Karen N.L. Li, University of Auckland

Monday, 16 November 2009, ASE-VASE-SPPI

AUGMENTING DSVL META-TOOLS WITH PATTERN SPECIFICATION, INSTANTIATION AND REUSE

Overview

- MaramaDSL - Marama extensions in *Microsoft DSL Tools* platform
- Design-for-reuse and design-by-reuse via patterns
- Pattern specification in meta-model
- Domain specific patterns vs. generic DSVL patterns
- Towards DSVL pattern language and knowledge base

Background

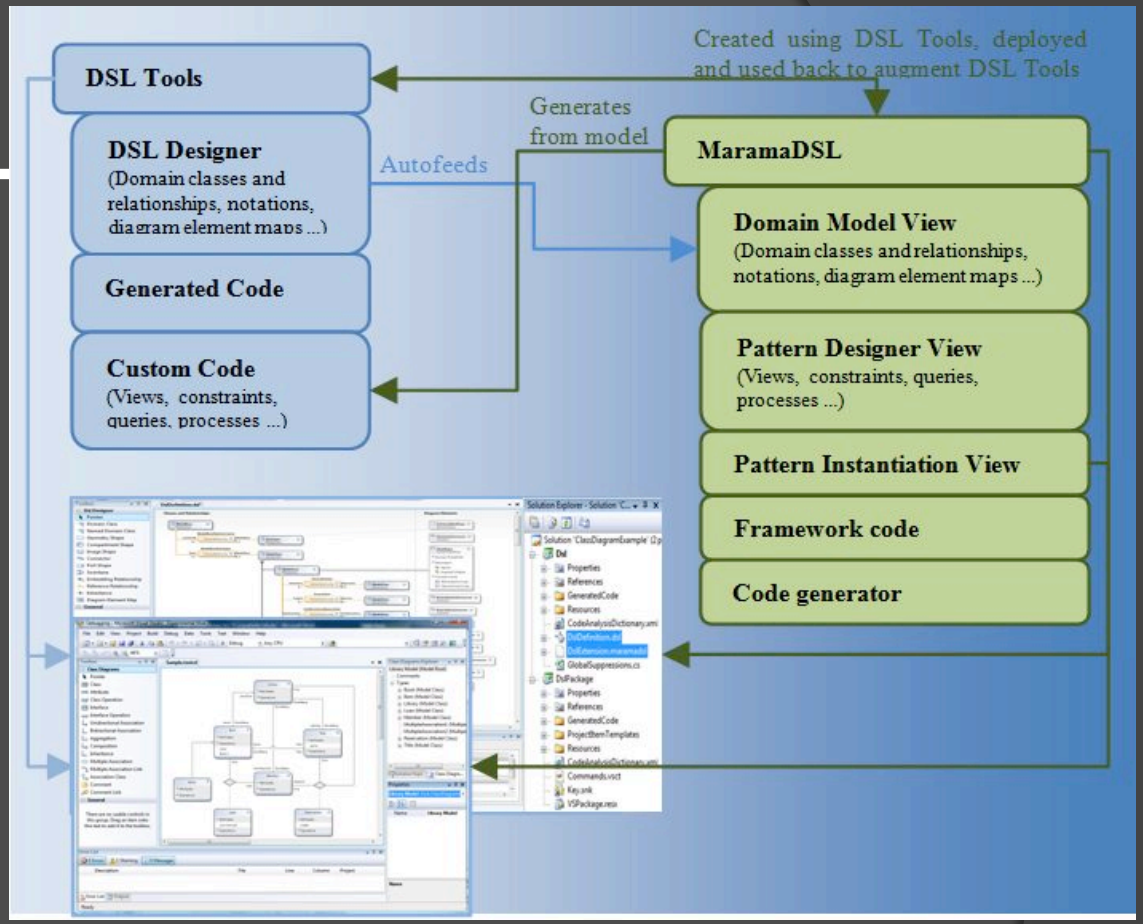
- DSVLs (a.k.a. DSMLs, DSLs)
  - Raising the level of abstraction of knowledge representations in domain-specific software applications
  - Tool (meta-tool) support for
    - Visually specifying DSVL meta-model (domain model, notations, views, constraints, critics, event handlers, model transformation ...)
    - Integration of multiple DSVLs within the same domain model
    - Automatically generating DSVL environments based on specifications
- Challenges for end users
  - Behavioural specifications hard
  - Multiple meta-DSVL metaphors used for structural and behaviour aspects
  - Lack of reuse support

