The Kell Calculus

A Family of Higher-Order Distributed Process Calculi

MYTHS/MIKADO/DART Meeting

Alan Schmitt
Jean-Bernard Stefani

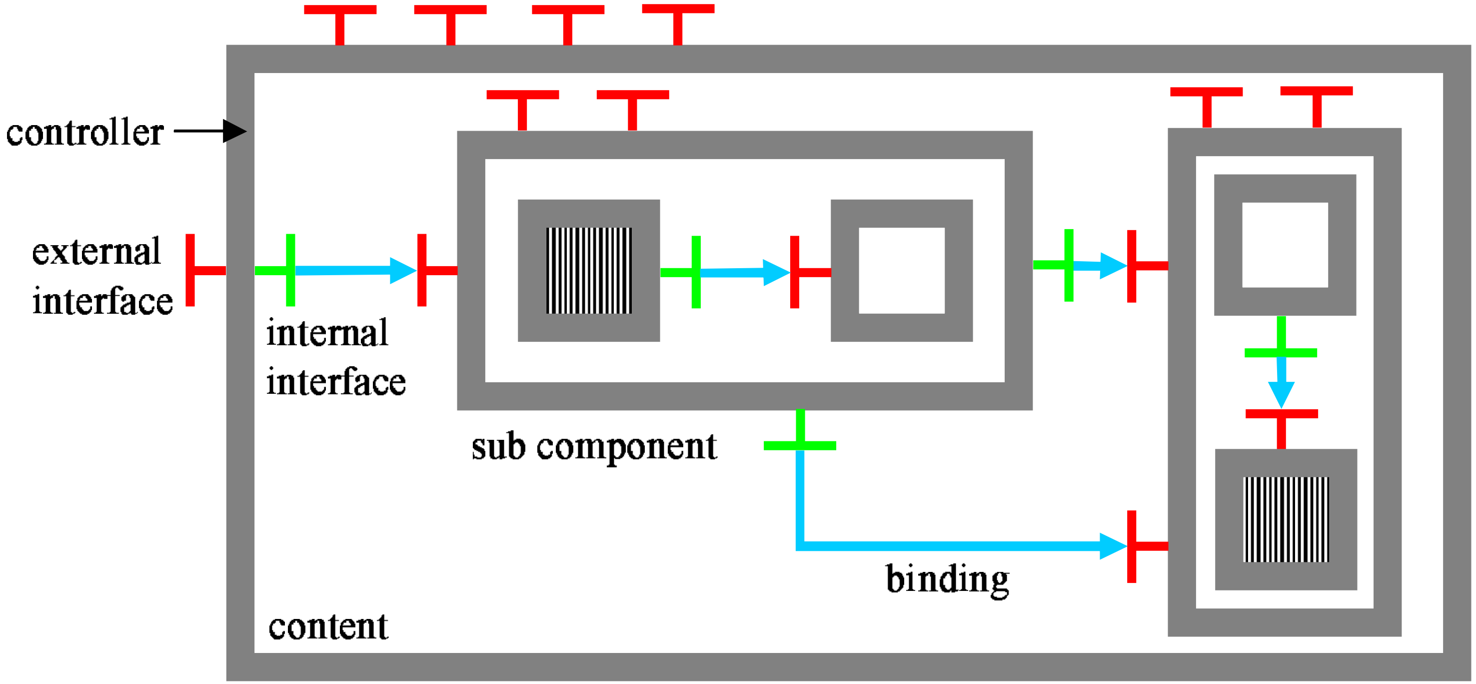
Introduction

- Calculus motivated by work in the Sardes project
- Goal: to model and simulate component-based programs and their environment
- Why the environment?
  - to model resource access and monitoring
  - to model different modes of failure

Outline

- Design Choices for a Component Modelling Calculus
- The Calculus and some Examples
- Equivalences

A component


controller
external
interface
internal
interface
sub component
binding
content

What we want to model

Fractal (http://fractal.objectweb.org)

- Hierarchical components
- Dynamic component deployment and failure
- Dynamic interface binding between components
- Messaging through bound interfaces
- Control capabilities

