A model of replicated asymmetric state machines

Roland Kuhn @ Actyx
Daniela Marottoli @ UBA Hernán Melgratti @ UBA
Emilio Tuosto @ GSSI

School of Informatics, Leicester

October 22, 2021

Research partly supported by the EU H2020 RISE programme under the Marie Skłodowska-Curie grant agreement No 778233

BEHAPI Behavioural Application Program Interfaces

Take-away message

Behavioural specs for the (existing) Actyx industrial platform to develop applications for factory automation

Take-away message

Behavioural specs for the (existing) Actyx industrial platform to develop applications for factory automation

With respect to the state-of-the-art, our models

- feature
  - pub-subscribe (instead of point-to-point)
  - generalised choices
  - arbitrary (and variable) number of instances

Take-away message

Behavioural specs for the (existing) Actyx industrial platform to develop applications for factory automation

With respect to the state-of-the-art, our models

- feature
  - pub-subscribe (instead of point-to-point)
  - generalised choices
  - arbitrary (and variable) number of instances
- focus on
  - availability (instead of consensus)
  - eventual-consistency (instead of eg. message loss, deadlock, session fidelity, etc.)

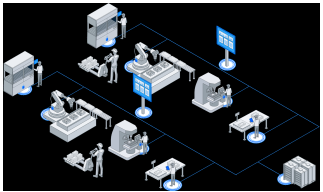
Plan

- Challenges in factory automation
- Our behavioural specifications
- Model-driven realisation
- Concluding remarks

– Prelude –

[Factory automation]

Industrial scenarios



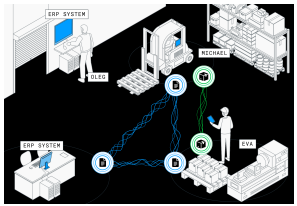
(courtesy of Actyx)

A highly collaborative environment

People + Real-time controllers + IT systems and networks:

- work divided among many autonomous production cells
- efficiency is determined by designing and controlling the flow of resource and information
- local failures must be tolerated for brief time periods

Industrial scenarios



(courtesy of Actyx)

Execution model

machine/operator/forklift/... $\mapsto$ Local twin (state machine)

- twins are replicated where needed
- events have unique IDs and
  - record facts (e.g., from sensors) or
  - decisions (e.g., from an operator)
  - spread asynchronously
- events are kept in local logs used to compute the state of the twin
- logs are replicated and merged

Challenges

Specify application-level protocols where decisions

- don't require consensus

Challenges

Specify application-level protocols where decisions

- don't require consensus
- are based on stale local states

Challenges

Specify application-level protocols where decisions

- don't require consensus
- are based on stale local states
- yet, collaboration has to be successful

– Behavioural Specifications –

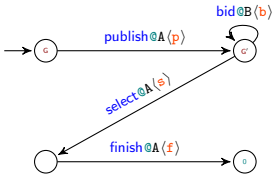
[Decide locally, agree globally...eventually]

Syntax

Global specs

$$G ::= c_1@R_1\langle l_1\rangle . G_1 + \cdots + c_n@R_n\langle l_n\rangle . G_n$$

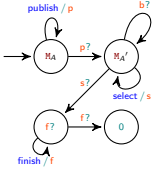
$$G = \text{publish}@A\langle p\rangle . G'$$
$$G' = \text{bid}@B\langle b\rangle . G'$$
$$+$$
$$\text{select}@A\langle s\rangle . \text{finish}@A\langle f\rangle . 0$$

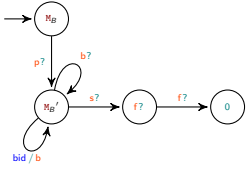


Local Specs

$$M ::= \kappa\cdot[t_1?M_1 \ \& \ \cdots \ \& \ t_n?M_n]$$

$$M_A = \{\text{publish} / p\}\cdot[p?M_A']$$
$$M_A' = \{\text{select} / s\}\cdot[b?M_A' \ \& \ s?\{\text{finish} / f\}\cdot f?0]$$
$$M_B = p?M_B'$$
$$M_B' = \{\text{bid} / b\}\cdot[b?M_B' \ \& \ s?f?0]$$



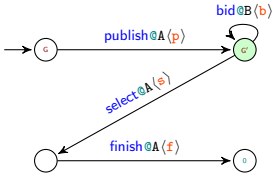


Syntax

Global specs

$$G ::= c_1@R_1\langle l_1\rangle . G_1 + \cdots + c_n@R_n\langle l_n\rangle . G_n$$