Motivation

- Common recurring problems and solutions exist in DSVLs
  - Model element composibility, cardinality, mutability, multiple views and their interoperability
  - Model organisation and constraints, value dependency, queries and workflows, diagram layout management, importing/exporting, tracing and debugging
- Design-for-reuse and design-by-reuse meta-patterns

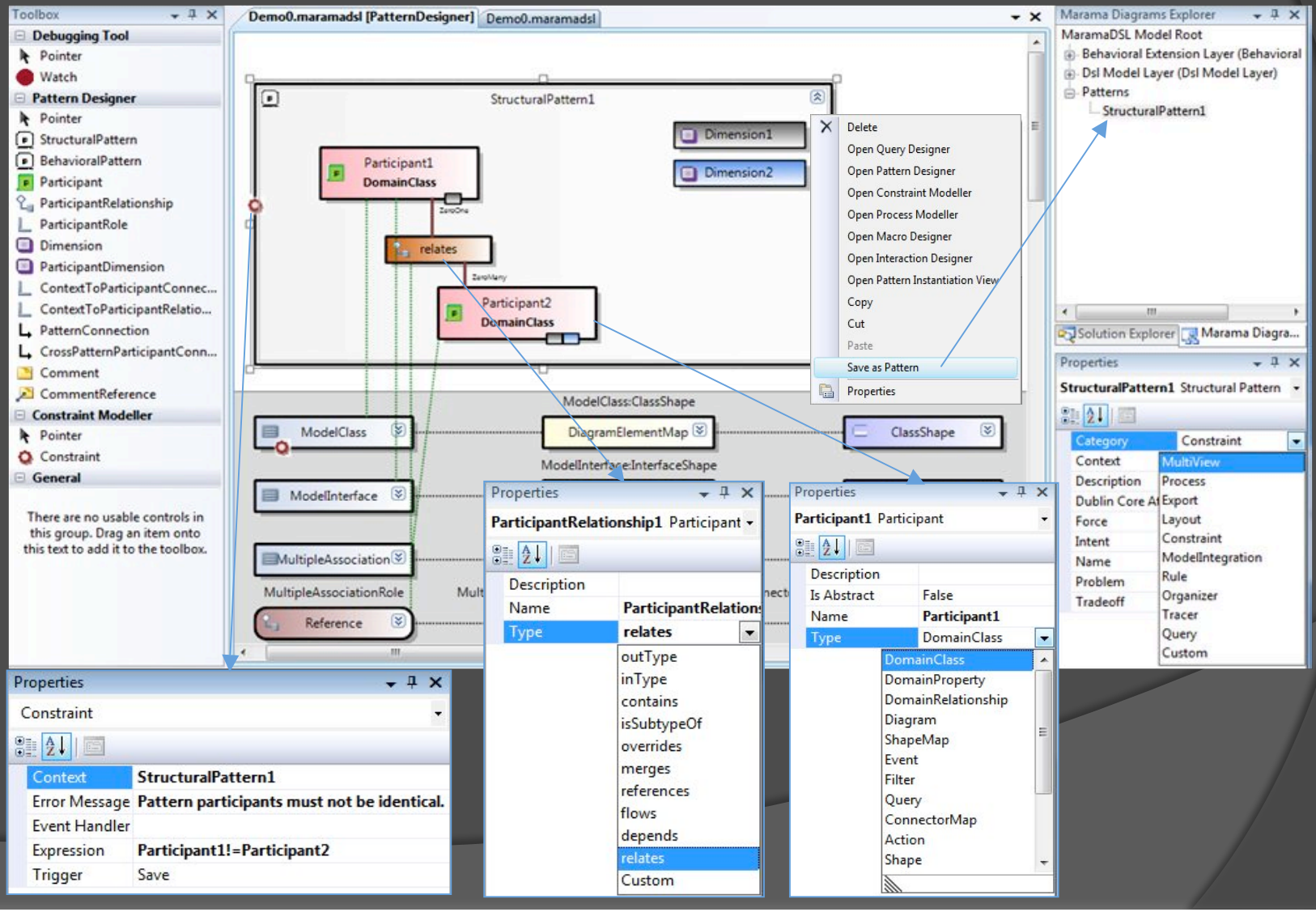
- The need for a pattern modelling language in parallel with DSVL meta-model
- The need for pattern instantiation tool support (should help with creation) for DSVL model instances
- High level pattern-oriented cross-domain reuse optimistic/optimal approach to remove barriers to use