Why we want to model

- Play the role of a precise and formal semantics
  - Abstract machines
  - Implementations
- Build some verification tools

  Static Type systems, static analyses
  - Component binding
  - Checking dependencies
  - Equivalent components

  Dynamic Correct code instrumentation for
  - security properties
  - fault detection
  - causality and resource monitoring

Design Principles

- π-calculus core
  - Parameterized on the input patterns
- Hiearchical localities (Kells)
  - Encapsulation
- Local actions
  - Tradeoff between implementation and of usability
  - Atomicity decisions left to programmer
  - Dynamic binding
- Higher-order communication and locality passivation
  - To model deployment, migration, and different failure modes
- Programmable membranes
  - To model control features and network failure

Related work

- First order π-calculus with localities and migration primitives (D-Join, Dπ, Nomadic Pict, Seal, . . .)
- Mobile Ambients and variants
- Distributed higher-order calculi
  - Facile, CHOCS, higher-order Dπ, Klaim, M-calculus

Kell-calculus: simplification of the M-calculus:

- No routing rules built in
- Simpler localities

Outline

- Design Choices for Component Modelling Calculus
- The Calculus and some Examples
- Equivalences

Syntax

$$P, Q ::= \mathbf{0} \quad | \quad P \mid Q \quad | \quad \nu a.P$$

- π calculus core

Syntax

$$P, Q ::= \mathbf{0} \quad | \quad P \mid Q \quad | \quad \nu a.P \quad | \quad x$$
$$| \quad a\langle P \rangle.Q \quad | \quad a\,[P]\,.Q$$

- π calculus core
- Higher-order output

Syntax

$$
\begin{aligned}
P, Q ::= \mathbf{0} \quad &| \quad P \mid Q \quad | \quad \nu a.P \quad | \quad x \\
&| \quad a\langle P\rangle.Q \quad | \quad a\,[P]\,.Q \\
&| \quad (\xi \triangleright P)
\end{aligned}
$$

- π calculus core
- Higher-order output
- Input parameterized by patterns ξ

Syntax

$$
\begin{aligned}
P, Q ::= \mathbf{0} \quad | \quad P \mid Q \quad | \quad \nu a.P \quad | \quad x \\
| \quad a\langle P\rangle.Q \quad | \quad a\,[P]\,.Q \\
| \quad (\xi \triangleright P)
\end{aligned}
$$

- π calculus core
- Higher-order output
- Input parameterized by patterns ξ
- Simplest patterns (jK):

$$
\xi ::= \xi_k \quad | \quad M \quad | \quad M \mid \xi_k \qquad M ::= \xi_m \quad | \quad \xi^{\downarrow} \quad | \quad \xi^{\uparrow} \quad | \quad M \mid M
$$

$$
\xi_k ::= a\,[x] \qquad \xi_m ::= a\langle x\rangle \qquad \xi^{\downarrow} ::= a\langle x\rangle^{\downarrow} \qquad \xi^{\uparrow} ::= a\langle x\rangle^{\uparrow}
$$

Reduction Examples

$$a\langle Q\rangle.T \mid (a\langle x\rangle \triangleright P) \longrightarrow T \mid P\{Q/x\}$$

$$a\langle Q\rangle.T \mid b\left[(a\langle x\rangle^{\uparrow} \triangleright P)\right].S \longrightarrow T \mid b\left[P\{Q/x\}\right].S$$

$$b\left[a\langle Q\rangle.T \mid R\right].S \mid (a\langle x\rangle^{\downarrow} \triangleright P) \longrightarrow b\left[T \mid R\right].S \mid P\{Q/x\}$$

$$a\left[Q\right].T \mid (a\langle x\rangle \triangleright P) \longrightarrow T \mid P\{Q/x\}$$