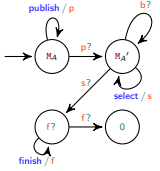
$$G = \text{publish}@A\langle p\rangle . G'$$
$$G' = \text{bid}@B\langle b\rangle . G' + \text{select}@A\langle s\rangle . \text{finish}@A\langle f\rangle . 0$$

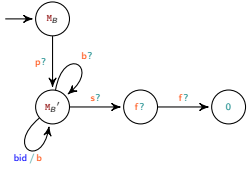


Local Specs

$$M ::= \kappa\cdot[t_1?M_1 \ \& \ \cdots \ \& \ t_n?M_n]$$

$$M_A = \{\text{publish} / p\}\cdot[p?M_A'] \qquad M_A' = \{\text{select} / s\}\cdot[b?M_A' \ \& \ s?\{\text{finish} / f\}\cdot f?0] \qquad M_B = p?M_B' \qquad M_B' = \{\text{bid} / b\}\cdot[b?M_B' \ \& \ s?f?0]$$





Semantics of specifications

Global (protocol spec)

$$\frac{\delta(G, l) \xrightarrow{c\,/\,1} G' \qquad \vdash l' : 1 \qquad l' \textit{ fresh}}{(G, l) \xrightarrow{c\,/\,1} (G, l \cdot l')}$$

where

l is an (idealised) global/shared log

$$\cdots + c_i@R_i\langle 1_i\rangle \,.\, G_i + \cdots \xrightarrow{c_i\,/\,1_i} G_i$$

$$\delta(G, \epsilon) = G$$

$$\delta(G, l) = \begin{cases} \delta(G', l \cdot l') & \text{if } G \xrightarrow{c\,/\,1} G', l \neq \epsilon, \vdash l' : 1 \\ \bot & \text{otherwise} \end{cases}$$

Local (components' spec)

$$\frac{\delta(M, l) = M' \qquad M' \downarrow_{c\,/\,1} \qquad \vdash l' : 1 \qquad l' \text{ fresh}}{(M, l) \xrightarrow{c\,/\,1} (M, l \cdot l')}$$

where

l is the local log accessible to M

$M' \downarrow_{c\,/\,1} \iff c\,/\,1$ enabled at M'

$$\delta(M, \epsilon) = M$$

$$\delta(M, e \cdot l) = \begin{cases} \delta(M_j, l) & \text{if } \vdash e : t_j, \\ & M = \kappa\cdot[\ldots \;\&\; t_j?M_j \;\&\; \ldots] \\ \delta(M, l) & \text{otherwise} \end{cases}$$

Syntax of systems

Events are univocally associated to the machines generating them.

Machines with local logs & a global one

$$\mathtt{S} = (\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$$

(Recall: the global log is an optical illusion)

Well-formedness

$(\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$ is well-formed if

$l = l_1 \cup \cdots \cup l_n$ and for all i, $l_i \sqsubseteq l$

Syntax of systems

Events are univocally associated to the machines generating them.

Machines with local logs & a global one

$$\mathtt{S} = (\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$$

(Recall: the global log is an optical illusion)

Well-formedness

$(\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$ is well-formed if

$l = l_1 \cup \cdots \cup l_n$ and for all i, $l_i \sqsubseteq l$

where $l_i \sqsubseteq l \iff l_i = \begin{matrix} e_{i,1} \\ \vdots \\ e_{i,n} \end{matrix}$ $\begin{matrix} e_1 \\ \vdots \\ e_m \end{matrix} = l$

Syntax of systems

Events are univocally associated to the machines generating them.

Machines with local logs & a global one

$$\mathtt{S} = (\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$$

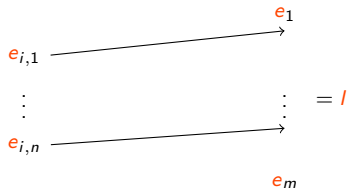
(Recall: the global log is an optical illusion)

Well-formedness

$(\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$ is well-formed if

$l = l_1 \cup \cdots \cup l_n$ and for all i, $l_i \sqsubseteq l$

where $l_i \sqsubseteq l \iff l_i =$



formally, there is an order-preserving

Syntax of systems

Events are univocally associated to the machines generating them.

Machines with local logs & a global one

$$\mathtt{S} = (\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$$

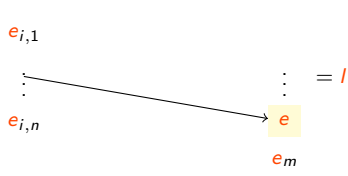
(Recall: the global log is an optical illusion)

Well-formedness

$(\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$ is well-formed if

$l = l_1 \cup \cdots \cup l_n$ and for all i, $l_i \sqsubseteq l$

where $l_i \sqsubseteq l \iff l_i =$



formally, there is an order-preserving and

Syntax of systems

Events are univocally associated to the machines generating them.