MaramaDSL

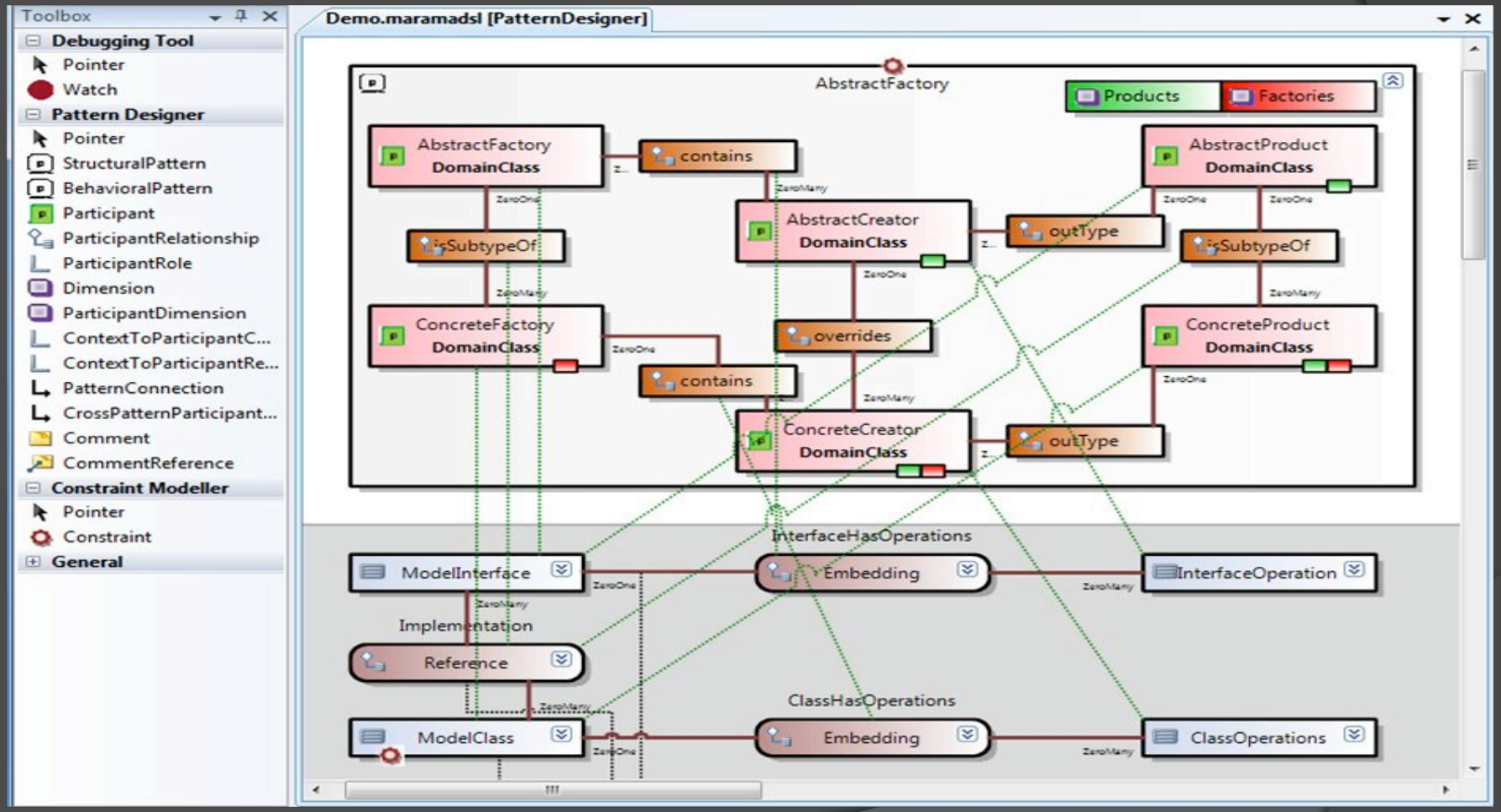


- *Microsoft DSL Tools* based meta-tool
- Complements *DSL Designer* with multiple designer views
- Working together with *DSL Tools* integrating pattern specification and usage into DSVL development process