Join patterns

$$a \begin{bmatrix} (d\langle x\rangle^{\downarrow} \mid u\langle y\rangle^{\uparrow} \mid b\,[z] \triangleright x \mid y \mid z) \\ c\,[d\langle P_d\rangle.Q_d]\,.Q_c \\ b\,[P_b]\,.Q_b \end{bmatrix} .Q_a \quad \Bigg| \quad u\langle P_u\rangle.Q_u \longrightarrow$$

$$a \begin{bmatrix} P_d \mid P_u \mid P_b \\ c\,[Q_d]\,.Q_c \\ Q_b \end{bmatrix} .Q_a \quad \Bigg| \quad Q_u$$

Join patterns

$$a\left[\begin{array}{c}(d\langle x\rangle^{\downarrow} \mid u\langle y\rangle^{\uparrow} \mid b\,[z] \triangleright x \mid y \mid z)\\ c\,[d\langle P_d\rangle.Q_d]\,.Q_c\\ b\,[P_b]\,.Q_b\end{array}\right].Q_a \;\Bigg|\; u\langle P_u\rangle.Q_u \longrightarrow$$

$$a\left[\begin{array}{c}P_d \mid P_u \mid P_b\\ c\,[Q_d]\,.Q_c\\ Q_b\end{array}\right].Q_a \;\Bigg|\; Q_u$$

Encoding recursion

$$(\xi \diamond P) \triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle$$

Assume that t and x are fresh in ξ, P, Q, and P', and that $(\xi \triangleright P) \mid Q \longrightarrow P'$

$$(\xi \diamond P) \mid Q \triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle \quad \mid \quad Q$$

Encoding recursion

$$(\xi \diamond P) \triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle$$

Assume that t and x are fresh in ξ, P, Q, and P', and that $(\xi \triangleright P) \mid Q \longrightarrow P'$

$$(\xi \diamond P) \mid Q \triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle \quad \mid \quad Q$$

Encoding recursion

$$(\xi \diamond P) \triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle$$

Assume that t and x are fresh in ξ, P, Q, and P', and that $(\xi \triangleright P) \mid Q \longrightarrow P'$

$$\begin{aligned}(\xi \diamond P) \mid Q &\triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle \quad \mid \quad Q \\ &\longrightarrow \nu t.P' \mid (\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \mid t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle\end{aligned}$$

Encoding recursion

$$(\xi \diamond P) \triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle$$

Assume that t and x are fresh in ξ, P, Q, and P', and that $(\xi \triangleright P) \mid Q \longrightarrow P'$

$$\begin{aligned}(\xi \diamond P) \mid Q &\triangleq \nu t.(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \quad \mid \quad t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle \quad \mid \quad Q \\ &\longrightarrow \nu t.P' \mid (\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle) \mid t\langle(\xi \mid t\langle x\rangle \triangleright P \mid x \mid t\langle x\rangle)\rangle \\ &\triangleq (\xi \diamond P) \mid P'\end{aligned}$$

Using passivation

- A kell $a\,[P]$ is both an evaluation context and a resource
- One may
  - freeze a kell in a message: $(a\,[x] \triangleright a\langle x\rangle)$
  - destroy a kell: $(a\,[x] \triangleright \mathbf{0})$
  - copy and rename a kell: $(a\,[x] \triangleright a\,[x] \mid b\,[x])$
  - insert new content into a kell: $(a\,[x] \triangleright a\,[x \mid b\,[P]])$

Matching and Parametric Patterns

- Generic matching
  - Outer shape of patterns fixed (Local Action)
  - Join patterns built in

$$\mathtt{match}(\xi \mid \xi', M \mid M') = \mathtt{match}(\xi, M) \oplus \mathtt{match}(\xi', M')$$

$$\mathtt{match}(\xi_m, a\langle P\rangle) = \mathtt{match}_m(\xi_m, a\langle P\rangle)$$

$$\mathtt{match}(\xi^{\downarrow}, a\langle P\rangle^{\downarrow_b}) = \mathtt{match}^{\downarrow}(\xi^{\downarrow}, a\langle P\rangle^{\downarrow_b})$$