Machines with local logs & a global one

$$\mathtt{S} = (\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$$

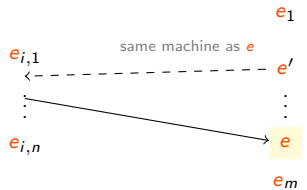
(Recall: the global log is an optical illusion)

Well-formedness

$(\mathtt{M}_1, l_1) \mid \ldots \mid (\mathtt{M}_n, l_n) \mid l$ is well-formed if

$l = l_1 \cup \cdots \cup l_n$ and for all i, $l_i \sqsubseteq l$

where $l_i \sqsubseteq l \iff l_i =$



formally, there is an order-preserving and downward-total morphism from l_i into l

Linking semantics of specs & systems

- Specs describe how to "produce/consume" events
  - Global: how/when roles produce events
  - Local: how/when instances consume events "skipping" irrelevant events

Linking semantics of specs & systems

- Specs describe how to "produce/consume" events
  - Global: how/when roles produce events
  - Local: how/when instances consume events "skipping" irrelevant events
- Communication is non-deterministic

The semantics of systems consists of two phases:

Linking semantics of specs & systems

- Specs describe how to "produce/consume" events
  - Global: how/when roles produce events
  - Local: how/when instances consume events
    "skipping" irrelevant events
- Communication is non-deterministic

The semantics of systems consists of two phases:

First: Events' generation

The local log of a machine is extended with the fresh events generated by the machine

Linking semantics of specs & systems

- Specs describe how to "produce/consume" events
  - Global: how/when roles produce events
  - Local: how/when instances consume events "skipping" irrelevant events
- Communication is non-deterministic

The semantics of systems consists of two phases:

First: Events' generation

The local log of a machine is extended with the fresh events generated by the machine

Second: Events' propagation

Emitted events propagate asynchronously & non-deterministically

Semantics of systems: formally

[Local]

$$\frac{i \in \mathrm{dom}\,\mathtt{S} \qquad \mathtt{S}(i) = (\mathtt{M}, l_i) \qquad (\mathtt{M}, l_i) \xrightarrow{\mathtt{c}\,/\,\mathtt{l}} (\mathtt{M}, l_i') \qquad l' \in l \bowtie l_i'}{(\mathtt{S}, l) \xrightarrow{\mathtt{c}\,/\,\mathtt{l}} (\mathtt{S}[i \mapsto (\mathtt{M}, l_i')], l')}$$

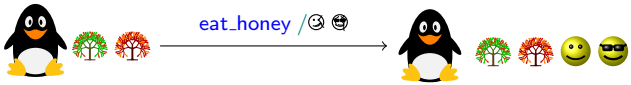
where

$$l_1 \bowtie l_2 = \{ l \mid l \subseteq l_1 \cup l_2 \ \wedge \ l_1 \sqsubseteq l \ \wedge \ l_2 \sqsubseteq l \}$$

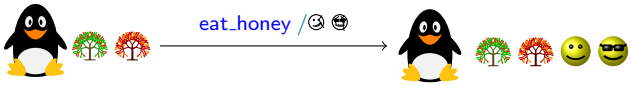
[Prop]

$$\frac{i \in \mathrm{dom}\,\mathtt{S} \qquad \mathtt{S}(i) = (\mathtt{M}, l_i) \qquad l_i \sqsubseteq l' \sqsubseteq l \qquad l_i \subset l'}{(\mathtt{S}, l) \xrightarrow{\tau} (\mathtt{S}[i \mapsto (\mathtt{M}, l')], l)}$$

