Towards a Theory of Glue

Simon Bliudze

Rigorous System Design Laboratory
EPFL IC, Station 14, CH-1015, Lausanne, Switzerland
Simon.Bliudze@epfl.ch

Interaction and Concurrency Experience 2012
Stockholm, June 16th, 2012
Outline

- Why behaviour types?
- Definition
- Co-algebraic example
- Some speculation about glue
Outline

- Why behaviour types?
- Definition
- Co-algebraic example
- Some speculation about glue
Component-based design

**Components:** semantic behaviour domain $\mathcal{B}$ (objects, actors, machines, CORBA components, JavaBeans, threads, etc.)

**Composition:** operators $\mathcal{B}^n \rightarrow \mathcal{B}$ or $2^\mathcal{B} \rightarrow \mathcal{B}$
Comparing component-based frameworks

How do we compare two component-based frameworks $C_1, C_2$?

**Common approach:**
- What behaviours can be obtained in a given model?

\[
C_1(2^B) \xrightarrow{?} C_2(2^B)
\]

- Not satisfactory:
  - $id \in C$ implies $B \subseteq C(2^B)$
  - $C(2^B) \subseteq B$
  - Hence, $C(2^B) = B$
  - Most “behaviour” models are Turing complete.
Example: Strong Expressiveness Preorder

From Bliudze & Sifakis, “A Notion of Glue Expressiveness for Component-Based Systems”. In CONCUR 2008.

\[
\begin{align*}
\mathcal{C}_1 \leq \mathcal{C}_2 & \iff \forall g_1 \in \mathcal{C}_1, \forall B \subset B, \exists g_2 \in \mathcal{C}_2 : g_1(B) \simeq g_2(B) \\
\end{align*}
\]

What if behaviour domains are not the same? \((\mathcal{B}_1 \neq \mathcal{B}_2)\)
Separation of concerns

- Encapsulating features into different entities
- Layers (e.g. network protocol stack, hardware abstraction)
- Base functionality vs.
  - component life-cycle management
  - introspection
  - non-functional behaviour
- Sequential computation vs. (memoryless) coordination
Memoryless coordination intuitively

Non-deterministic choice:

\[
\begin{align*}
P & \rightarrow P' \\
Q & \rightarrow Q'
\end{align*}
\]

It is important to distinguish from interleaving

\[
\begin{align*}
P & \rightarrow P' \\
Q & \rightarrow Q' \\
\Rightarrow \\
P \mid Q & \rightarrow P' \mid Q
\end{align*}
\]

Prefixing:

\[
a \rightarrow
\]
Outline

- Why behaviour types?
- Definition
- Co-algebraic example
- Some speculation about glue
Understanding the types of behaviour

- What is the behaviour of several parallel components without any coordination constraints?
- How does one compare two behaviours?
- How does one model the simultaneous application of several composition operators?
Parallel composition

\((B, \|, 0)\) — a monoid (associative, not necessarily commutative).

**Traces**

\[ B = (A, T), \text{ where } A \subseteq A \text{ and } T \subseteq A^* \]

\[ 0 = (\emptyset, \{\varepsilon\}) \]

\[ B_1 \| B_2 = (A_1 \cup A_2, T_{12}) : \]

\[ 0 \Rightarrow 0 \]

**Labelled Transition Systems**

\[ B = (Q, A, \rightarrow), \text{ where } Q \text{ is finite, } A \subseteq A, \rightarrow \subseteq Q \times 2^A \times Q \]

\[ 0 = (\{\ast\}, \emptyset, \emptyset) \]

\(A\) is a universal set of actions
Comparing behaviours
Property satisfaction, specification conformance

**Simulation preorder:** $\sqsubseteq \subseteq B \times B$, s.t.
- it is preserved by $\parallel (B_2 \sqsubseteq B_3 \Rightarrow B_1 \parallel B_2 \sqsubseteq B_1 \parallel B_3)$,
- $0 \sqsubseteq B$.

(Trace containment, simulation — not necessarily pre-congruences.)

