IMT SCHOOL FOR ADVANCED STUDIES LUCCA

A type language for message passing component-based systems

Zorica Savanović[1], Hugo Vieira[2] and Letterio Galletta[1]

ICE 2020, June 19, 2020

[1]IMT School for Advanced Studies, Lucca, Italy
[2]University of Beira Interior, Portugal

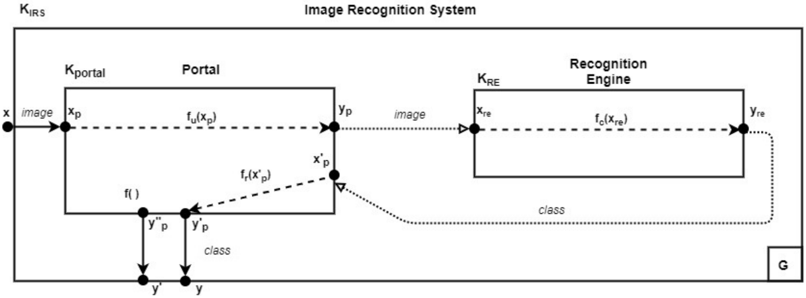
Introduction

- Code reusability is a key principle in Component-Based Development (CBD).[1]
- Solutions for code reuse in distributed software systems are lacking.
- A component should be able to carry out a certain sequence of input/output actions in order to fulfil its role in the protocol.

  Too strict.
- Components respond to an external stimulus.

  Too wild.
- Carbone, Montesi and Vieira [2] proposed a language (Governed components language): merging reactive components with choreographic specifications of communication protocols [3].
- Our contribution is at the level of the **type language** that allows to capture component's behaviour so as to check its compatibility with a protocol.
- Once the component's type is identified, there is no further need to check the implementation.

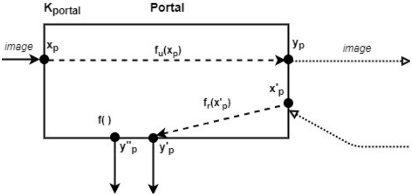
Image Recognition System



Base Components

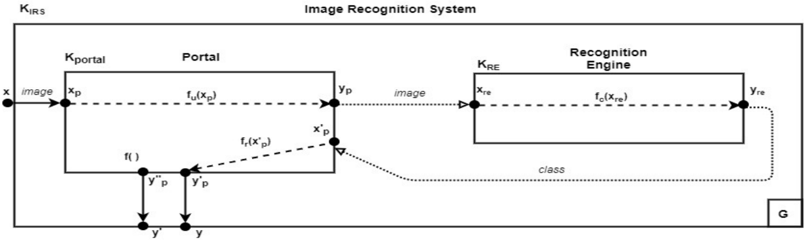
Base component K_{Portal}

Portal



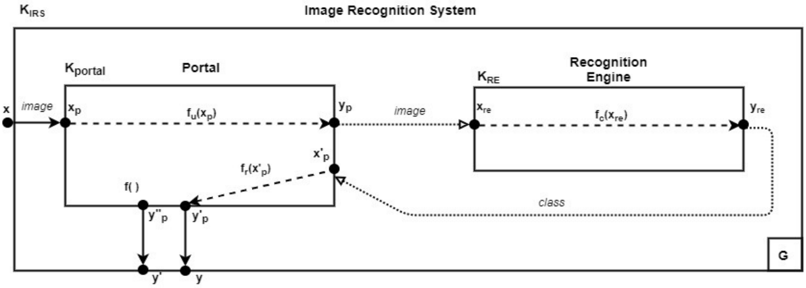
- $f_u(x_p) = image$, $f_r(x'_p) = class$ and $f(\ ) = version$
- Received *images*/*classes* are processed in a FIFO discipline
- $f(\ )$ can always perform an output regardless of inputs

"One-shot" protocol G



$$G = \text{Portal} \xrightarrow{image} \text{RE}; \text{RE} \xrightarrow{class} \text{Portal}; \textbf{end}$$

Recursive protocol G



$$G' = \mu \mathbf{X}.\text{Portal} \xrightarrow{image} \text{RE}; \text{RE} \xrightarrow{class} \text{Portal}; \mathbf{X}$$

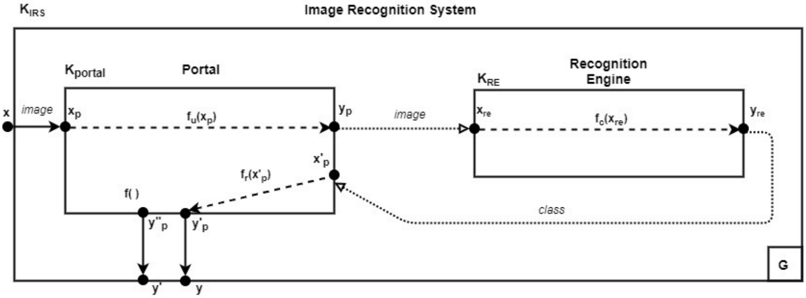
The Type language syntax

Types and input interfaces	Dependency kinds	Boundaries
$T \triangleq < X_b; \mathbf{C} > \quad X_b \triangleq \{x_1(b_1), \dots, x_k(b_k)\}$	$M ::= N \mid \Omega$	$\mathbf{B} ::= N \mid \infty$
Constraints	Dependencies	
$\mathbf{C} \triangleq \{y_1(b_1) : \mathbf{B}_1 : [\mathbf{D}_1], \dots, y_k(b_k) : \mathbf{B}_k : [\mathbf{D}_k]\}$	$\mathbf{D} \triangleq \{x_1 : M_1, \dots, x_k : M_k\}$	$k \geq 0 \quad N \in \mathbb{N}_0$