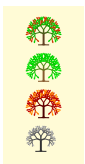
Events generation

If 

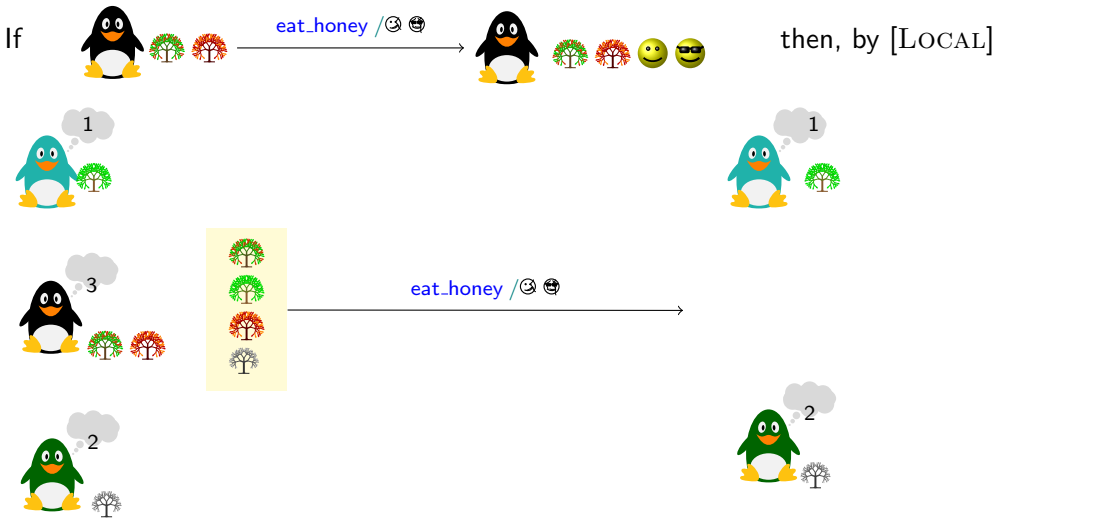
Events generation

If 

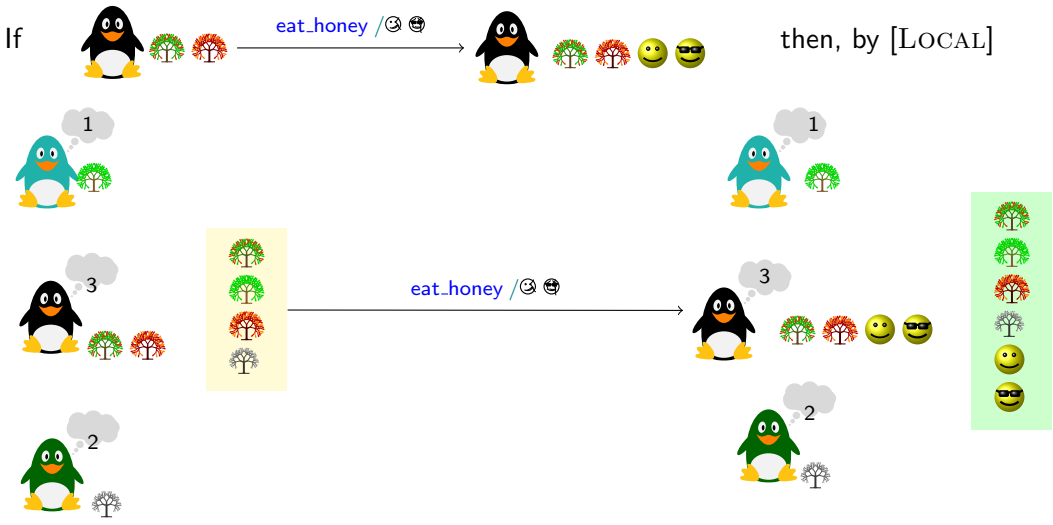
then, by [LOCAL]



