Karen N.L. Li, University of Auckland

Monday, 17 November 2009, ASE-KISS

A GENERALISED EVENT HANDLING FRAMEWORK

Karen Li, John Hosking, John Grundy

Overview

- Earlier developed 3 exemplar DSVLs for event-based system specification
  - ViTABaL-WS: web service compositions at high abstraction level
  - Kaitiaki: diagramming-based design tool interactions at low-medium abstraction level
  - MaramaTatau: declarative meta-model structural dependencies and constraints at mixed low/high abstraction level
- Generalised to an integrated visual approach for event handling specification
  - Derived a canonical event behaviour model
  - Enabled interoperability between the exemplar event models
  - Supported synthesized runtime visualisation

Background

- The widely used event-driven software paradigm
  - Publish/Subscribe relationships between components
  - Flexibility for constructing dynamic interactions
  - Loosely-coupled system behaviours
  - Events, conditions (“filters”) and actions
  - Workflow management systems; database systems; distributed computing; graphics and modelling frameworks; software tools e.g. meta-tools
- Difficulty in specifying and understanding the execution of event-driven systems
  - Complexity of behaviours
  - Low-level abstraction with implementation details
  - Not suitable for end users to specify

Related techniques

- Declarative rules, functions and constraints
  - Declarative semantics allow users to concentrate on high level relationships
  - Textual rule-based languages
  - Complex behaviour specification/visualisation often suppressed
- States and flows
  - Easy analysis of runtime changes
  - Sacrifice system structural details
  - Low level details raising scalability issues with highly concurrent systems
- Program-by-demonstrations
  - Focus on dynamic behavioural changes and visualisations
  - Limited in specification power
- Hybrid visual/textual approaches

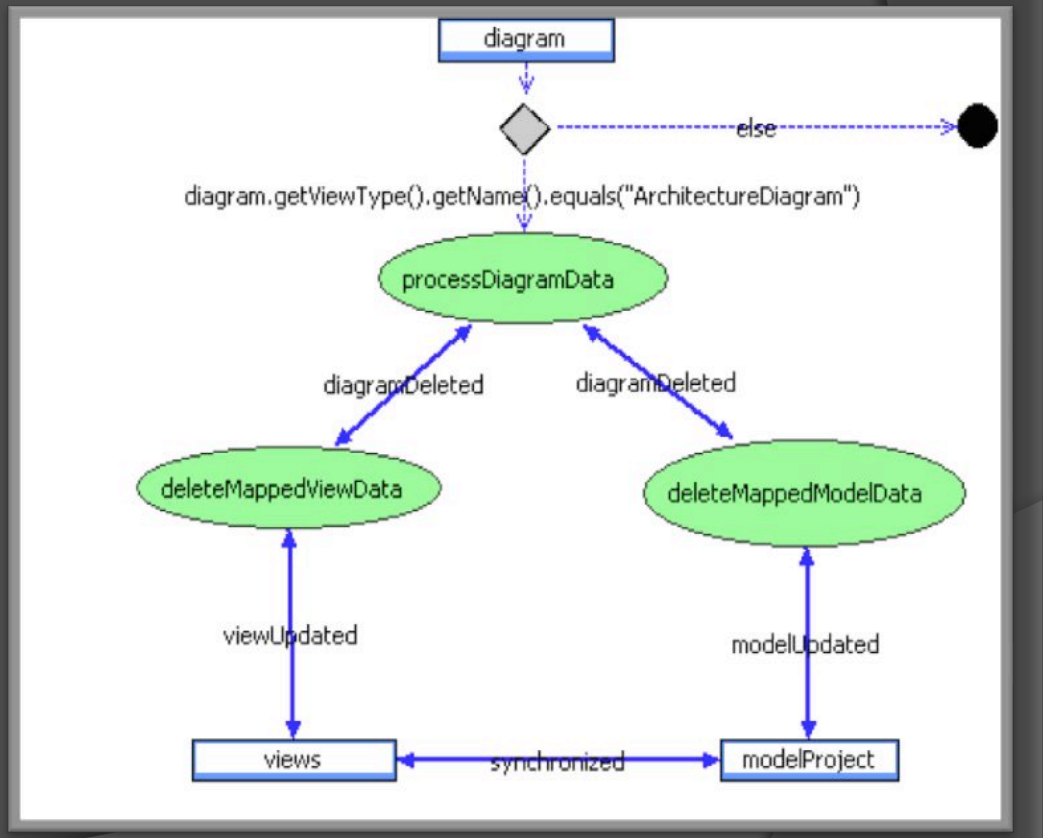
Objectives

- Explore different visual metaphors for event handling specifications
- Generalise them to an integrated framework
- Provide tool support for the integration/interoperability of event handling specifications
- Provide integrated visualisation support for tracing/debugging event propagations at runtime

