall Equation**: $(\nu n)n[P] \approx \mathbf{0}$, if $n \notin \text{fn}(P)$.

Some Related Formalisms

- The π -calculus (Milner, Parrow, Walker)
- Localities and failures (Amadio, Prasad)
- The distributed join calculus (Fournet, Gonthier, et al)
- Locality based Linda (De Nicola, Ferrari, Pugliese)
- λ dist (Sekiguchi, Yonezawa)
- Distributed π -calculus (Sewell)
- The spi calculus (Abadi, Gordon)

Summary

Ambients are hierarchical, mobile, administrative domains, with no action at a distance.

Ambients incorporate both **classical formalisms** (TMs, π , λ) and **net-centric concepts** (distributed devices, packets, mobile code, encryption).

Theory based on both CHAM and LTS paradigms.

An implementation is in progress (Cardelli).