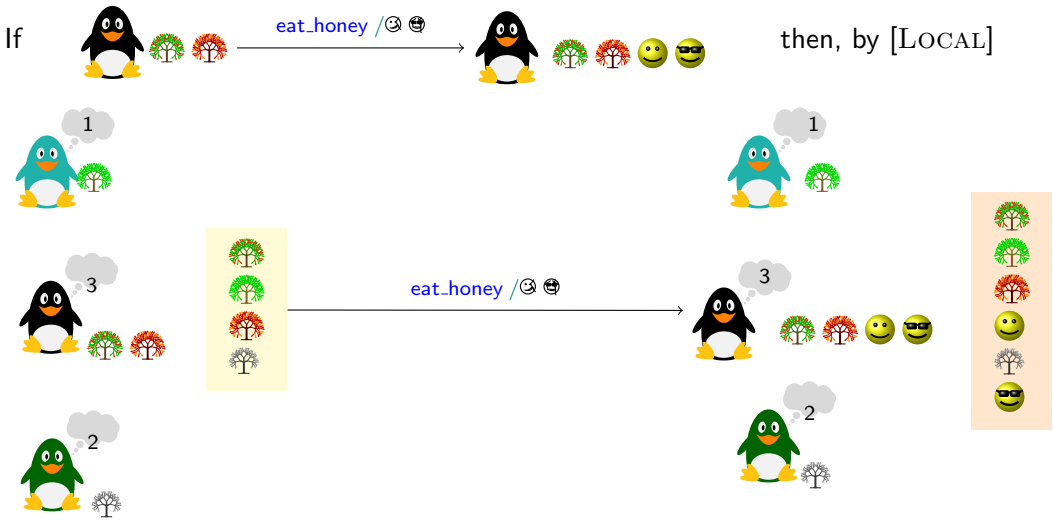
Events generation



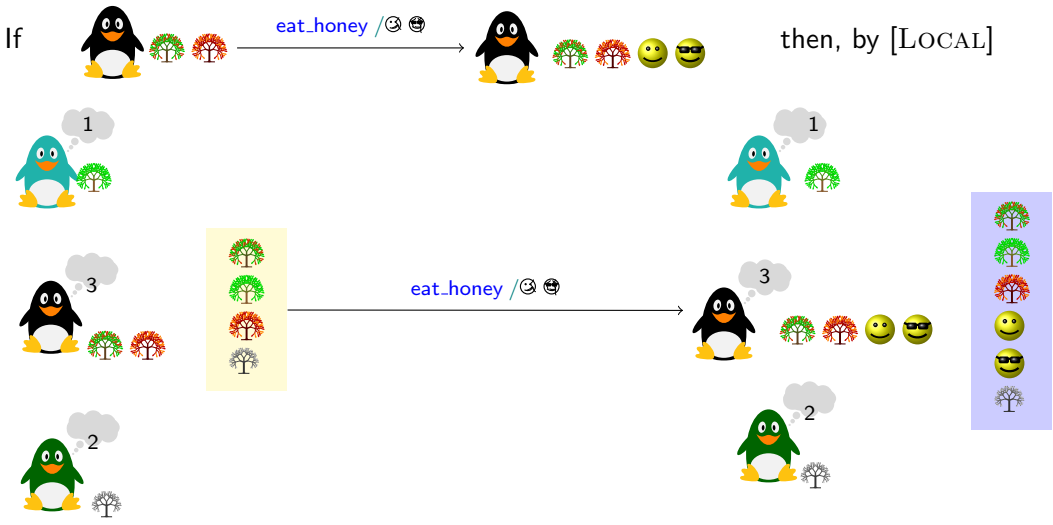
Events generation



Events generation

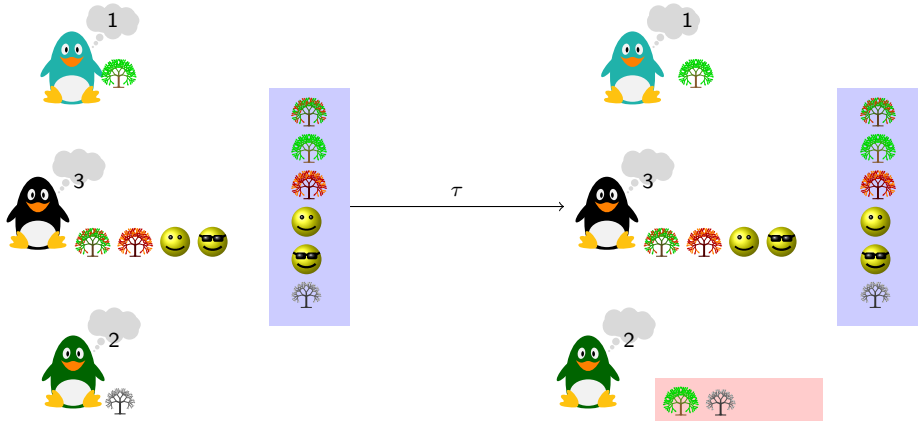


Events generation



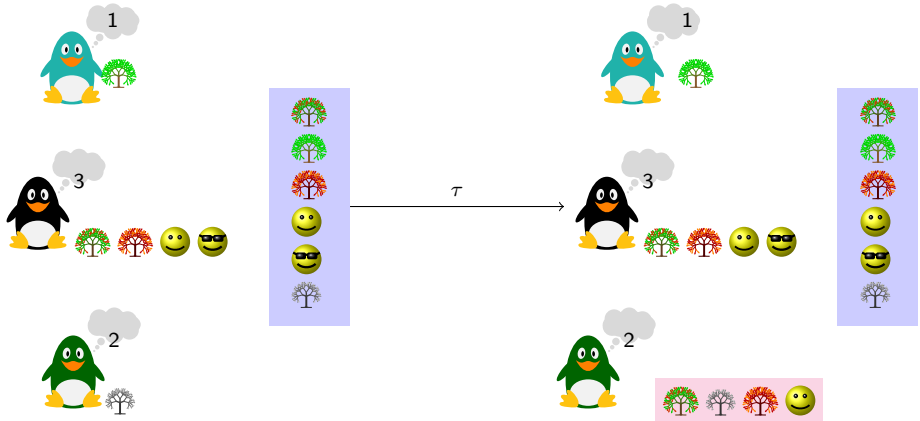
Propagation

By rule [PROP]