$$\mathtt{match}(\xi^{\uparrow}, a\langle P\rangle^{\uparrow_b}) = \mathtt{match}^{\uparrow}(\xi^{\uparrow}, a\langle P\rangle^{\uparrow_b})$$

$$\mathtt{match}(\xi_k, a\,[P]) = \mathtt{match}_k(\xi_k, a\,[P])$$

- Instantiation with jK patterns

$$\mathtt{match}_m(a\langle x\rangle, a\langle P\rangle) \triangleq \{^{P}/_{x}\} \qquad \mathtt{match}^{\downarrow}(a\langle x\rangle,^{\downarrow} a\langle P\rangle^{\downarrow_b}) \triangleq \{^{P}/_{x}\}$$

$$\mathtt{match}^{\uparrow}(a\langle x\rangle,^{\uparrow} a\langle P\rangle^{\uparrow_b}) \triangleq \{^{P}/_{x}\} \qquad \mathtt{match}_k(a\,[x]\,, a\,[P]) \triangleq \{^{P}/_{x}\}$$

Outline

- Design Choices for Component Modelling Calculus
- The Calculus and some Examples
- Equivalences

Context Bisimulation: a Tutorial

In the setting of the Higher-order π-calculus:

- An input evolves to an abstraction: $a(X).P \xrightarrow{a} (X).P = F$
- An output evolves to a concretion: $a\langle P_1\rangle P_2 \xrightarrow{\overline{a}} \langle P_1\rangle P_2 = C$
- They communicate: $a(X).P \mid a\langle P_1\rangle P_2 \xrightarrow{\tau} F@C = P\{^{P_1}/_X\} \mid P_2$

Context Bisimulation: a Tutorial

In the setting of the Higher-order π-calculus:

- An input evolves to an abstraction: $a(X).P \xrightarrow{a} (X).P = F$
- An output evolves to a concretion: $a\langle P_1\rangle P_2 \xrightarrow{\overline{a}} \langle P_1\rangle P_2 = C$
- They communicate: $a(X).P \mid a\langle P_1\rangle P_2 \xrightarrow{\tau} F@C = P\{^{P_1}/_X\} \mid P_2$

The relation $\mathcal{R}$ is a (early) context simulation iff $P \mathrel{\mathcal{R}} Q$ implies

- For all $P \xrightarrow{\tau} P'$, there exists Q' such that $Q \xrightarrow{\tau} Q'$ and $P' \mathrel{\mathcal{R}} Q'$;
- For all $P \xrightarrow{a} F$ and for all C, there exists G such that $Q \xrightarrow{a} G$ and $F@C \mathrel{\mathcal{R}} G@C$;
- For all $P \xrightarrow{\overline{a}} C$ and for all F, there exists D such that $Q \xrightarrow{\overline{a}} D$ and $F@C \mathrel{\mathcal{R}} F@D$.

Context Bisimulation for the Kell-calculus

Approach similar to the Higher-order π calculus

Abstractions We need to remember the whole pattern

- join patterns
- message source (local, up, down) or nature (message, kell)
- $(\xi \triangleright P) \xrightarrow{\alpha} (\xi)P$

Concretions We need to make sure that every case of message source is covered (see next slide)

- $a\langle P\rangle.Q \xrightarrow{a} a\langle P\rangle \parallel Q$

Congruence properties are harder to prove, as some processes in concretions are also in evaluation context

What labels?

- Complex labels and concretions, but simple bisimulations

$$\begin{array}{rcl} & \xrightarrow{a} & a\langle P\rangle \parallel Q = C_1 \text{ and } F@C_1 \\ a\langle P\rangle.Q & \xrightarrow{a^{\downarrow b}} & a\langle P\rangle^{\downarrow b} \parallel Q = C_2 \text{ and } F@C_2 \\ & \xrightarrow{a^{\uparrow b}} & a\langle P\rangle^{\uparrow b} \parallel Q = C_3 \text{ and } F@C_3 \end{array}$$

- Simple labels and concretions, but complex bisimulations

$$\begin{array}{rcll} & & & \text{and } F@C \\ a\langle P\rangle.Q & \xrightarrow{a} & a\langle P\rangle \parallel Q = C & \text{and } F@b\,[C] \\ & & & \text{and } b\,[F]\,@C \end{array}$$

