Variability Modeling and Resolution in Component-based Robotics Systems

Luca Gherardi, Davide Brugali

University Of Bergamo, Italy

RoboCup@Work Camp - Sankt Augustin - May 09, 2012
Agenda

- Application and Application Family
- Feature Models
- Process for modeling and resolving the variability
  - Application family design
  - Variability modeling
  - Variability resolution
- Live demo
- Conclusions and future works
Application family

- **Given one or more libraries of components**
  - Model (e.g. BRIDE, …)
  - One or more implementations conform to the model

- **We define application family:**
  - A set of applications that can be built by using a set of components taken from the libraries
Application family

- These applications differ in a set of variation points:
  - Which components are used (mandatory and optional)
  - How these components are configured:
    - How their interfaces are implemented
    - How their parameters value are set
  - How these components are connected

- All these possible choices are called variants
  - For each variation point more variants are available
The resolution of all the variation points produces a specific application and we call it **application configuration**

- According to the configuration the application will satisfy different requirements

**Problems:**

- Not all the possible combinations of variants are allowed
- High # of components -> huge # of possible application configurations

**The possibility of symbolically associate the available configurations to the application requirements is really important.**
We propose Feature models as a formalism for modeling the variability of an application family

Feature models
- Allows the representation of all the possible configurations of an application family
- Captures the commonalities and variabilities among these applications

Feature diagrams
- Graphical representation of the feature models
Feature models concepts

Feature: an end-user characteristic of an application
- Mandatory and optional features
- Containments: one-to-one, Or, Alternative
- Explicit constraints: requires and excludes
- Instance

- Motion Planning
  - Path Planner
    - PRM
  - Collision checker
  - 3D environment model
    - RRT
The process that allows the developer to model the variability of a family of applications and to resolve it consists of three steps:

- Application family design
- Variability modeling
- Variability resolution

Here the process is presented by using 2 eclipse tools:

- BRIDE (BRICS Integrated Development Environment)
- Feature Model Plugin
The developer defines a template for a family of applications by using BRIDE
- All the components
- A set of connections (default connections)

The template can be transformed in a model for a specific application by defining:
- Optional connections
- Implementations of the components
- Values of the configuration parameters
Application Family Design

Diagram showing the relationships between DepthCamera, MeshGenerator, and FilteringComponent.
Application Family Design
The developer defines the feature model that represents the family of applications by using our Feature Model Plugin.

- Each feature represents a possible requirement of an application and it is associated to a set of transformations.
- An instance represents a specific application which satisfies a set of requirements.
Feature Models Meta-Model
Variability Modeling

Diagram:
- Perception
  - Mesh Generation
    - Marching Cubes
    - Delaunay Triangulation
  - Filtering
  - Camera Resolution
    - Low
    - Medium
    - High
The transformations define how the template has to be modified in order to produce the model of a specific application

- Implementation transformation
- Connection transformation
- Property transformation
Feature Models Meta-Model
Variability Modeling

RoboCup@Work Camp
Sankt Augustin, Germany
May 9, 2012
Variability Modeling

![Variability Modeling Diagram](image-url)
Variability Resolution

- The developer creates an instance of the feature model by selecting the requirements of his application.
- The plugin checks that all the constraints are satisfied.
- The plugin generates the model of the desired application.
Variability Resolution
Variability Resolution

Property

<table>
<thead>
<tr>
<th>Core</th>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Description</td>
<td>Camera Resolution</td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td>Resolution</td>
</tr>
<tr>
<td></td>
<td>Type</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>1000x600</td>
</tr>
</tbody>
</table>

DepthCamera

BRICS_3D::Hardware = CameraDriver

Resolution

cameraPointCloudOut

Properties

Operations
Conclusions and future works

- We presented a process for modeling and resolving the variability of an application family
  - Specification of variation points and variants
  - Specification of constraints between variants
  - Possibility of automatically creating the application-model by specifying only its requirements

- Future works will regard the integration of the Feature Models with BRIDE-ROS
Thank you!