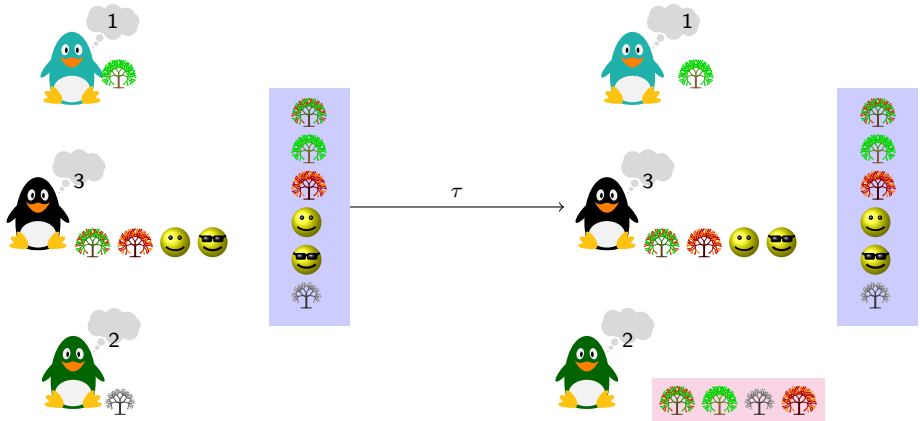
Propagation

By rule [PROP]



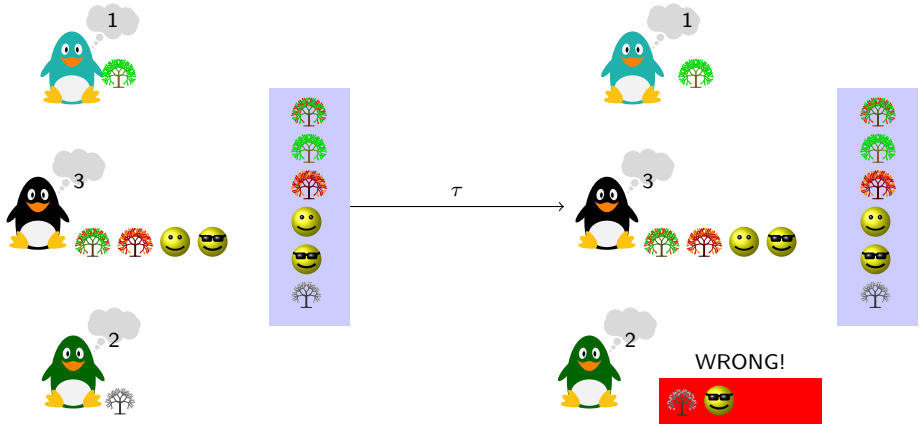
Propagation

By rule [PROP]



Propagation

By rule [PROP]



Properties of our semantics

Theorem: Well-Formedness preservation

If

$\mathtt{S}$ is well-formed and $\mathtt{S} \xrightarrow{[Local]/[Prop]}{}^{\star} \mathtt{S}'$

then

$\mathtt{S}'$ is well-formed

Theorem: Eventual Consistency

If

$\mathtt{S} = (\mathtt{M}_1, I_1) \mid \ldots \mid (\mathtt{M}_n, I_n) \mid I$ is well-formed

then

$\mathtt{S} \xrightarrow{\tau}{}^{\star} (\mathtt{M}_1, I) \mid \ldots \mid (\mathtt{M}_n, I) \mid I$

– From models to implementations –

[On realisation]

Exercise on realisation

It is hard to get realisation right (even without multi-instances or choices!)

A trivial protocol

Take

$$\mathtt{G} = \mathtt{prepare@P}\langle \mathtt{piece} \rangle \,.\, \mathtt{pack@A}\langle \mathtt{product} \rangle \,.\, 0$$

Do the following machines realise G?

$$\mathtt{M}_P = \{\mathtt{prepare} \,/\, \mathtt{pieces}\}\cdot 0 \qquad \text{and} \qquad \mathtt{M}_A = \mathtt{t}_1?\,\{\mathtt{pack} \,/\, \mathtt{product}\}\cdot 0$$

Exercise on realisation

It is hard to get realisation right (even without multi-instances or choices!)

A trivial protocol

Take

$$\mathtt{G} = \mathtt{prepare@P}\langle \mathtt{piece} \rangle \,.\, \mathtt{pack@A}\langle \mathtt{product} \rangle \,.\, 0$$

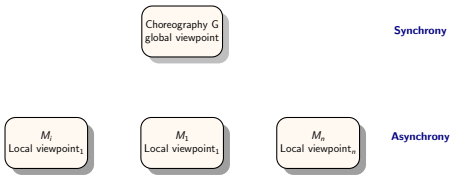
Do the following machines realise G?

$$\mathtt{M}_P = \{\mathtt{prepare} \,/\, \mathtt{pieces}\}\cdot 0 \qquad \text{and} \qquad \mathtt{M}_A = \mathtt{t}_1?\,\{\mathtt{pack} \,/\, \mathtt{product}\}\cdot 0$$

Do they do that "correctly"?

