An Introduction to the Theory and Application of Domain-Specific Modeling Languages

Jonathan Sprinkle
University of Arizona

This work is supported by the United States Air Force Office of Scientific Research.
• In the end:
  – We can accomplish more with tools, than with elbow grease
• In the beginning:
  – Our tools come from waste, or leftovers
• In the end:
  – We can use tools to create new tools that are better than the first tools!!
Domain-Specific Modeling: An abstract perspective

Domain Concepts

Unrestricted Implementation
Domain-Specific Modeling: An Abstract Perspective

Domain Concepts

Defns of Domain Assumptions and Givens
Domain-Specific Code Generation

DS Code Generator

Domain “Instance”
Domain-Specific Modeling

- Create *model* of the system
- Perform
  - Analysis
  - Architecture exploration
  - Simulation
- Generate
  - Configuration
  - Code
  - Executables

*From the same models!*

Example Domains & Environments:
- VLSI Layout (e.g., Altera)
- Engg Drawing (e.g., AutoCAD)
- Physical Modeling (e.g., SolidWorks)
- Signal Processing (e.g., LabVIEW)
- Controls (e.g., Simulink)
While Event $e_i$, and in State, $s_c$
After, $e_i$.delay, and in State, $s_c$,
Stop clock
If exists Transition $t_e$: ($src=s_c$, $dst=s_n$), set $s_c = s_n$
Else if $s_c$.parent=null, set $e_i = e_i.amSrc.sequence.dst$
Else transition through $s_c$.parent
Advance clock
Formal Definition of a Domain-Specific Language

\[ L = \langle C, A, S, M_C, M_S \rangle \]
<table>
<thead>
<tr>
<th>Archetypal Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>Specific classes of entities that exist in a given system or domain. Domain models are entities themselves and may contain other entities. Entities are instances of classes. Classes (thus entities) may have attributes.</td>
</tr>
<tr>
<td>Associations</td>
<td>Binary and n-ary associations among classes (and entities).</td>
</tr>
<tr>
<td>Specialization</td>
<td>Binary association among classes with IS-A semantics.</td>
</tr>
<tr>
<td>Hierarchy</td>
<td>Binary association among classes with “aggregation through containment” semantics. Performs encapsulation and information hiding.</td>
</tr>
<tr>
<td>Constraints</td>
<td>A binary expression that defines the static semantic correctness of a region of the model: if the objects of the region are “correct,” the expression evaluates to TRUE.</td>
</tr>
</tbody>
</table>
• MIC helps unify the specification, and the final implementation

• DSMLs provide an interface close to the language of the domain experts

• Common abstractions permit a few metamodeling concepts to be applied to myriad domain specifications


We are always looking for good graduate students.

http://ece.arizona.edu/