- The generalised framework should be able to adapt the event-based communication model to a wide range of application domains, and also support complex and interactive system design and implementation

TA metaphor from ViTABaL-WS

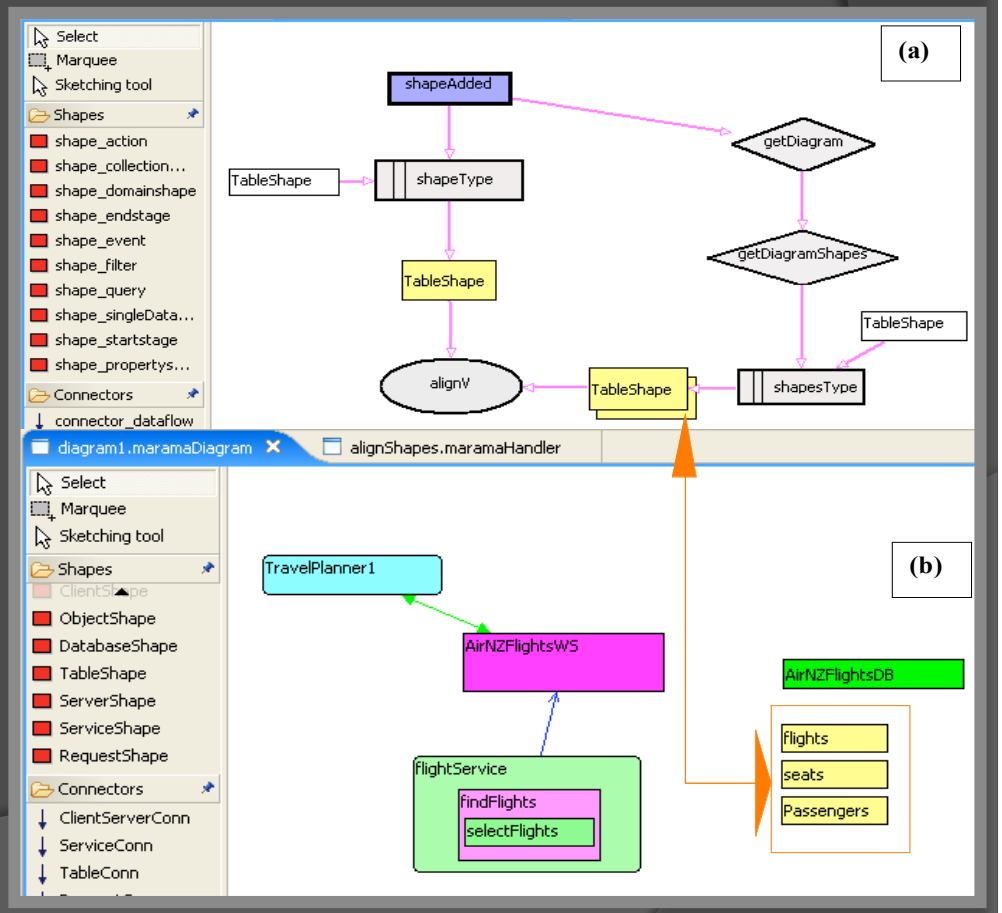
- Tool Abstraction, to describe event-based inter-connections between abstract components (encapsulating event response details)
- High level abstraction



- Separated concerns in data and function encapsulations: events, event generators/receivers, responding behaviours
- Data, control and event flows

EQFA metaphor from Kaitiaki

- Event-Query-Filter-Action, providing visual design level notation for specifying diagram/GUI event handling mechanisms
- Handle both state-change and action events
- Intermediate level abstraction