- Two type extraction procedures:
- For base components
- For composite components
  - Interfacing component
  - Local protocol (projection of a global protocol)

The type of K_{IRS}, G "one-shot"

$$T_{IRS} =< \{x(image)\}; \{y(class) : 1 : [\{x : \Omega\}], y'(version) : \infty : [\varnothing]\} >$$



SCHOOL FOR ADVANCED STUDIES LUCCA

Type of K_{IRS}, G recursive protocol

$$< \{x(image)\}; \{y(class) : \infty : [\{x:0\}], y'(version) : \infty : [\varnothing]\} >$$

- Input two values on port x

Type of K_{IRS}, G recursive protocol

$$< \{x(image)\}; \{y(class) : \infty : [\{x:0\}], y'(version) : \infty : [\emptyset]\} >$$

- Input two values on port x
- $< \{x(image)\}; \{y(class) : \infty : [x:2], y'(version) : \infty : [\emptyset]\} >$

Type of K_{IRS}, G recursive protocol

$$< \{x(image)\}; \{y(class) : \infty : [\{x:0\}], y'(version) : \infty : [\varnothing]\} >$$

- Input two values on port x
- $< \{x(image)\}; \{y(class) : \infty : [x:2], y'(version) : \infty : [\varnothing]\} >$
- Output from port y one value

Type of K_{IRS}, G recursive protocol

$$< \{x(image)\}; \{y(class) : \infty : [\{x : 0\}], y'(version) : \infty : [\varnothing]\} >$$

- Input two values on port x
- $< \{x(image)\}; \{y(class) : \infty : [x : 2], y'(version) : \infty : [\varnothing]\} >$
- Output from port y one value
- $< \{x(image)\}; \{y(class) : \infty : [x : 1], y'(version) : \infty : [\varnothing]\} >$

Type of K_{IRS}, G recursive protocol

$$< \{x(image)\}; \{y(class) : \infty : [\{x:0\}], y'(version) : \infty : [\varnothing]\} >$$

- Input two values on port x
- $< \{x(image)\}; \{y(class) : \infty : [x:2], y'(version) : \infty : [\varnothing]\} >$
- Output from port y one value
- $< \{x(image)\}; \{y(class) : \infty : [x:1], y'(version) : \infty : [\varnothing]\} >$
- CAN DO: $y'!.x?.y!.x?.y!.x?$
- CANNOT DO: $x?.y!.y!.x?.x?.y'!$ (dependency)

Main Results

Theorem (Subject Reduction)

If $K \Downarrow T$ and $K \xrightarrow{\lambda(v)} K'$ and v has type b then $T \xrightarrow{\lambda(b)} T'$ and $K' \Downarrow T'$.

Theorem (Progress)

If $K \Downarrow T$ and $T \xrightarrow{\lambda(b)} T'$ and $\lambda(b) \neq \tau$ then b is the type of a value v and $K \xRightarrow{\lambda(v)} K'$ and $K' \Downarrow T'$.

$K \xRightarrow{\lambda(v)} K'$ denotes a sequence of transitions
$K \xrightarrow{\tau} \cdots K'' \xrightarrow{\lambda(v)} K''' \xrightarrow{\tau} \cdots K'$.

IMT SCHOOL FOR ADVANCED STUDIES LUCCA

Difference with respect to related approaches

- The approach proposed by Carbone, Montesi and Vieira [4]: we consider a different approach, avoiding the implementation check each time a component is to be used.
- Open Multiparty Sessions [5]: our components are potentially more reusable considering the I/O flexibility provided the reactive flavour;
- CHOReVOLUTION project[6]: our type-based approach that aims at abstracting from the implementation and providing more general support for component substitution and reuse.
- FACTum[7]: do not provide any means to automatically extract types from given components.

---

[4] M. Carbone, F. Montesi, and H. T. Vieira. Choreographies for reactive programming. CoRR, abs/1801.08107, 2018.

[5] F. Barbanera and M. Dezani-Ciancaglini. Open multiparty sessions.

[6] CHOReVOLUTION project. http://www.chorevolution.eu.

[7] Marmsoler Diego, and Habtom Kashay Gidey. "Interactive verification of architectural design patterns in FACTum." Formal Aspects of Computing 31.5 (2019): 541-610.

IMT SCHOOL FOR ADVANCED STUDIES LUCCA

Concluding Remarks

- We introduce a type language for the choice-free subset of the GC language
- Type language (syntax)
- We do static typing: inspecting the source code so as to avoid runtime errors.
- Subject reduction and Progress
- Typing descriptions such as ours are crucial to promote component reusability
- Support for protocols with branching;
- Subtyping;
- Conveying the theoretical model to concrete applications.

THANK YOU FOR ATTENTION!