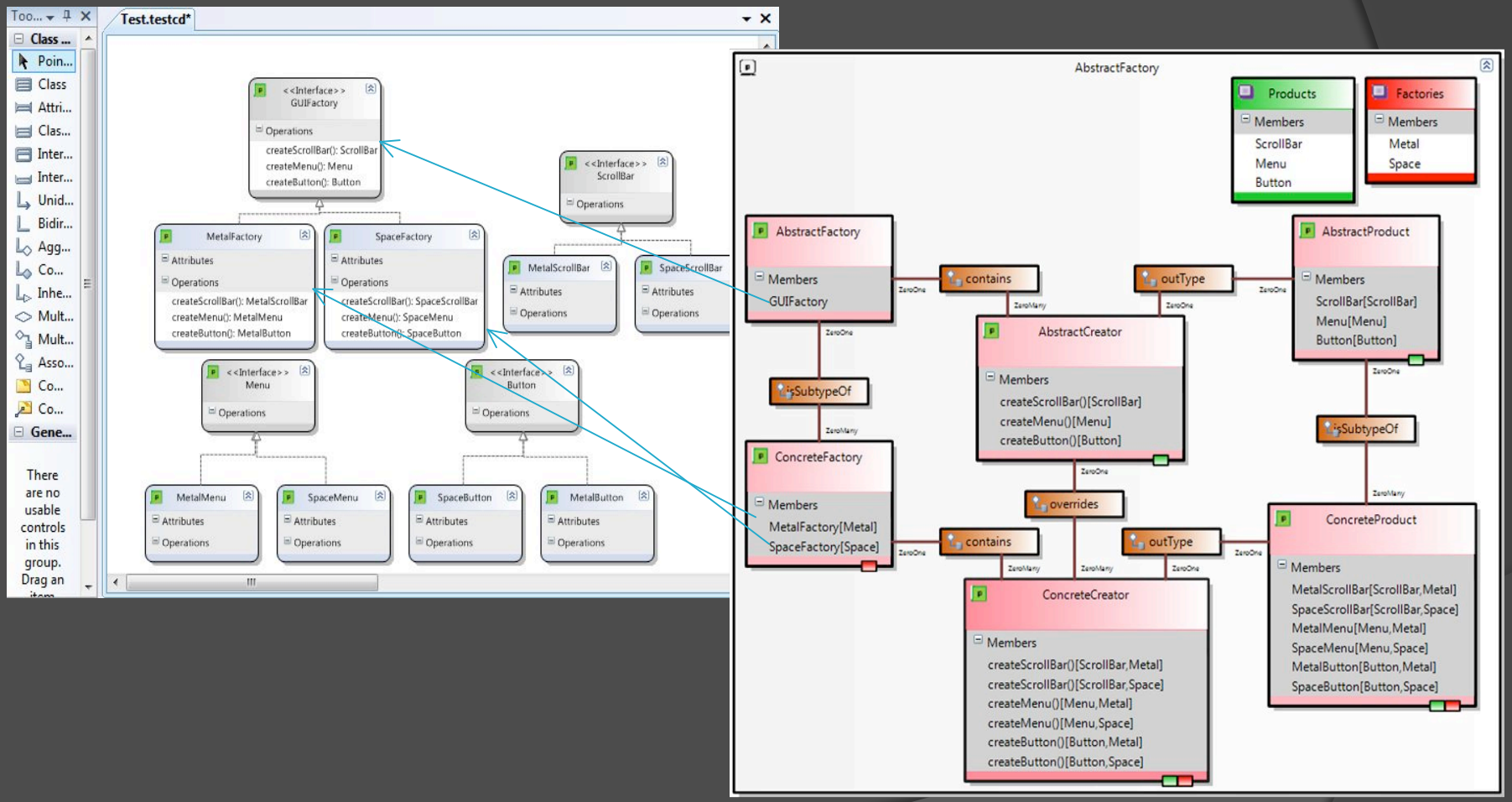
Pattern specification