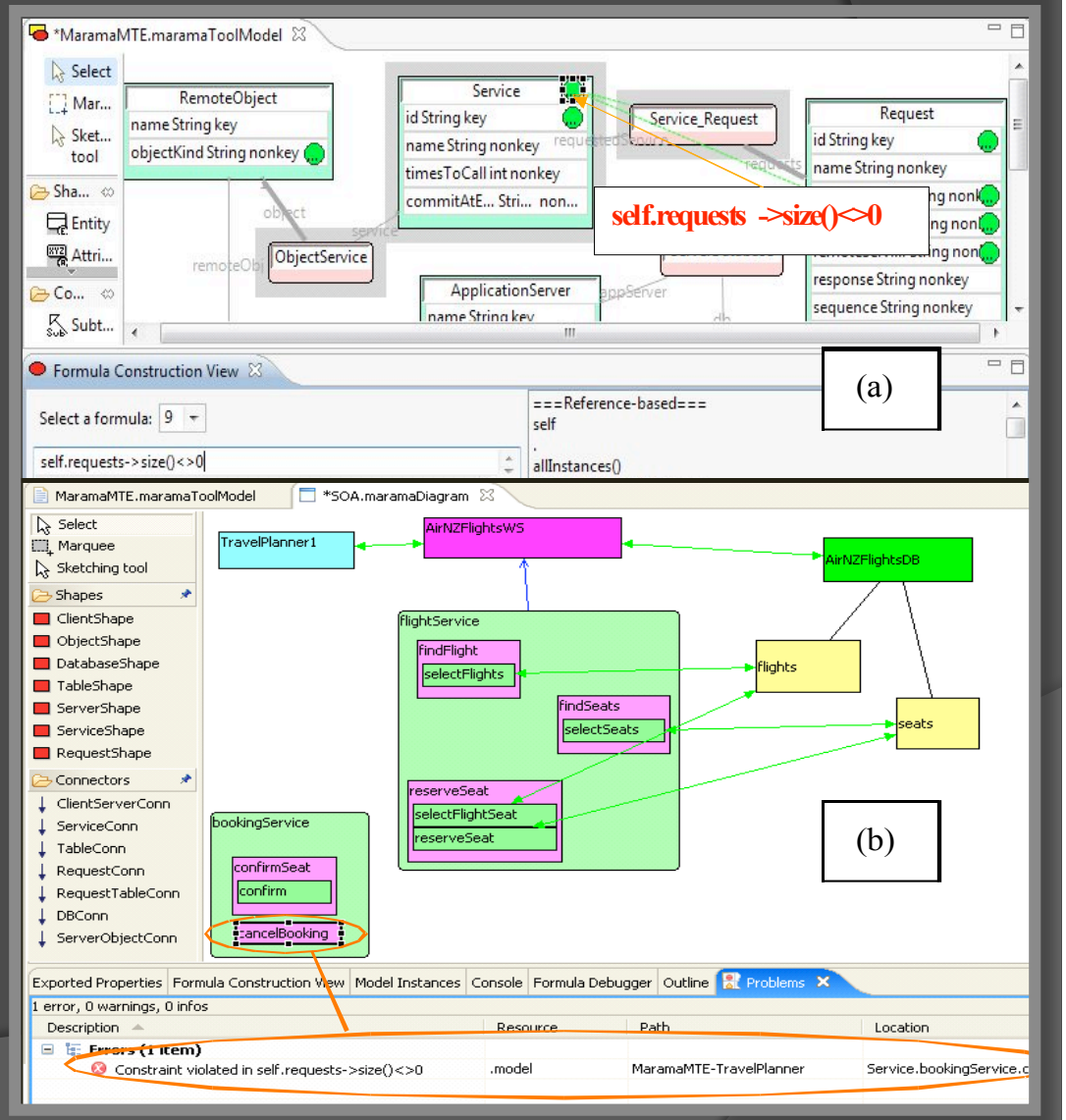


- Reusable, discrete building blocks for composition
- Explicit data flows
- Push and pull triggering
- Domain diagram elements involved as placeholders for tool state

