Extending xADL with Statechart Behavioral Specification

Leila Naslavsky, Lihua Xu, Marcio Dias, Hadar Ziv, and Debra J. Richardson

Motivation

- ❖ Software Architecture in Dependable Systems
  - – Design (guide, analysis and evaluation of system)
  - – Implementation (conformance testing)
  - – Integration (conformance and reliability analysis)
  - – **Operation (residual testing and analysis)**
  - – **Maintenance (regression testing)**

- ❖ Architecture-based Analysis and Testing
  - – Structural and Behavioral Architecture Descriptions
  - – Structural and Behavioral Architecture Analysis Tools
- ❖ Behavioral Description on Architecture Description Languages
  - – Not available in many ADLs
  - – When available, use the particulars of specific ADLs
- ❖ xADL: standard and extensible XML-based representation for Sw. Arch.
  - – "vanilla" xADL lacks support for behavioral description

Approach

❖ Associate UML Statecharts with architecture elements
 – Adapt UML statecharts conceptual model for software architecture

❖ Build upon xADL and an existing analysis tool (Argus-I)
 – Augment xADL with behavioral specifications (e.g. statecharts)
 – Refactor Argus-I to support xADL

Contributions and Status

❖ Conceptual model and XML schema for Statecharts

❖ Augment component-behavior modeling in xADL to support XML representation of statecharts

❖ Work in progress
  – Refactor Argus-I to utilize this representation of statecharts instead of the current extension for C2 architectures
  – Extend both Argus-I and xADL tools to further enhance their integration, intend to improving specific test techniques

Background and Related Work within our Research Group

- ❖ xADL 2.0 (Architecture Research Group – UCI)
  - – Designed for developing various architecture types
  - – Extensible
  - – Lack of semantic definition of component behavior
- ❖ Argus-I (Testing and Analysis Research Group - UCI)
  - – Specification-based analysis tool
  - – Focus on both the component and architecture levels
  - – Support both structural and behavioral analysis
  - – Rely on a specific ADL and its representations

- ❖ **RTMC (Regression Testing via Model Checking) – Lihua Xu**
  - – Architecture-based testing tool
  - – Generation of regression test cases based on formal specifications
  - – Model checker as part of test generation tool
- ❖ **Component and Object Level Residual Testing – Leila Naslavsky**
  - – Test coverage criteria over call-graph and design models