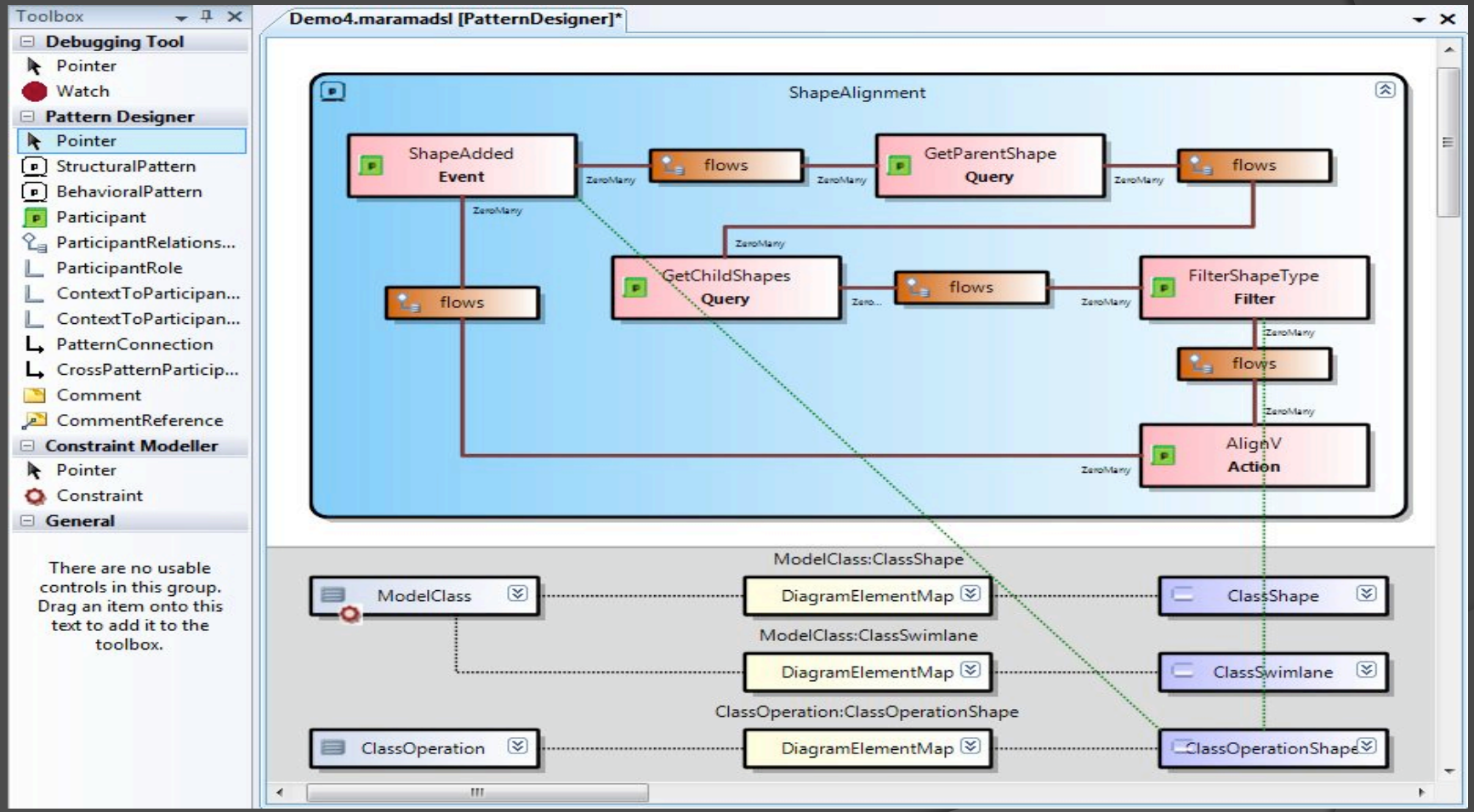
Abstract Factory specification on UML meta-model