Spreadsheet metaphor from MaramaTatau

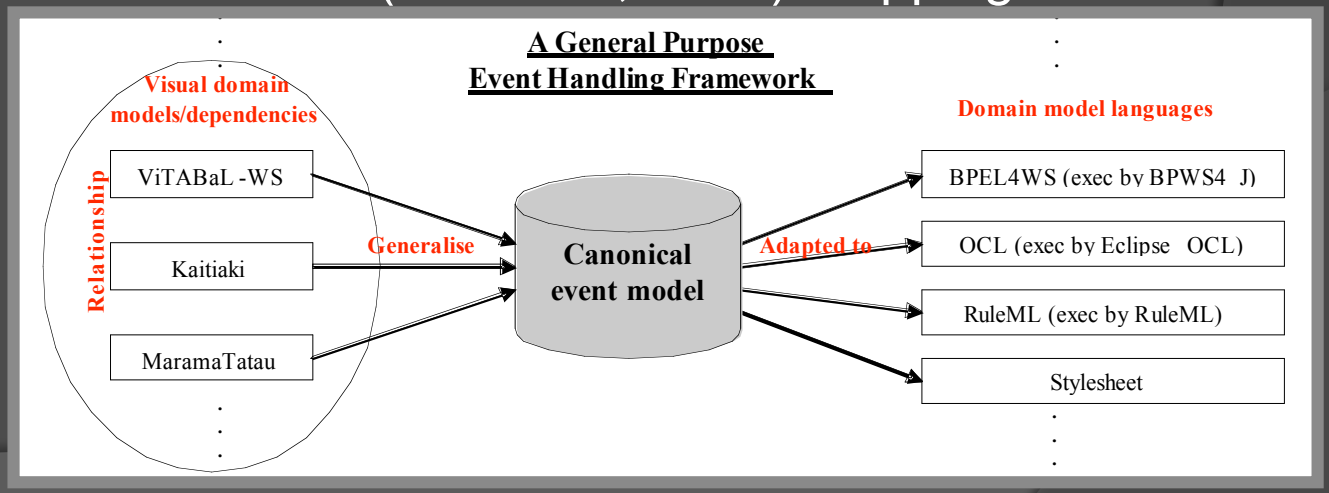
- Specify model level dependencies and constraints at meta-model
- Handle state-change events
- Mixed low/high level abstraction