Statechart Conceptual Model
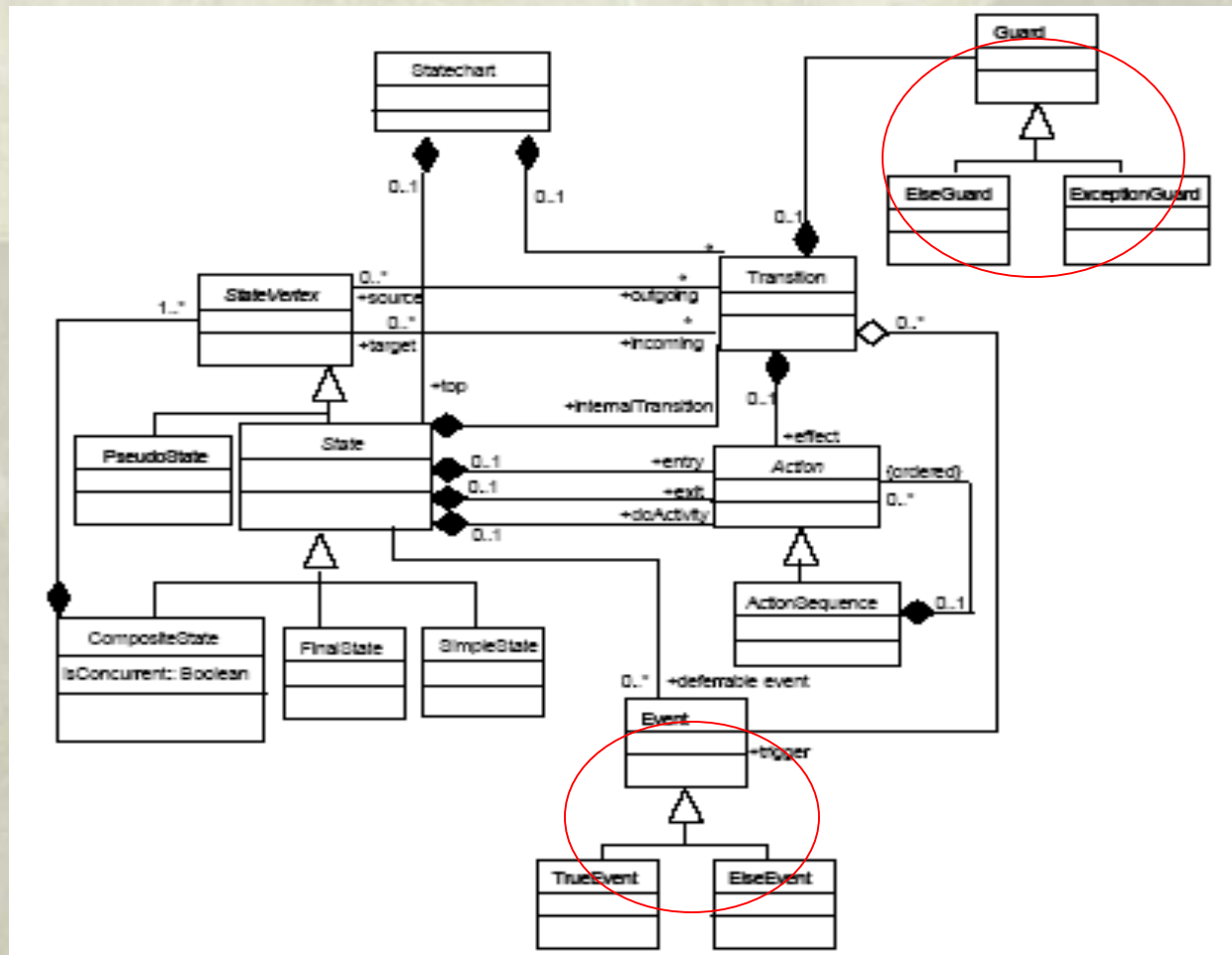
Statechart XML-Schema

❖ Express conceptual
model as xml-schema

❖ Main element:
- *StateChart*

❖ Included elements:
- *top* state
- *transition*

```
<xs:element name ="StateChart">
  <xs:complexType>
    <xs:sequence>
<xs:element name="transitions" type=TransitionType"
    maxOccurs="unbounded"/>
      <xs:element name="top" type="StateType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Augmenting component behavior in xADL

- ❖ Extend component type:
  - – *ourComponentType*
- ❖ Add attribute *Behavior* of type *StatechartType*

```
<xsd:complexType name="ourComponentType">
  <xsd:complexContent>
    <xsd:extension base="ComponentType">
      <xsd:sequence>
        <xsd:element name="Behavior" type="StatechartType"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

Conclusions

❖ In order to improve system dependability
- – Software Architecture Analysis & Testing

❖ Architectural descriptions should be "open" with respect to analysis and test tools
- – It should be possible to integrate and apply a given analysis or test capability to a given architectural representation with flexibility
- – Architectural descriptions should be able to interact with any analysis technique that works with information in the architecture specifications

❖ This work can be adapted/extended to be useful in the field of architecture-based analysis and testing
- - Reusing Argus-I to do analysis of xADL
- - Integrating xADL XML-schema and extensions with the efforts in architecture-based regression testing and residual testing

Future Work

- ❖ Extending this work to support other formalisms for behavioral specification of software architectures

- ❖ Researching new possibilities in analysis and testing of software systems based on architectural description