Using SysML in a RTC-based Robotics Application: a case study with a demo

Kenji Hiranabe (Change Vision, Inc)
Noriaki Ando (AIST)
Agenda

• Introduction
• Background and Goals
• Problem
• Analysis and Design via Demo
• Conclusion
• Future Ideas
• Kenji Hiranabe, Change Vision, Inc. (maker of Astah)
• Astah is a UML editor popular in Japan
  – http://astah.net/

• Astah/SysML
  – Newly developed
  – Focused on “Usability” and “Web collaboration”

• RTC plug-in
  – Plug-in for Astah/SysML to generate RTC.xml to OpenRTM
Project members

Japanese Project

- Noriaki Ando
- Kenji Hiranabe
- Toshihiro Okamura
- Makoto Sekiya
- Toyotaka Torii

English Project

- Geoffrey Biggs
- Isaac Hara
- Toshiki Iwanaga
- Toshihiro Okamura

Translation:

1. SysML to RTC
2. OpenRTM to Honda RTM
Background and Goals

• Evaluate how SysML can help design a component(RTC)-based robotic application using a simple problem.

1. SysML to RTC

• Try a demonstration test to verify that one common model can work and interoperate on multiple RTM implementations.
  – OpenRTM-aist
  – Honda R&D RTM

2. OpenRTM to Honda RTM
Demonstrate the movements (Spiral and Back-and-Forth) by controlling multiple autonomous robots from externally. Operator can switch between the autonomous mode and demonstration mode.

Hardware architecture is already known, we use Roomba with PC that can control it using Wi-Fi and use Kinect to switch the mode.
# OMG RTC Family

<table>
<thead>
<tr>
<th>Name</th>
<th>Vendor</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenRTM-aist</td>
<td>AIST</td>
<td>C++, Python, Java</td>
</tr>
<tr>
<td>OpenRTM.NET</td>
<td>SEC</td>
<td>.NET(C#, VB, C++/CLI, F#, etc..)</td>
</tr>
<tr>
<td>miniRTC, microRTC</td>
<td>SEC</td>
<td>RTC implementation for CAN • ZigBee based systems</td>
</tr>
<tr>
<td>Dependable RTM</td>
<td>SEC/AIST</td>
<td>Functional safety standard (IEC61508) capable RTM implementation</td>
</tr>
<tr>
<td>RTC CANOpen</td>
<td>SIT, CiA</td>
<td>Standard for RTC mapping to CANOpen by CiA (Can in automation) and implementation by SIT</td>
</tr>
<tr>
<td>PALRO</td>
<td>Fuji Soft</td>
<td>C++ PSM implementation for small humanoid robot</td>
</tr>
<tr>
<td>OPRoS</td>
<td>ETRI</td>
<td>Developed by Korean national project</td>
</tr>
<tr>
<td>GostaiRTC</td>
<td>GOSTAI, THALES</td>
<td>C++ PSM implementation on URBI</td>
</tr>
<tr>
<td>Honda R&amp;D RTM</td>
<td>Honda R&amp;D</td>
<td>C++, Python, FSM Component</td>
</tr>
</tbody>
</table>
Analysis and Design Diagrams in Astah / SysML
Overview

Analysis
1. Requirement Analysis
   - Core requirements
   - Derived requirements
   - Requirement list
   - Context diagram
   - Use cases

2. System Analysis
   - System Structure & Interface
   - Demo components
   - Controller and requirements
   - State machine

Design
1. Structure Design
2. Behavior Design
3. Review
**Demonstration by robots**

Id = DR001

- text = Demonstrate the movements (figure-eight and back-and-forth movements) by controlling multiple autonomous robots from externally. Operator can switch between the autonomous mode and demonstration mode.

- Hardware architecture is already known, we use Roomba with PC that can control it using Wi-Fi and use Kinect to switch the mode.

**Provide demonstration synchronously**

Id = DR002

- text = Perform a synchronized demonstration of the movements (figure-eight and back-and-forth movements) by controlling multiple robots.

**Control of Roomba**

Id = DR004

- text = Use Roomba as a demonstration robot which can be controlled from externally and put the Wi-Fi enabled PC on it to control it.

**Operator control**

Id = DR003

- text = Operator can switch between the autonomous mode and demonstration mode.

**Autonomous drive**

Id = DR006

- text = Robot can run on the autonomous mode.

