

# Towards GCAB in Coq

Jean-Bernard Stefani

joint work with C. Di Giusto and A. Schmitt

INRIA

- 1 Motivations
- 2 CAB: A process calculus interpretation of BIP
- 3 GCAB: Towards reconciling BIP and Fractal

## 1 Motivations

## 2 CAB: A process calculus interpretation of BIP

## 3 GCAB: Towards reconciling BIP and Fractal

- ① Understanding various software engineering / programming structures: components, features, aspects
  - forms of modularity
  - structures with sharing
- ② Programming model including dynamic modularity with preemption
  - isolation example

$$\text{SafePluginReceipt} = a(x).\text{new } i, s. \text{Watchdog}(s, i) \mid i : s[x] \mid \text{Alarm}(i)$$
$$\text{Watchdog}(s, i) = (i \mid s[z]).0$$

Reconciling two influential component models:

- BIP
  - hierarchical components
  - gluing as parallel composition with superimposed synchronisation
  - multipoint synchronization under priority constraints
  - target: embedded, real-time systems
- Fractal
  - hierarchical components with sharing
  - reflective structure for dynamic reconfiguration
  - target: dynamic systems

- 1 Motivations
- 2 CAB: A process calculus interpretation of BIP
- 3 GCAB: Towards reconciling BIP and Fractal

- Primitive components: labelled transitions systems.
- Composite components  $S = (B_1, \dots, B_n)$  with behavioral rules obeying the format:

$$r : \frac{\{B_i \xrightarrow{a_i} B'_i\}_{i \in I} \quad \{B_j \xrightarrow{b_j^k} \mid k \in [1..m_j]\}_{j \in J}}{(B_1, \dots, B_n) \xrightarrow{a} (B'_1, \dots, B'_n)}$$

- Given a family  $\mathcal{P}$  of primitive components,
- CAB composites are built by superimposition of **glue processes**  $P, Q, \dots$  on components

$$\begin{array}{ll} S ::= & \text{component ensembles} \\ | \emptyset & \text{null ensemble} \\ | \{C_1, \dots, C_n\} & \text{finite ensemble} \end{array}$$
$$\begin{array}{ll} C ::= I[S \star P] & \text{component} \\ | I[A] & A \in \mathcal{P} \text{ primitive component} \end{array}$$
$$I, I_i \in \mathcal{N}_I$$



$P, Q ::=$		<i>processes</i>
	$0$	<i>null process</i>
	$  X$	<i>process variable</i>
	$  \alpha.P$	<i>prefix</i>
	$  P   Q$	<i>parallel</i>
	$  \mu X.P$	<i>recursion</i>
$\alpha ::=$	$\langle pr :: a :: syn \rangle$	<i>actions</i>

$pr ::=$

- $\emptyset$  *priority constraints*
- $| \{l_1 : a_1, \dots, l_n : a_n\}$  *void*
- preemptive actions*

$syn ::=$

- $\emptyset$  *synchronisation constraints*
- $| \{l_1 : a_1, \dots, l_n : a_n\}$  *void*
- synchronised actions*

$a, a_i \in \mathcal{N}_p \quad l, l_i \in \mathcal{N}_l$

Behavioral rules for glue processes:

$$\text{Act } \alpha.P \xrightarrow{\alpha} P$$

$$\text{Par1 } \frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}$$

$$\text{Par2 } \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'}$$

$$\text{Rec } \frac{P\{\mu X.P / X\} \xrightarrow{\alpha} P'}{\mu X.P \xrightarrow{\alpha} P'}$$

Behavioral rules for composites:

$$\text{Comp} \frac{P \xrightarrow{\langle pr :: a :: \{l_i : a_i \mid i \in I\}\rangle} P' \quad \{C_i \xrightarrow{l_i : a_i} C'_i \mid i \in I\} \quad \{C_i \mid i \in I\} \subseteq S \quad S \models pr}{I[S \star P] \xrightarrow{l : a} I[(S \setminus \{C_i \mid i \in I\}) \cup \{C'_i \mid i \in I\} \star P']}$$

$$\text{Tau} \frac{C \xrightarrow{h : \tau} C'}{I[\{C\} \cup S \star P] \xrightarrow{l : \tau} I[\{C'\} \cup S \star P]}$$

BIP glues can be encoded in CAB as glue processes of the form  $\llbracket r \rrbracket$ , where:

$$r : \frac{\{C_i \xrightarrow{a_i} C'_i\}_{i \in I} \quad \{C_j \xrightarrow{c_j^k} \mid k \in [1..m_j]\}_{j \in J}}{(C_1, \dots, C_n) \xrightarrow{\text{tag}} (C'_1, \dots, C'_n)}$$

$$\llbracket r \rrbracket = !\langle \{h_j : c_j^k \mid k \in [1, m_j]\}_{j \in J}, \text{tag}, \{h_i : a_i\}_{i \in I} \rangle$$

$$!\alpha.P = \text{rec } X. \alpha.(P \parallel X)$$

## Theorem

*BIP systems defined over a set  $\mathcal{P}$  of components can be faithfully encoded in CAB( $\mathcal{P}$ ): any BIP system  $S$  is strongly bisimilar to its encoding  $\llbracket S \rrbracket$ .*

## Theorem

*CAB( $\emptyset$ ) is Turing complete.*

Proved by encoding of Minsky machines in CAB( $\emptyset$ ).

## Theorem

*CAB( $\emptyset$ ) without priorities is not Turing complete.*

Proved by encoding of CAB( $\emptyset$ ) without priorities in Petri nets.