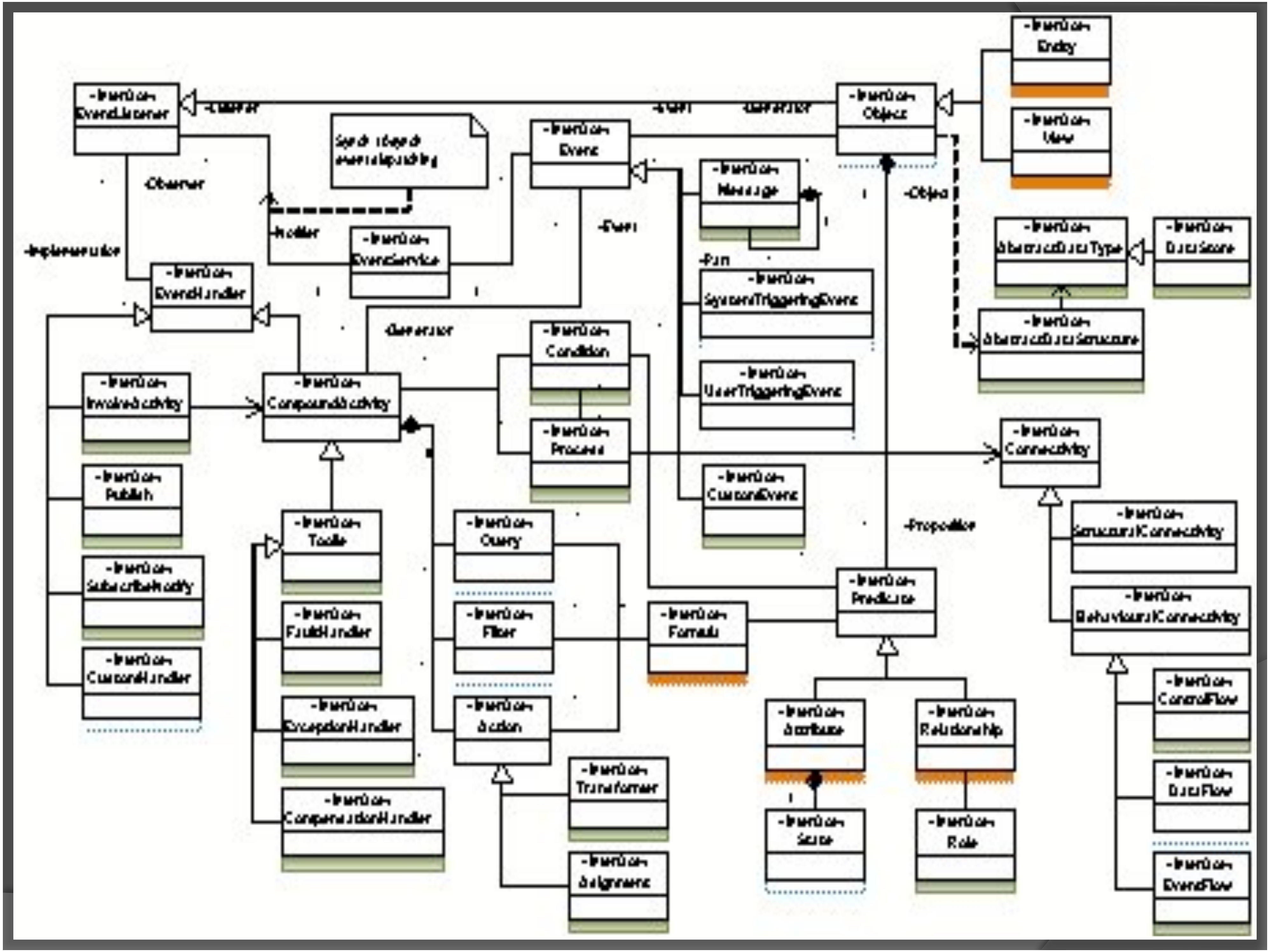
- Overlay declarative formula/ constraint annotation
- Spreadsheet-like OCL construction
- Integrated validation mechanism



Approach to generalisation

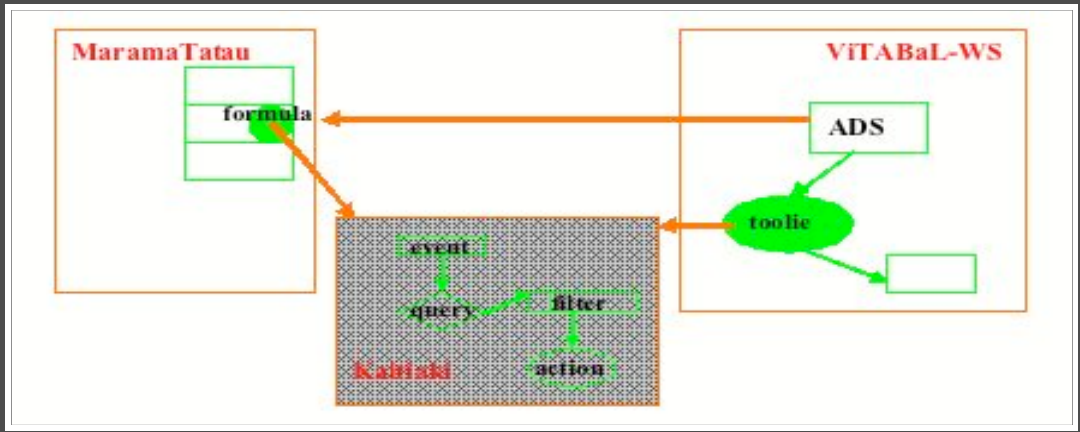
- Evolving Frameworks Pattern Language (Roberts and Johnson, 96)
  - The Three Examples pattern for establishing a framework
- Identified common abstraction
  - Combined atomic primitives extended by the 3 exemplar VLs
  - Removed redundancies
  - Added bridging elements
- Reserve metaphoric views in the style of the 3 exemplars
- Allow mapping between related concepts in each metaphor for model transformation – MaramaTorua (Huh et al, 2007) mapping specifications



