Incremental Component-Based Construction and Verification of a Robotic System

Ananda Basu, Matthieu Gallien, Charles Lesire, Thanh-Hung Nguyen, Saddek Bensalem, Félix Ingrand, Joseph Sifakis

Control Architectures of Robots 2007
LAAS Architecture

• Functional Level: modules developed using GenoM; provide services and posters

• Navigation Loop: Laser + Aspect + NDD + RFLEX

• Centralized execution control: R2C, safety constraints and rules
BIP
Behavior Interaction Priorities

• Complex systems are built by assembling components (building-blocks)

• Components are systems characterized by their interface, an abstraction that is adequate for composition and reuse

• Large components are obtained by "gluing" together simpler ones
Component-based construction

- Develop a *rigorous* and *general basis* for real-time system design and implementation

- Concept of component and associated composition operators for *incremental* description and *correctness by construction*

- Concept for real-time architecture *encompassing* heterogeneity, paradigms and styles of computation

- *Automated* support for component *integration* and generation of *glue code meeting given requirements*
Formal framework

- Build a component $C$ satisfying a given property $P$ from:
  - $C$ a set of atomic components modeling behavior
  - $\mathcal{GL}$ a set of glue operators on components

- Glue operators
  - model mechanisms used for communication and control such as protocols, controllers, buses...
  - restrict the behavior of their arguments
The BIP Framework

Layered component model

Composition (incremental description)
BIP: Behavior

- An atomic component has:
  - a set of ports \( P \), for interaction with other components.
  - a set of control states \( S \).
  - a set of variables \( V \).
  - a set of transitions of the form:
    - \( p \) is a port,
    - \( g \) is a guard, boolean expression on \( V \),
    - \( f \) is a function on \( V \) (block of code).

\[
\begin{align*}
\text{in} & \quad x & \text{in} & \quad y := f(x) & \text{out} & \quad y \\
\text{empty} & \quad \text{in} & \quad 0 < x & \text{out} & \quad \text{full} \\
\text{out} & \quad \text{in} & \quad x & \text{out} & \quad y \\
\end{align*}
\]
BIP : Interaction

- A connector is a set of ports that can be involved in an interaction
- Port attributes (*complete*, *incomplete*) are used to distinguish between *broadcast* and *rendezvous*
- Interactions: \{tick1,tick2,tick3\}\{out1\}\{out1,in2\}\{out1,in2,in3\}
Componentization of the functional level

- Functional Level ::= Module+
- Module ::= Service+ . Control Task . Poster+
- Service ::= Execution Task . Activity
- Control Task ::= Timer . Scheduler Activity
BIP model of a service

GenoM

BIP
BIP model of a module

- **GenoM**
  - Control Task
  - Execution Tasks
  - Control IDS
  - Functional IDS
  - Request
  - Report
  - Services Interface
  - Posters interface

- **BIP**
  - Execution Task
  - Activity
  - Service
  - Timer
  - Scheduler Activity
  - Control Task
  - Module
  - Poster
  - Poster

---

*GenoM BIP*
BIP model of the NDD module
Execution

• Generation of a multithreaded BIP engine
• Executes interactions → functions called in a "GenoM" library
• Poster data managed via GenoM posterLib and shared memory
• Request and reports sent via mailboxes ⇒ interfaces with tcl, OpenPRS, test programs...
Execution control

- Constraints modeled as connectors
  see goTo.trigger connector

- Observers for on-line safety properties
  time constraints violation
Verification

- Deadlocks
- Model-Checking
  
  e.g. verify that `goTo.trigger` is always executed after `SetX` services are complete
- Time properties

  Observers representing the desire properties; used offline in exploration to verify the property, and online for monitoring
NDD period verification

```
tick c := 0
trigger

finish
tick c >= p

ERROR

.tick c < p
c := c+1

finish
```

State diagram:
- **IDLE**
  - Transition on tick: c := 0
- **EXEC**
  - Transition on tick: c >= p
  - Transition on c < p: c := c+1
  - Transition on finish
- **ERROR**
  - Transition on tick: c < p, c := c+1
Ongoing work

- Modeling of other modules:  
  *Aspect, Laser, RFLEX, PoM* ⇒ *navigation loop*

- Preparation of associated libraries for integration within BIP modules
Ongoing work

- **Constraints:**

  NDD navigation (exec) possible only if PoM has been launched (*Pos* poster contains a relevant position)
Ongoing work

- **Time property:**

  Laser scans an obstacle at $t$, which enters the loop (Aspect, NDD, RFLEX) and induces a stop (or avoidance) of the robot at $t'$. What's the delay ($|t'-t|_{max}$) we can guarantee?
Current Limitations/Prospectives

• **Philosophical:**

  - complexity of verification techniques for the whole architecture?
  - state-space exploration, tick-based representation

  - integration of the executive as a BIP component?
  - by acquiring macro actions? (*Move*, *TakePicture*...)
  - by acquiring the complete plan?

  In this case, what about plan verification?
• Recherche PostDoc sur ce sujet...