**Operated by Kinect**

Id = DR005

- text = Use Kinect to switch between autonomous mode and the demonstration mode.
req [Derived Requirements]

- **Control of Roomba**
  - Id = DR004
  - Text: Use Roomba as a demonstration robot which can be controlled from externally and put the Wi-Fi enabled PC on it to control it.

- **Demonstration by robots**
  - Id = DR001
  - Text: Demonstrate the movements (figure-eight and back-and-forth movements) by controlling multiple autonomous robots from externally. Operator can switch between the autonomous mode and demonstration mode.
  - Hardware architecture is already known, we use Roomba with a PC that can control it using Wi-Fi and use Kinect to switch the mode.

- **Operated by Kinect**
  - Id = DR005
  - Text: Use Kinect to switch between autonomous mode and the demonstration mode.

- **Operator control**
  - Id = DR003
  - Text: Operator can switch between the autonomous mode and demonstration mode.

- **Spiral drive**
  - Id = DR011
  - Text: Demonstrate the figure-eight movement.

- **Sharp turning**
  - Id = RR003
  - Text: It shall be capable of turning in a sufficient range at the conference hall’s demonstration space.

- **Speed control**
  - Id = RR004
  - Text: Speed of movement of robot should be controllable.

- **Forward/reverse motion**
  - Id = RR001
  - Text: Robot can move forward and reverse.

- **Back and forth**
  - Id = DR012
  - Text: Demonstrate the back-and-forth movement.

- **Stop immediately**
  - Id = RR002
  - Text: It shall stop the motion without an excess of the inertia when its mode has been switched.
req [Robot requirements]

- **Control of Roomba**: Use Roomba as a demonstration robot which can be controlled externally and put the Wi-fi enabled PC on it to control it.
- **Obstacle detection**: Controlling of negotiating obstacles is available only on autonomous mode.
- **Autonomous drive**: Roomba can run on the autonomous mode.
- **Forward/reverse motion**: Roomba can move forward and reverse.
- **Sharp turning**: Roomba shall be capable of turning in a sufficient range at the conference hall's demonstration space.
- **Stop immediately**: Roomba shall stop the motion without an excess of the inertia when its mode is switched.
- **Speed control**: Speed of movement of robot should be controllable.
- **Mode switching**: Operating mode of robots can be controlled from the receiver.
- **Wireless control**: Robots can be controlled wirelessly.
- **Provide demonstration synchronously**: Perform a synchronized demonstration of the movements (figure-eight and back-and-forth movements) by controlling multiple robots.

**Receiver**

**Robot**
req [Controller requirements]

- **Kinect**
  - Operated by Kinect
    - Id = DR005
    - text = Use Kinect to switch between autonomous mode and the demonstration mode.

- **Controller**
  - Provide demonstration synchronously
    - Id = DR002
    - text = Perform a synchronized demonstration of the movements (figure-eight and back-and-forth movements) by controlling multiple robots.

- **Operator control**
  - Id = DR003
  - text = Operator can switch between the autonomous mode and demonstration mode.
Robot Demo System

- Autonomous drive
- Move spiral
- Move back and forth

Operator

<!-- Block -->
<<external>>
Obstacle

<<block>>
<<system_of_interest>>
Controller

<<block>>
<<system_of_interest>>
Demonstration Robot
ibd [Demo system components]

```
<<interface>>
RobotService

+ stop() : void
+ autodrive() : void
+ drive(velocity : Long, radius : Long) : void
```

```
controller : Controller

servicePort

out : ~DriveVelocity

RobotService

in : DriveVelocity

openmRobot : Robot

servicePort

in : DriveVelocity

hrmRobot : Robot

RobotService
```
ibd [block] Robot [Physical structure]

RobotService

servicePort

distance : TimedDouble

receiver : Receiver

in : DriveVelocity

drive : DriveVelocity

serial

SCI

serial

Roomba
ibd [Physical structure of controller]
stm [State machine of controller]
Conclusion

• SysML “Block”s map to “RTC”s nicely.
• <<Satisfy>> relationships between “Requirements” and “Components” can be visualized to show the intentions of components reasonably.
• An Easy-to-use tool (Astah/SysML) boosted effectiveness of modeling.
• Communication between teams worked well using web-based model sharing feature of the tool.
Future Topics

• Real-time aspects into the model
• Relate Safety Case models (Software Assurance Case Model/Safe ML) with SysML models
• SysML Profile for RTC.
• Traceability and impact analysis from/to requirements to components via the tool.
We are exhibiting the demo, and tools. Please visit us.