**Traces:** $B = (A, T), \ T \subseteq A^*$

$B_1 \sqsubseteq B_2 \iff A_1 \subseteq A_2 \land T_1 \subseteq T_2 \rangle_{A_1}$

$(T \rangle_{A} \dashv \text{ subsequences in } A^*)$

**Labelled Transition Systems:** $B = (Q, A, \rightarrow), \ \rightarrow \subseteq Q \times 2^A \times Q$

Consider the maximal relation $R \subseteq Q_1 \times Q_2$ s.t.

$q_1 R q_2 \Rightarrow \forall q_1 \xrightarrow{a} q'_1, \ \exists q'_2 \in Q_2, b \subseteq A_2 : \left( q_2 \xrightarrow{b} q'_2 \land a \subseteq b \land q'_1 R q'_2 \right)$

$B_1 \sqsubseteq B_2 \iff (A_1 \subseteq A_2) \land (R \text{ is total on } Q_1)$
Composition as constraints

Dataflow:

Control:

Concurrency coordination:

\[ a = c \]
\[ b \neq c \]
\[ \sim \sim \]
Comparing behaviours & Composition operators
Component substitutability

**Composition operators:** \( f : B^n \rightarrow B \)
- \( f(B_1, \ldots, B_n) \sqsubseteq B_1 \parallel \cdots \parallel B_n, \)
- \( B_i \preceq \tilde{B} \Rightarrow f(B_1, \ldots, B_i, \ldots, B_n) \preceq f(B_1, \ldots, \tilde{B}, \ldots, B_n). \)

**Semantic preorder:** \( \preceq \subseteq B \times B \) s.t.
- \( \preceq \) is preserved by \( \parallel \) (i.e. \( B_2 \preceq B_3 \Rightarrow B_1 \parallel B_2 \preceq B_1 \parallel B_3 \))

(Ready simulation, bisimulation relations — precongruences.)

**Meet operator:** \( \otimes : B \times B \rightarrow B \) s.t.
- \( (B/\simeq, \preceq, \otimes), \) with \( \simeq \overset{\text{def}}{=} \preceq \cap \preceq^{-1}, \) is a meet-semilattice.

**Traces, LTS**
\( \preceq \) = \( \subseteq. \) (Another example below.)
Software architectures

Components: 

\[ \{B_i\}_{i=1}^4 \]

\[
f_1(B_1, B_2, B_3, B_4)
\]

\[
f_2(B_1, B_2, B_3, B_4)
\]

\[
(f_1 \otimes f_2)(B_1, B_2, B_3, B_4)
\]
**Parallel composition**

\((\mathcal{B}, \|, 0)\) — a monoid (associative, not necessarily commutative).

**Simulation preorder**

\(\sqsubseteq \subseteq \mathcal{B} \times \mathcal{B}\), s.t.

- it is preserved by \(\|\) \((\mathcal{B}_2 \sqsubseteq \mathcal{B}_3 \Rightarrow \mathcal{B}_1 \| \mathcal{B}_2 \sqsubseteq \mathcal{B}_1 \| \mathcal{B}_3)\),
- \(0 \sqsubseteq \mathcal{B}\).

**Semantic preorder & meet operator**

\(\preceq \subseteq \mathcal{B} \times \mathcal{B}\) and \(\otimes : \mathcal{B} \times \mathcal{B} \rightarrow \mathcal{B}\), s.t.

- \(\preceq\) is preserved by \(\|\) \((\mathcal{B}_2 \preceq \mathcal{B}_3 \Rightarrow \mathcal{B}_1 \| \mathcal{B}_2 \preceq \mathcal{B}_1 \| \mathcal{B}_3)\)
- \((\mathcal{B}/\simeq, \preceq, \otimes)\), with \(\simeq \overset{\text{def}}{=} \preceq \cap \preceq^{-1}\), is a meet-semilattice.
Outline

- Why behaviour types?
- Definition
- Co-algebraic example
- Some speculation about glue
Deterministic functor coalgebras

\( F \)-coalgebras:
\((S, f : S \to F(S)), \) where \( S \) — state-space, \( f \) — transition structure