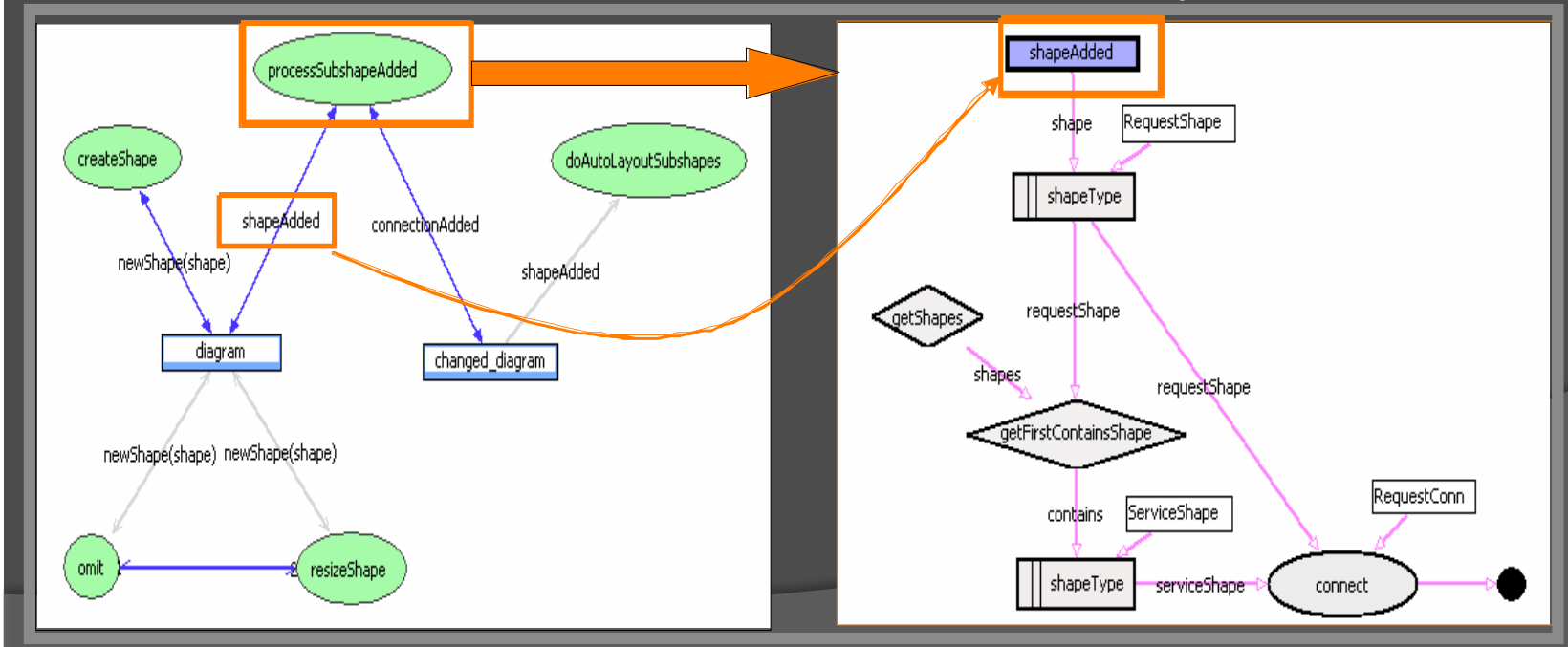


EventListener
Object
Entity
View
Event
Message
EventService
EventHandler
SystemTriggeringEvent
UserTriggeringEvent
CompoundActivity
Condition
Process
InvokeActivity
Publish
SubscribeNotify
CustomHandler
Tools
FaultHandler
ExceptionHandler
CompensationHandler
Query
Filter
Action
Transformer
Assignment
Formula
Predicate
Attribute
Relationship
State
Role
CustomEvent
Connectivity
StructuralConnectivity
BehaviouralConnectivity
ControlFlow
DataFlow
EventFlow
DataStore

