

Formal Methods for Component Description, Coordination and Adaptation

- a position, some facts and some open issues -

Pascal POIZAT

poizat@lami.univ-evry.fr

<http://www.lami.univ-evry.fr/~poizat/recherche-fr.php>

Équipe SPECIF, LaMI - UMR 8042 CNRS & Université d'Évry Val d'Essonne, GENOPOLE[®]



What is needed ?

- high level of abstraction
 - ⇒ abstract away from implementation issues
 - ⇒ better verification means, decidability

What is needed ?

- high level of abstraction

- behavioural IDLS

- ⇒ fact: IDLs are not sufficient

- ⇒ BIDLs as contracts only

- + verification / (semi)-automatic adaptation means

- ⇒ for both full ADL and Coordination

Specific aspect-dedicated languages based on this schema:

1. a Static description Language (SL)

⇒ ADT, Z, B, ...

Specific aspect-dedicated languages based on this schema:

1. a Static description Language (SL)
 - ⇒ ADT, Z, B, ...
2. a Dynamic (behavioural) description Language (DL)
 - Modal (Temporal) Logics, PA or Patterns
 - ⇒ PA preferred

Specific aspect-dedicated languages based on this schema:

1. a Static description Language (SL)
 - ⇒ ADT, Z, B, ...
2. a Dynamic (behavioural) description Language (DL)
 - Modal (Temporal) Logics, PA or Patterns
 - ⇒ PA preferred
3. a Glue Language (GL)
 - ⇒ Modal Logics

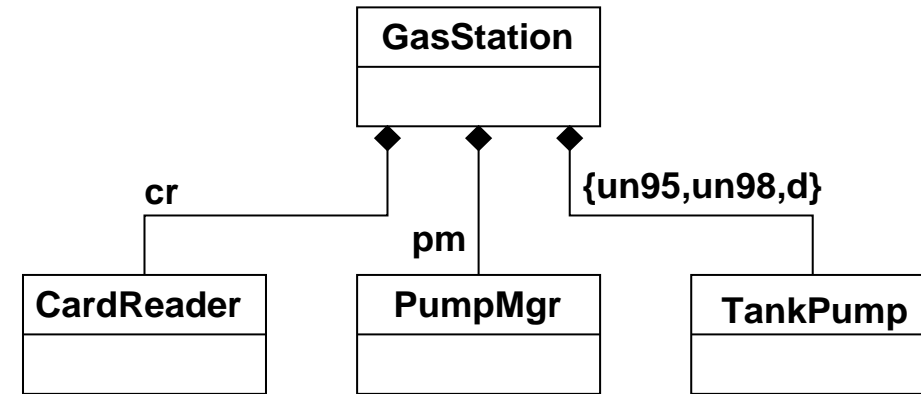
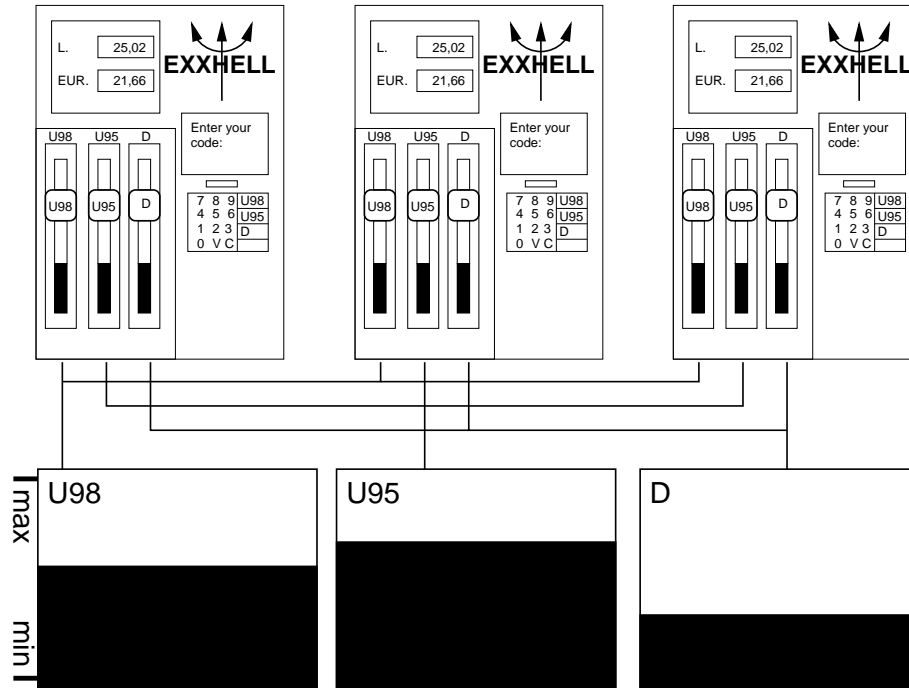
Specific aspect-dedicated languages based on this schema:

1. a Static description Language (SL)
 - ⇒ ADT, Z, B, ...
2. a Dynamic (behavioural) description Language (DL)
 - Modal (Temporal) Logics, PA or Patterns
 - ⇒ PA preferred
3. a Glue Language (GL)
 - ⇒ Modal Logics

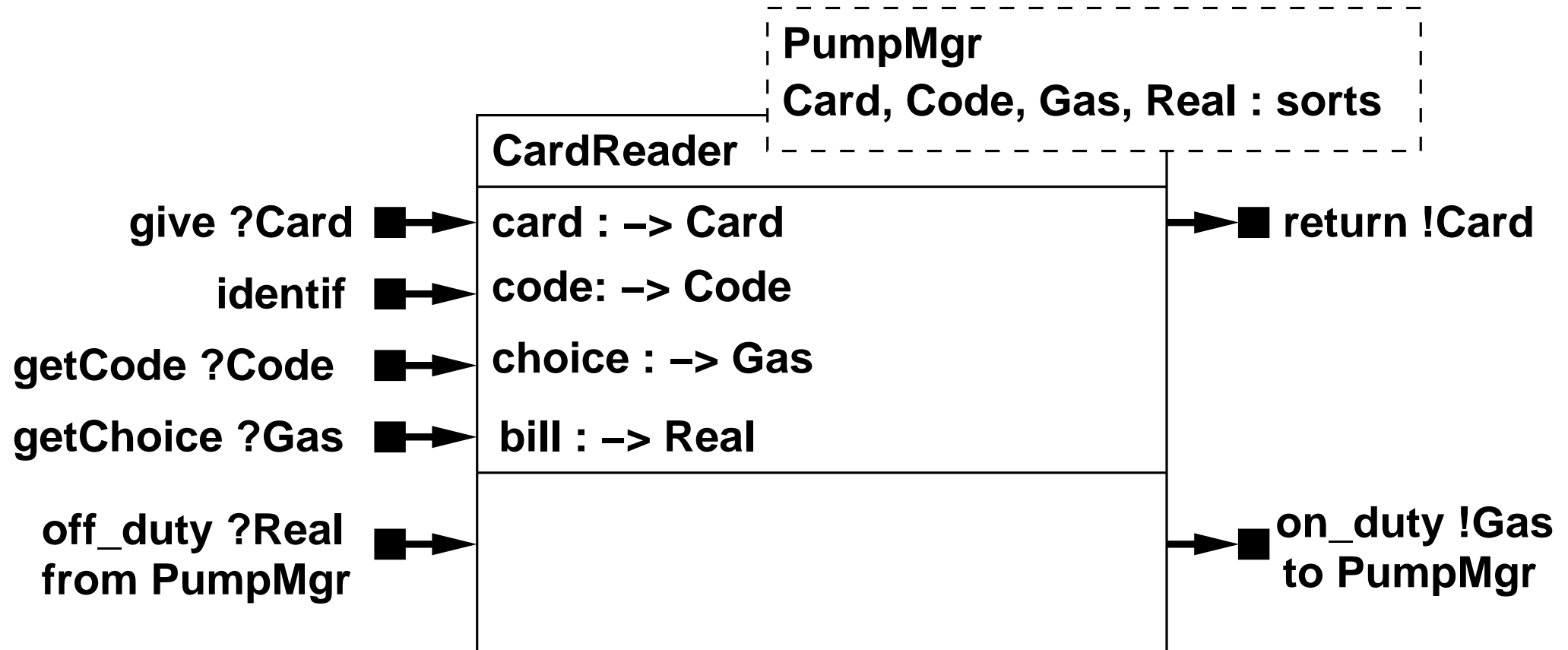
Benefits: high expressiveness, high abstraction level, glue as coordination