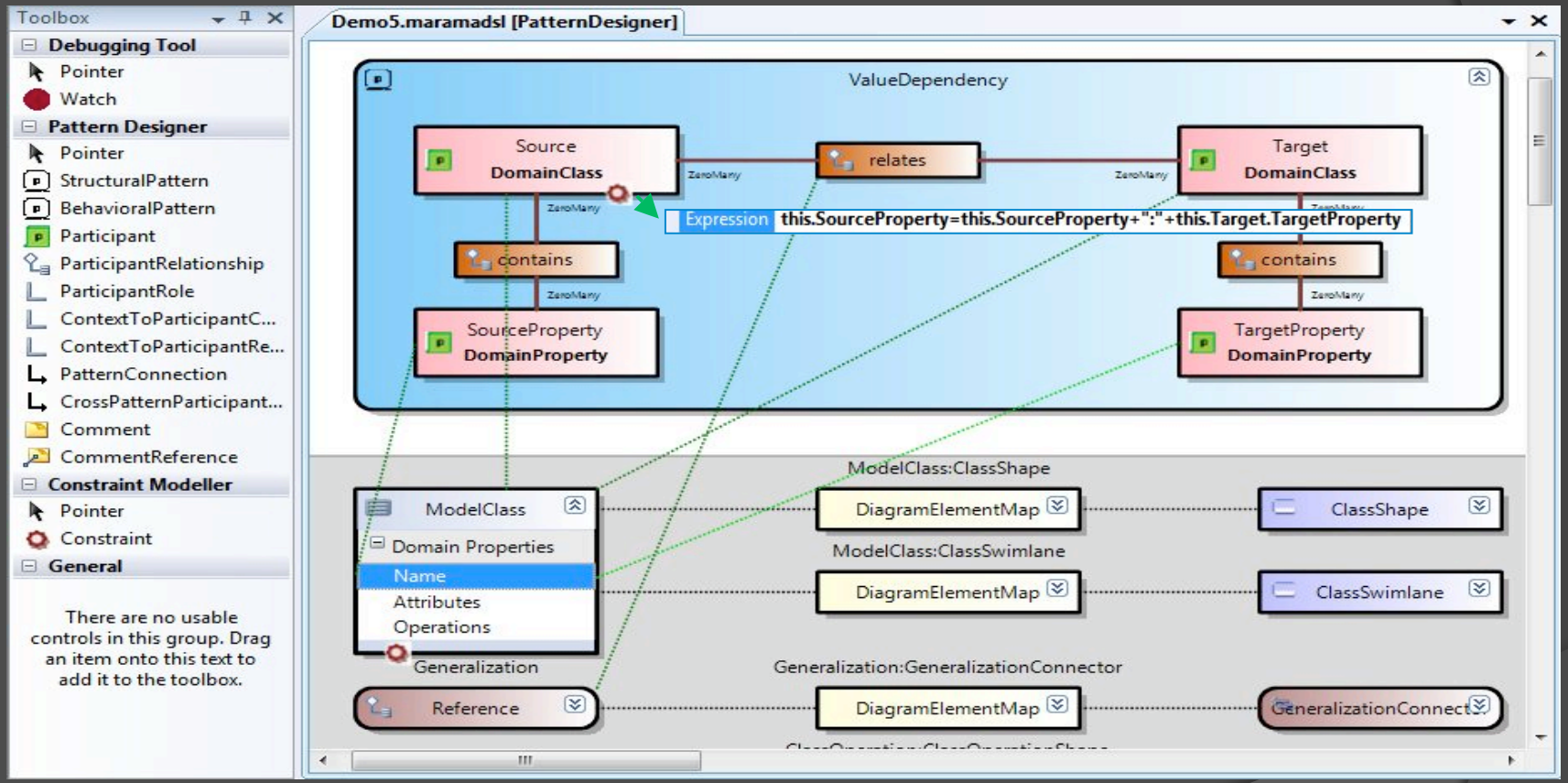
Abstract Factory instantiation on UML model instance