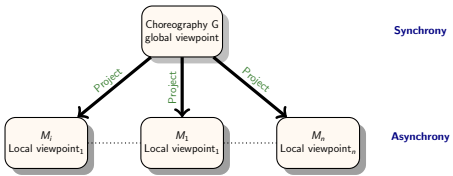
Choreographies provide a principled development approach

Natural support for choreographic design



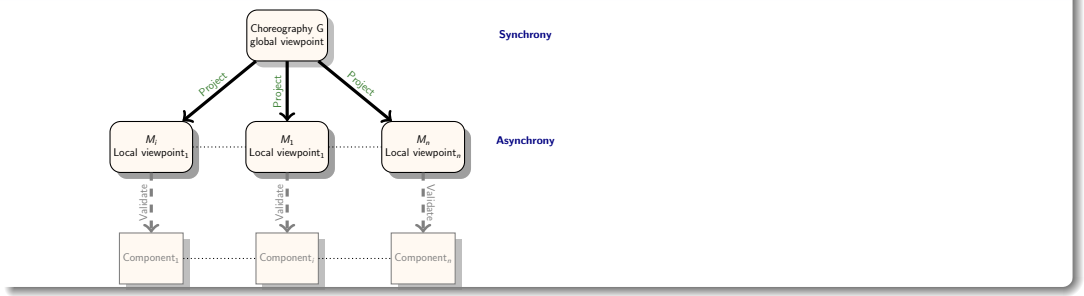
Choreographies provide a principled development approach

Natural support for choreographic design



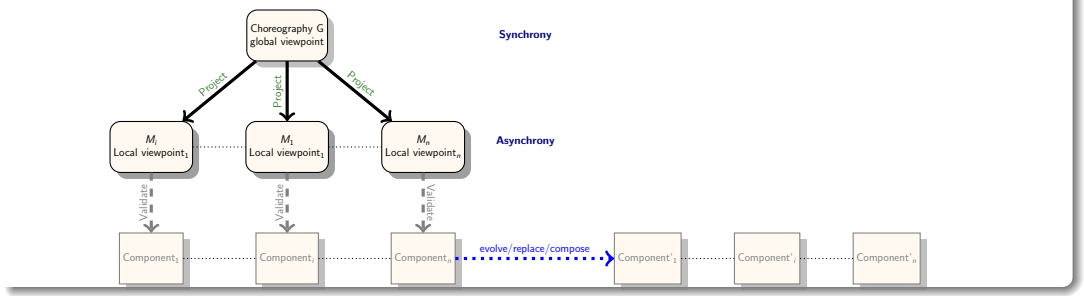
Choreographies provide a principled development approach

Natural support for choreographic design



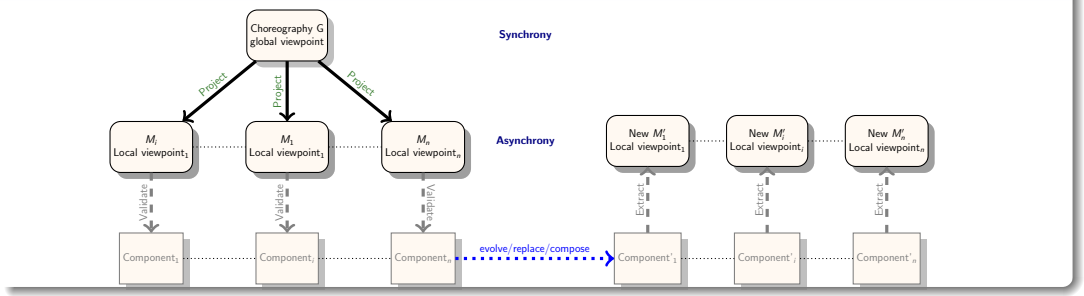
Choreographies provide a principled development approach

Natural support for choreographic design



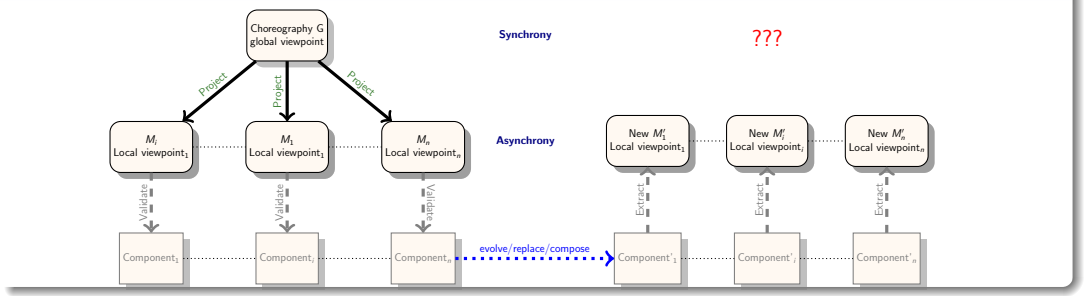
Choreographies provide a principled development approach