Some examples:
- \( M = (B \times \text{Id})^A \) input-enabled Mealy machines
- \( D = B \times (1 + \text{Id})^A \) deterministic automata (no initial state)
- \( N = B^2 \times \mathcal{P}_\omega(\text{Id})^A \) non-deterministic automata

Coalgebra homomorphism:
\[
\begin{align*}
S_1 \xrightarrow{h} & S_2 \\
F(S_1) \xrightarrow{F(h)} & F(S_2)
\end{align*}
\]

Coalgebra bisimulation:
\[
\begin{align*}
S_1 \xleftarrow{\pi_1} & R \xrightarrow{\pi_2} S_2 \\
F(S_1) \xleftarrow{F(\pi_1)} & F(R) \xrightarrow{F(\pi_2)} F(S_2)
\end{align*}
\]

Non-Deterministic functors:
\( F ::= \text{Id} \mid B \mid F \times F \mid F^A \mid F + F \mid \mathcal{P}_\omega^*(F) \) (\( B \) is a join-semilattice with \( \perp \))
Example: DF-coalgebra-based behaviour type (1/3)

Parallel composition & zero behaviour

\[(S_1, f_1) \parallel (S_2, f_2) \overset{\text{def}}{=} (S_1 \times S_2, (s_1, s_2) \mapsto f_1(s_1) \boxtimes_{S_1, S_2} f_2(s_2))\]

\[x \boxtimes_{X, Y} y \overset{\text{def}}{=} \begin{cases} 
(x, y), & F = \text{Id}, \\
\downarrow, & F = \text{B}, \\
(x_1 \boxtimes_{X, Y} y_1, x_2 \boxtimes_{X, Y} y_2), & F = F_1 \times F_2, x = (x_1, x_2), y = (y_1, y_2), \\
\lambda a. \left(x(a) \boxtimes_{X, Y} y(a)\right), & F = G^A.
\end{cases}\]

\[0_F = (1, f_F^0) \quad f_F^0(*) \overset{\text{def}}{=} \begin{cases} 
*, & F = \text{Id}, \\
\perp, & F = \text{B}, \\
(f_F^0(*), f_F^0(*)), & F = F_1 \times F_2, \\
\lambda a.f_G^0(*), & F = G^A.
\end{cases}\]
Example: DF-coalgebra-based behaviour type (2/3)

Simulation preorder

\[ x \leq_F^R y \iff \begin{cases} 
(x, y) \in R, & F = \text{Id}, \\
x \lor y = y, & F = \text{B}, \\
x_1 \leq_{F_1}^R y_1 \land x_2 \leq_{F_2}^R y_2, & F = F_1 \times F_2, x = (x_1, x_2), y = (y_1, y_2), \\
\forall a \in A, \ x(a) \leq_{G}^R y(a), & F = G^A, 
\end{cases} \]

\[(S_1, f_1) \sqsubseteq (S_2, f_2) \iff \exists R \subseteq S_1 \times S_2 : \text{total on } S_1 \text{ and s.t. } \forall (s_1, s_2) \in R, f_1(s_1) \leq_{R}^F f_2(s_2)\]

Semantic preorder & meet operator

\[(S_1, f_1) \preceq (S_2, f_2) \iff \exists R \subseteq S_1 \times S_2 : \text{bisimulation total on } S_1\]

\[(S_1, f_1) \otimes (S_2, f_2) \overset{\text{def}}{=} (R, g) \text{ — maximal bisimulation (g unique for DF)}\]
**Proposition:** \((DF/\cong, \preceq, \otimes)\) is a meet-semilattice.