Interoperability



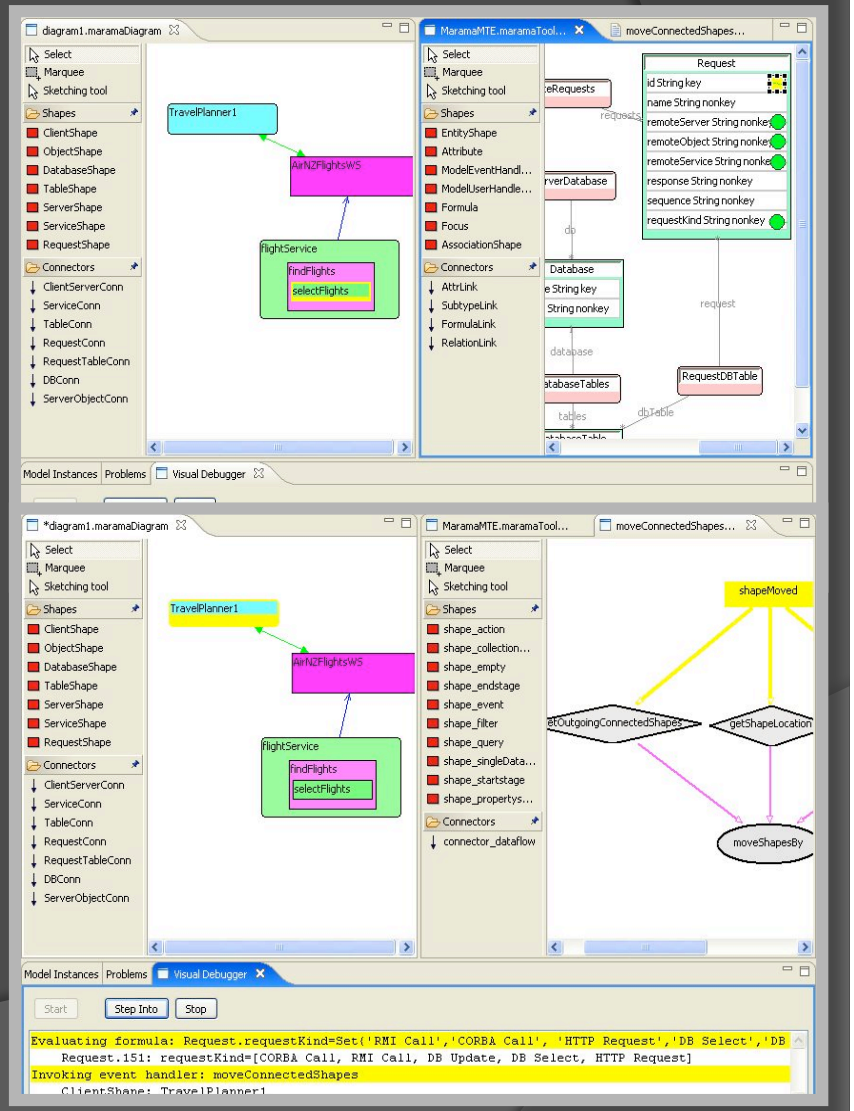
- TA metaphor is used to define high-level abstract data structures and functions and their coordination
- Abstract data structures are further constrained using formulas
- Abstract functions are further refined referencing EQFA specs
- Formulas can also reference EQFA specs



Synthesised visual debugging

- Tool support for tracing and visualising event propagations and their effects
- Visualise both static dependency structure and dynamic event handling behaviour
- User controlled step-by-step visualisation
- Reuse design-level abstraction in runtime visualisation

- Aim to represent visual debugging at a high abstraction level, based on user-defined queries in a visual query language



Framework evolution

- Both black-box and white-box framework
- Reuse and extension via both inheritance and composition
- Evolve framework by abstracting from additional examples to make it more general
  - Our next step is to integrate BPMN
  - Examine what abstractions to reuse from the canonical model
  - Examine new features and support to add

Evaluation concerns

- Multiple VLs at different abstraction levels being used
  - Hard mental operations and steep learning curve, trading-off for specification accessibility and flexibility
- Higher level description of the metaphors and guidelines needed

- Aim to develop a higher-level visual notation to describe interoperable/integrated specifications
  - High level integrated event behaviour model
  - Critics, layout, automation, guidance and round-trip engineering

Conclusion

- A synergy of 3 DSVLs for event handling specification in Marama meta-tools
- Canonical representation of event handling specifications enables further behavioural paradigms to be easily integrated into the framework