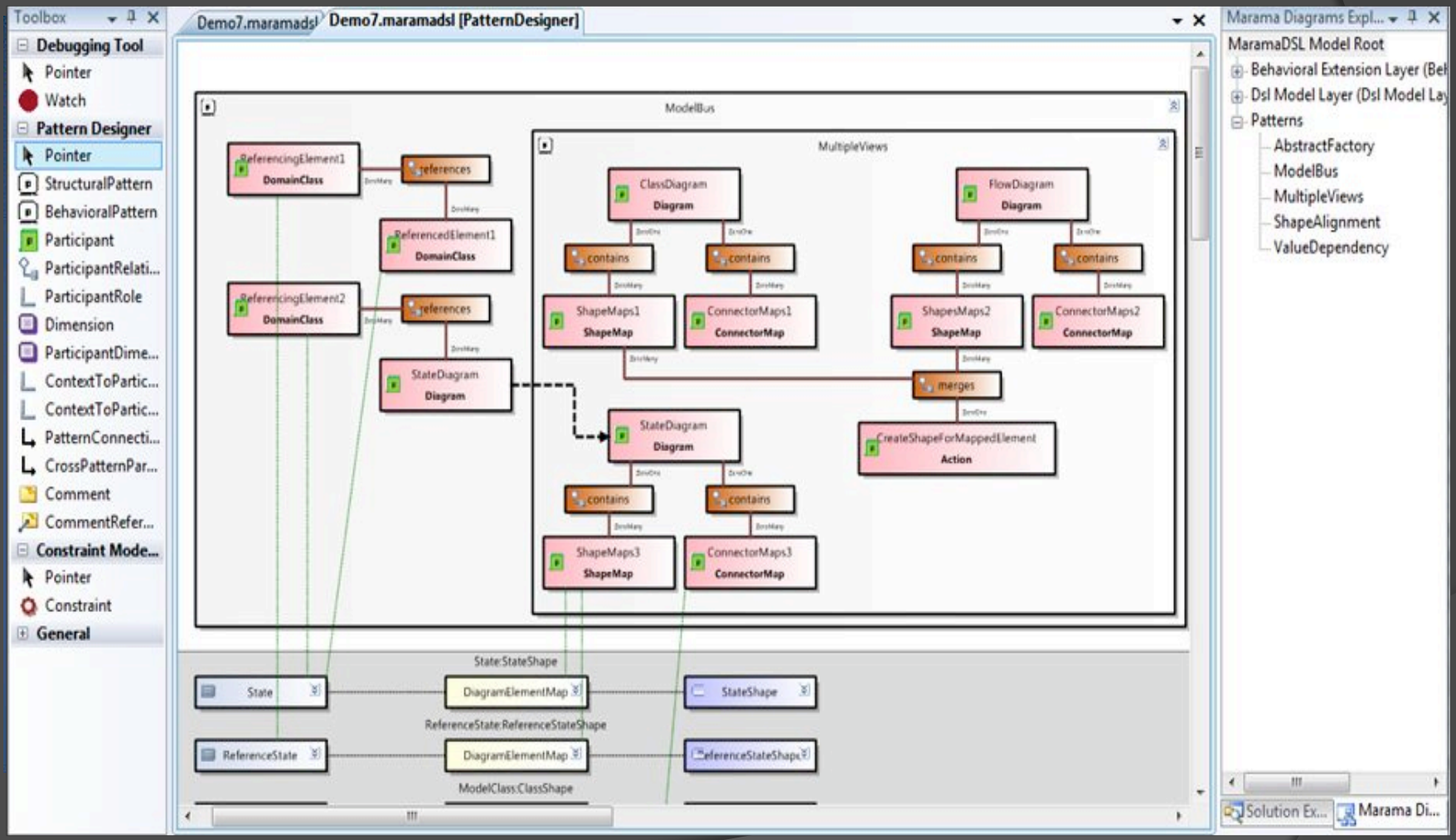
Generic DSVL patterns – Visual Constraint



Generic DSVL patterns – Model Constraint



Pattern composition – towards pattern language



Conclusion

- Ultimate goal – facilitate sharing of design knowledge among DSVL communities
  - Pattern validation (completeness, consistency, soundness)
  - Force analysis/balancing of patterns in Pattern Language
  - Design decision support via higher level visual metaphor patterns
  - Automatic capture of design knowledge
  - Pattern publish and discovery – use Jens' WOP OWL/RDFs