**Proof** — Let \((R, g) = (S_1, f_1) \otimes (S_2, f_2)\) and \((S, f) \preceq (S_1, f_1), (S_2, f_2)\)

\[
\begin{array}{c}
(T, h) 
\downarrow

(R_1, g_1) 
\downarrow 
(S_1, f_1) 
\uparrow

(R_2, g_2) 
\downarrow

(S, f) 
\uparrow

(S_2, f_2) 
\leftarrow
(S, f) 
\leftarrow (R, g)
\end{array}
\]

1. \(T = \{(s', s_1, s'', s_2) \mid (s', s_1) \in R_1, (s'', s_2) \in R_2\text{ and } s' = s''\}\).
2. Projection \(T_{2,4} \subseteq R\).
3. \(T \cong \{(s, s_1, s_2) \mid (s, s_1) \in R_1, (s, s_2) \in R_2\} — \text{bisimulation on } (S, f)\) and \((R, g)\) total on \(S\).
Outline

- Why behaviour types?
- Definition
- Co-algebraic example
- Some speculation about glue
Memoryless glue operators

Separation of concerns: separating the computation of a system from the application of glue operators coordinating its atomic components.

1. A glue operator should not add state to the coordinated system.
2. The actions possible in a global state of the composed system should be completely determined by the properties of the corresponding states of the constituent subsystems.

Glue operators: Let \((B, \ldots)\) be a behaviour type over \(F\)-coalgebras. Composition operator \(gl : B^n \to B\) is a glue operator iff there exists a natural transformation \(sync : F^n \to F\), such that, for any \(\{B_i = (S_i, f_i)\}_{i=1}^n\), \(gl(B_1, \ldots, B_n) = (S, f)\) with

1. \[ S = \prod_{i=1}^{n} S_i, \]
2. \[ f(s) = sync(f_1(s_1), \ldots, f_n(s_n)), \text{ for all } s = (s_1, \ldots, s_n) \in S. \]
Choice operator: \((S_1, f_1) + (S_2, f_2) \overset{\text{def}}{=} (S_1 + S_2, f_1 + f_2)\).

Hypothesis: There is no glue operator \(gl\), such that \(gl \simeq +\).

Consider something like \((\{q_1\}, \{q_1 \overset{a}{\rightarrow} q_1\}\}) and \((\{q_2\}, \{q_2 \overset{b}{\rightarrow} q_2\}\})."
Some speculation

Choice hypothesis

**Choice operator:** $(S_1, f_1) + (S_2, f_2) \overset{\text{def}}{=} (S_1 + S_2, f_1 + f_2)$.

**Hypothesis:** There is no glue operator $gl$, such that $gl \simeq +$.

// Consider something like $(\{q_1\}, \{q_1 \xrightarrow{a} q_1\})$ and $(\{q_2\}, \{q_1 \xrightarrow{b} q_2\})$. //
Some speculation
Choice hypothesis

**Choice operator:** \((S_1, f_1) + (S_2, f_2) \stackrel{\text{def}}{=} (S_1 + S_2, f_1 + f_2)\).

**Hypothesis:** There is no glue operator \(g\), such that \(g \simeq +\).

Consider something like \(\left(\{q_1\}, \{q_1 \xrightarrow{a} q_1\}\right)\) and \(\left(\{q_2\}, \{q_1 \xrightarrow{b} q_2\}\right)\).
Some speculation
Choice hypothesis

**Choice operator:** \((S_1, f_1) + (S_2, f_2) \overset{\text{def}}{=} (S_1 + S_2, f_1 + f_2)\).

**Hypothesis:** There is no glue operator \(g\), such that \(g \simeq +\).

Consider something like \((\{q_1\}, \{q_1 \overset{a}{\rightarrow} q_1\})\) and \((\{q_2\}, \{q_1 \overset{b}{\rightarrow} q_2\})\).
Some speculation
Choice hypothesis

**Choice operator:** \((S_1, f_1) + (S_2, f_2) \overset{\text{def}}{=} (S_1 + S_2, f_1 + f_2)\).

**Hypothesis:** There is no glue operator \(g\), such that \(g \simeq +\).

Consider something like \((\{q_1\}, \{q_1 \xrightarrow{a} q_1\})\) and \((\{q_2\}, \{q_1 \xrightarrow{b} q_2\})\).

**Hypothesis 2:** For any glue operator \(g\),

\[g(B_1, B_2) \simeq B_1 + B_2 \implies B_1 \simeq B_2.\]