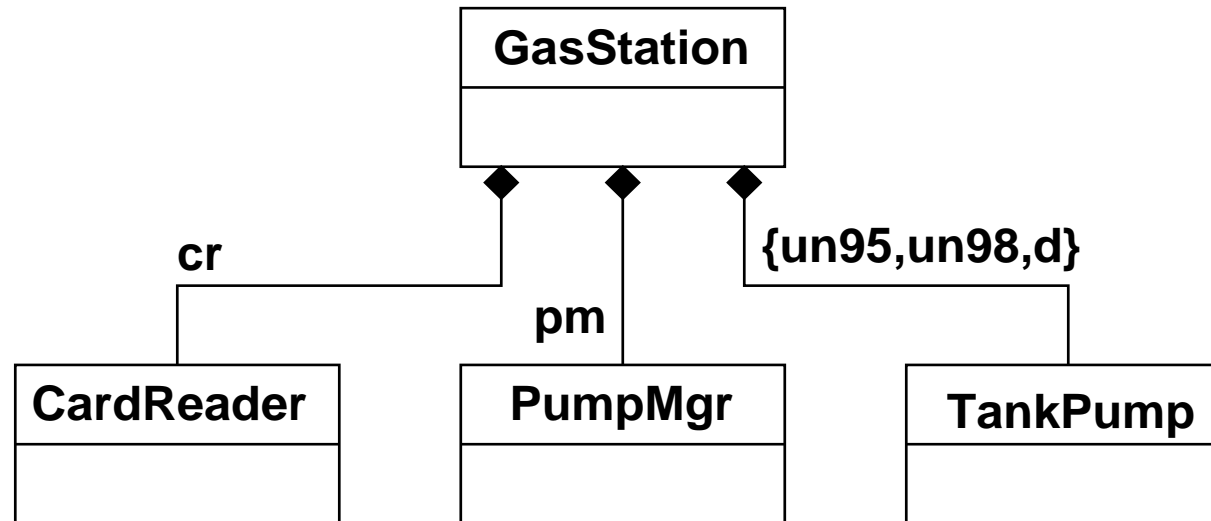
e.g.: UML+Formal Data, Korrigan (from [FOCLASA'2003])

GasStation



CardReader, interface (Korrigan)





$cr.[on_duty!Un98 \uparrow pm]true \Rightarrow pm.[activate \uparrow un98]true$

$\mathbf{ALL}(S).\tilde{\psi} \stackrel{def}{=} \bigwedge_{x \in S} x.\tilde{\psi}$, $\mathbf{ONE}(S).\tilde{\psi} \stackrel{def}{=} \bigvee_{x \in S} (x.\tilde{\psi} \wedge_{y \in S, x \neq y} \neg y.\tilde{\psi})$

- (explicit) synchronizing: $c_1.[a]\mathbf{true} \Leftrightarrow c_2.[a]\mathbf{true}$
- connecting event ports with different names:
 $c_1.[a_1]\mathbf{true} \Leftrightarrow c_2.[a_2]\mathbf{true}$
- value passing: $\forall x : \mathit{Nat}(c_1.[a?x]\mathbf{true} \Leftrightarrow c_2.[a!x]\mathbf{true})$
- broadcasting (1-to-N, all in the same time):
 $server.[send]\mathbf{true} \Rightarrow \mathbf{ALL}(client.i : [1..N]).[receive]\mathbf{true}$
- exclusive (1-with-N, but one at a time):
 $server.[send]\mathbf{true} \Rightarrow \mathbf{ONE}(client.i : [1..N]).[receive]\mathbf{true}$
- exclusive states:
 $\neg(cooler_1.(active = true) \wedge cooler_2.(active = true))$

** FACTS **

If we restrict to PA as DL, then:

- animation / prototyping
- equivalences or preorders
- compositionnality
- deadlock freedom checking
- (semi)-automatic adapters

eg: Trendy application domain: WS (see paper)
but ...

** FACTS **

If we restrict to PA as DL, then:

- animation / prototyping
- equivalences or preorders
- compositionnality
- deadlock freedom checking
- (semi)-automatic adapters

eg: Trendy application domain: WS (see paper)
but ...

- few tools (CADP, FSP, xCLAP)
- the STS adaptation problem

■ CBSE ones

- ⇒ Graphical notations
- ⇒ Seamless processes

- CBSE ones
- Formal ones
- ⇒ Fact: there are PA for nearly any aspect of interest (data, time, stochastics, locations, ...)

- CBSE ones

- Formal ones

- ⇒ Fact: there are PA for nearly any aspect of interest (data, time, stochastics, locations, ...)

- ⇒ what about the PA benefits ?

- ⇒ can the [GL/DL/SL] framework be adapted for each one ?

- ⇒ how to "weave" several ones ?
ongoing work: (Logical) Fibring
(Logics as OMG MM M2)

- ⇒ abstraction/complexity issues

- ongoing work: mailbox analysis study
(FIFO vs. Dict./FIFFO)

time for discussion ...