- 1 Motivations
- 2 CAB: A process calculus interpretation of BIP
- 3 GCAB: Towards reconciling BIP and Fractal**

GCAB generalizes CAB in four ways:

- Pure port synchronization  $\longrightarrow$  value passing on ports.
- Tree structure for composites  $\longrightarrow$  directed graph.
- Static composite structure  $\longrightarrow$  dynamic structure.
- CCS-based glue language  $\longrightarrow$   $\pi$ -calculus-based glue language.



# GCAB (static) core: syntax

$\kappa ::= \Gamma \odot S$

*configurations*

$\Gamma, \Delta ::=$

*control graphs*

$\emptyset$

*empty graph*

$| \{h_1 \triangleright l_1, \dots, h_n \triangleright l_n\}$

*graph*

$S ::=$

*component ensembles*

$\emptyset$

*null ensemble*

$\{C_1, \dots, C_n\}$

*finite ensemble*

$C ::= l[P]$

*component*

$h, l \in \mathcal{N}_l$

$$\text{Act } \alpha.P \xrightarrow{\alpha} P$$

$$\text{Par1 } \frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}$$

$$\text{Par2 } \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'}$$

$$\text{Rec } \frac{P\{\mu X.P / X\} \xrightarrow{\alpha} P'}{\mu X.P \xrightarrow{\alpha} P'}$$

$$\text{GComp} \frac{
 \begin{array}{c}
 \{\Gamma \odot S_i \xrightarrow{l_i:a_i} \Gamma \odot S'_i \mid i \in I\} \quad P \xrightarrow{\langle pr::a::\{l_i:a_i \mid i \in I\} \rangle} P' \\
 \{I \triangleright l_i \mid i \in I\} \subseteq \Gamma \quad \bigcup_{i \in I} S_i \subseteq S \quad \Gamma \odot S \models_I pr
 \end{array}
 }{
 \Gamma \odot \{I[P]\} \cup S \xrightarrow{l:a} \Gamma \odot \{I[P']\} \cup (S \setminus \bigcup_{i \in I} S_i) \cup \bigcup_{i \in I} S'_i
 }$$
  

$$\text{GTau} \frac{
 \Gamma \odot S \xrightarrow{h:\tau} \Gamma \odot S' \quad I \triangleright h \in \Gamma
 }{
 \Gamma \odot \{I[P]\} \cup S \xrightarrow{l:\tau} \Gamma \odot \{I[P]\} \cup S'
 }$$

$$\begin{aligned} \Gamma \odot S \models_I \{l_i : a_i \mid i \in I\} &\iff \exists \{C_i \mid i \in I\}, \\ &\{C_i \mid i \in I\} \subseteq S \\ &\wedge \forall i \in I, \\ &\quad l_i = \mathbf{nm}(C_i) \\ &\quad \wedge \neg(\Gamma \odot S \xrightarrow{l_i:a_i}) \\ &\quad \wedge I \triangleright l_i \in \Gamma \end{aligned}$$

$$\mathcal{F}(R_1, R_2) = \langle R'_1, R'_2 \rangle$$

$$R'_1 = R_1 \cup r(R_1, R_2)$$

$$R'_2 = R_2 \cap r(R_2, R_1)$$

$$r(R_1, R_2) = \{ \langle \kappa, l : a, \kappa' \rangle \mid \text{gcomp}(R_1, R_2, \kappa, l, a, \kappa') \} \\ \cup \{ \langle \kappa, l, \kappa' \rangle \mid \text{gtau}(R_1, \kappa, l : \tau, \kappa') \}$$

$$\text{gtau}(R_1, \kappa, l, \kappa', n) \iff$$

$$\exists \Gamma, P, S, S', h,$$

$$\wedge \kappa = \Gamma \odot \{l[P]\} \cup S$$

$$\wedge \kappa' = \Gamma \odot \{l[P]\} \cup S'$$

$$\wedge l \triangleright h \in \Gamma$$

$$\wedge \langle \Gamma \odot S, h : \tau, \Gamma \odot S' \rangle \in R_1$$

$$\langle R_1, R_2 \rangle \sqsubseteq \langle T_1, T_2 \rangle \iff R_1 \subseteq T_1 \wedge T_2 \subseteq R_2$$

$\mathcal{F}$  is an order-preserving function on a complete lattice

$$\rightarrow \stackrel{\Delta}{=} \pi_1(\mu\mathcal{F})$$

## Theorem

$\rightarrow$  is the least well supported model of the GCAB core rules and it is complete, i.e.  $\mu\mathcal{F} = (\rightarrow, \rightarrow)$ .

- Creating new components:  $\text{new}(x, P)$
- Adding an edge:  $\triangleright a$
- Removing an edge:  $\triangleleft a$
- Updating processes:  $\text{swap}(a, X, P)$

# Formalizing GCAB in Coq ?

- Start with GCAB core
- Essential use of negative premises in rule GComp  $\implies$  no inductive Coq definition possible for  $\rightarrow$ .
- Strategy: formalize  $\mathcal{F}$  fixpoint construction.
  - Define `gcomp` and `gtau` as inductive predicates with relation parameters
  - Define `r` and  $\mathcal{F}$  as functions on relations
  - Prove  $\mathcal{F}$  monotonous wrt  $\sqsubseteq$
  - Obtain  $\mu\mathcal{F}$  as intersection of pre-fixed points
  - Prove  $\mu\mathcal{F}$  of the form  $(R, R)$  and define  $\rightarrow \triangleq R$
  - Define inductive principle on  $\mathcal{F}$ .

Question: can we envisage Coq tools to support this way of dealing with negative premises ?