Natural support for choreographic design



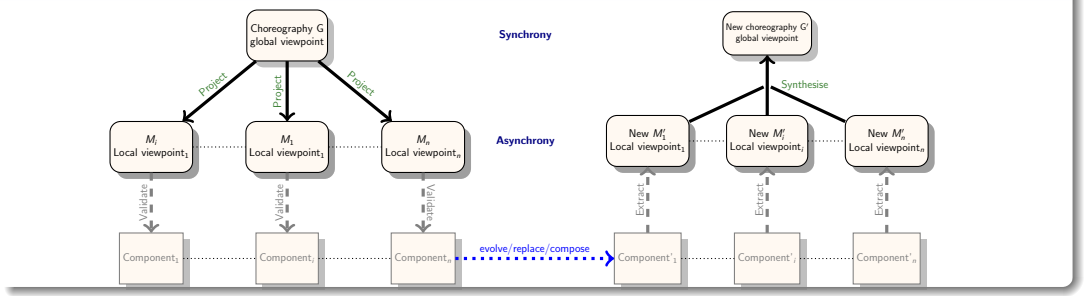
Choreographies provide a principled development approach

Natural support for choreographic design



Choreographies provide a principled development approach

Natural support for choreographic design



Realisation...choreographically

Well-formedness of global specs

Each guard, say $\mathtt{l}_i$, should be

- causal consistent
  - each selector in (the continuation of) $\mathtt{l}_i$ reacts to $\mathtt{l}_i$
  - each role involved in the continuation of $\mathtt{l}_i$ cannot react to more events on $\mathtt{l}_i$ than selectors on the branch
- determined
  - each role in the continuation of $\mathtt{l}_i$ reacts to $\mathtt{l}_i[0]$
  - selectors in the continuation of $\mathtt{l}_i$ react to the same set of event types in $\mathtt{l}_i$
- confusion-free
  - guards of different branches start with distinct event types
  - an event type cannot occur in more than one guard

A glimpse of top-down design

Definition (Projection)

Given a global type G and a subscription σ, the *projection of* G *over a role* $\mathtt{R}$ *with respect to* σ, written $\mathsf{G}\downarrow^{\sigma}_{\mathtt{R}}$, is co-inductively defined as follows:

$$\Big(\sum_{i\in I} \mathsf{c}_i@\mathtt{R}_i\langle \mathtt{l}_i\rangle \,.\, \mathsf{G}_i\Big)\downarrow^{\sigma}_{\mathtt{R}} \quad = \quad \kappa\cdot\Big[\&_{i\in I}\,(\mathsf{c}_i@\mathtt{R}_i\langle \mathtt{l}_i\rangle \,.\, \mathsf{G}_i)\downarrow^{\sigma}_{\mathtt{R}}\Big]$$

where $\kappa = \{(\mathsf{c}_i, \mathtt{l}_i) \mid i \in I \wedge \mathtt{R}_i = \mathtt{R}\}$ and

$$(\mathsf{c}_i@\mathtt{R}_i\langle \mathtt{l}_i\rangle \,.\, \mathsf{G}_i)\downarrow^{\sigma}_{\mathtt{R}} = \begin{cases} \mathsf{filter}(\mathtt{l}_i, \sigma(\mathtt{R}))\ ?\ (\mathsf{G}_i\downarrow^{\sigma}_{\mathtt{R}}) & \text{if } \mathsf{filter}(\mathtt{l}_i, \sigma(\mathtt{R})) \neq \epsilon \\ 0 & \text{otherwise} \end{cases}$$

– Wrapping up –

Conclusions

We have a useful basis to take design decisions

- a formal model ensuring relevant properties
- projectable global specs
- a prototype implementation for top-down engineering

Conclusions

We have a useful basis to take design decisions

- a formal model ensuring relevant properties
- projectable global specs
- a prototype implementation for top-down engineering

We would like to have

- our models as reference documentation for Actyx's developers
- "minimal" subscriptions
- tools / develop behavioral typing / inference (ie going bottom-up)
- compensations (hence causality tracking) / active monitoring?
- failure handling (in event propagation)

Thank you!