- Our current choice: very simple labels (sets of names)

Observables

Like labels, observables $\downarrow_a$ are very simple:

$$P \downarrow_{\overline{a}} \quad \text{iff} \quad \begin{array}{ll} P \equiv \nu\widetilde{c}.a\,[P_a]\,.Q_a \mid Q & \text{with } a \notin \widetilde{c} \\ \text{or } P \equiv \nu\widetilde{c}.a\langle P_a\rangle.Q_a \mid Q & \text{with } a \notin \widetilde{c} \\ \text{or } P \equiv \nu\widetilde{c}.b\,[a\langle P_a\rangle.Q_a \mid P_b]\,.Q_b \mid Q & \text{with } a \notin \widetilde{c} \end{array}$$

$$P \downarrow_{\xi.\mathsf{sk}} \quad \text{iff} \quad \begin{array}{ll} P \equiv \nu\widetilde{c}.(\xi \triangleright Q) \mid R & \text{with } \xi.\mathsf{sk} \cap \widetilde{c} = \emptyset \\ \text{or } P \equiv \nu\widetilde{c}.b\,[(\xi \triangleright Q) \mid P_b]\,.Q_b \mid R & \text{with } \xi.\mathsf{sk} \cap \widetilde{c} = \emptyset \end{array}$$

ξ.sk is the multiset on names used for input. For instance:

$$a\langle P\rangle.\mathsf{sk} = a\langle P\rangle^{\downarrow}.\mathsf{sk} = a\langle P\rangle^{\uparrow}.\mathsf{sk} = a\,[P]\,.\mathsf{sk} = a$$

$$(M \mid M').\mathsf{sk} = M.\mathsf{sk} \mid M'.\mathsf{sk}$$

Theorems

- Strong context bisimilarity $\sim^c$ is based on the LTS $\xrightarrow{\alpha}$
- Strong barbed bisimilarity $\sim_b$ is based on the reduction $\longrightarrow$ and a definition for observables

We have:

- For all P and Q, $P \xrightarrow{\tau} \equiv Q$ iff $P \longrightarrow Q$.
- Under some conditions for the pattern languages (matching may not distinguish bisimilar messages), $\sim^c$ is a congruence.
- If the pattern language also contains the jK simple patterns, the largest congruence included in $\sim_b$ coincides with $\sim^c$.

Technical details in LNCS volume on Global Computing 2004

Current and Future work

- Equivalences
  - Tractable Bisimulations (no universal quantification on concretions and abstractions)
  - Weak approach
- Type systems
  - Inspired by the M-calculus and Dπ type systems
- Testing the calculus expressivity
  - Complete modelisation of Fractal
  - Application to Dream (http://dream.objectweb.org)
- Locality sharing
  - In Fractal, a component may have more than one parent
  - Very useful feature to represent shared resources
  - Joint work with ENS Lyon

Bonus slide: Complex patterns

$$\xi ::= J \quad | \quad \xi_k \quad | \quad J \mid \xi_k$$

$$J ::= \xi_m \quad | \quad \xi^{\downarrow} \quad | \quad \xi^{\uparrow} \quad | \quad J \mid J$$

$$\xi_m ::= a\langle\overline{\rho}\rangle$$

$$\xi^{\uparrow} ::= a\langle\overline{\rho}\rangle^{\uparrow}$$

$$\xi^{\downarrow} ::= a\langle\overline{\rho}\rangle^{\downarrow}$$

$$\xi_k ::= a\,[x]$$

$$\rho ::= a\langle\overline{\rho}\rangle \quad | \quad \rho \mid \rho$$

$$\overline{\rho} ::= x \quad | \quad \rho \quad | \quad (a)\langle\overline{\rho}\rangle \quad | \quad \overline{a}\langle\overline{\rho}\rangle \quad | \quad ((m) \neq a)\langle\overline{\rho}\rangle \quad | \quad \_$$