The Generative Software Problem

By: Jon Leigh
Outline

● 5-10 min summary of the paper

● Generative Software Development vs Other Software Reuse Methods

● Practicality
Summary

- Domain Specific Languages (DSL)
- Domain Engineering vs Application Engineering
- Generative Domain Model
- Configuration and Transformation Views
- Network of Domains
- Model Driven Development
Summary

- Domain Specific Languages (DSL)
- Domain Engineering vs Application Engineering
- Generative Domain Model
- Configuration and Transformation Views
- Network of Domains
- Model Driven Development
Domain Specific Languages

- Are part of Domain implementation
- Address a different problem domain than Generic Specific Languages
- Can be text based languages, embedded in languages (C++ templates), to graphical style design (Bouml Code Generation)
- In terms of Generative Software Development are used to describe the problem
DSL Example - Survey DSL

Q1 Are you batman?
  1 Yes
  2 No
Q2 Where does the muffin man live?
  1 I don't know
  2 On Elm Street
  3 Durry Lane
Q3 Which OS do you prefer?
  1 Windows
  2 Linux
  3 Mac OSx
Summary

- Domain Specific Languages (DSL)
- Domain Engineering vs Application Engineering
- Generative Domain Model
- Configuration and Transformation Views
- Network of Domains
- Model Driven Development
Fig. 1. Main processes in system-family engineering
We've seen this before...
And this before as well...
Summary

- Domain Specific Languages (DSL)
- Domain Engineering vs Application Engineering
- Generative Domain Model
- Configuration and Transformation Views
- Network of Domains
- Model Driven Development
Fig. 2. Mapping between problem space and solution space
Summary

- Domain Specific Languages (DSL)
- Domain Engineering vs Application Engineering
- Generative Domain Model
- Configuration and Transformation Views
- Network of Domains
- Model Driven Development
Fig. 3. Configuration view on the mapping between problem space and solution space
Transformation View

Fig. 4. Transformational view on the mapping between problem space and solution space
Summary

- Domain Specific Languages (DSL)
- Domain Engineering vs Application Engineering
- Generative Domain Model
- Configuration and Transformation Views
- Network of Domains
- Model Driven Development
Network of Domains

Fig. 5. Different arrangements of mappings between problem and solution spaces
I DON'T WANNA BE A PRODUCT OF MY ENVIRONMENT

I WANT MY ENVIRONMENT TO BE A PRODUCT OF ME
Summary

- Domain Specific Languages (DSL)
- Domain Engineering vs Application Engineering
- Generative Domain Model
- Configuration and Transformation Views
- Network of Domains
- Model Driven Development
Model Driven Development

- Model is supposed to be actual software
- Example: UML to a DSL (Theoretical)
Outline

● 5-10 min summary of the paper

● Generative Software Development vs Other Software Reuse Methods

● Practicality
Fig. 10. Relationship between generative software development and other fields (from [47])
Who will use Generic Software Development?

- The big guys like NASA, government, Microsoft, etc
- Little guys may use the big guys tools, but probably won't implement their own
Disadvantages

- Complex
- Must have a significant amount of products in a software family to see a return on investment
- Are not customizable (they are meant for a specific domain so this makes sense)
Outline

- 5-10 min summary of the paper
- Generative Software Development vs Other Software Reuse Methods
- Practicality
Real World Examples

- Instrument Software System
- Integrated Development Environments (Netbeans, Visual Studio)
- Tools based off Domain Specific Languages (like the drawing tool we saw last time)
Any Questions?