On The Evolution of CAEX: A Language Engineering Perspective

Luca Berardinelli\textsuperscript{1}, Rainer Drath\textsuperscript{2}, Emanuel Maetzler\textsuperscript{1}, Manuel Wimmer\textsuperscript{1}

luca.berardinelli@tuwien.ac.at, rainer.drath@de.abb.com, emanuel.maetzler@tuwien.ac.at, manuel.wimmer@tuwien.ac.at

\textsuperscript{1} Business Informatics Group, Institute of Software Technology and Interactive Systems, Vienna University of Technology

\textsuperscript{2} ABB AG, Corporate Research Center Germany
Outline

- AutomationML
- CAEX - Computer Aided EXchange
- CAEX: Language Realization
- CAEX: Language Overview
- Language Evolution
- CAEX Co-Evolution Methodology
- Conclusion
Emerging standard (IEC 62714) for tool data exchange in CPPS engineering

Foundation for harmonizing engineering data coming from a heterogeneous tool network by means of a unified format and data model
- AutomationML is built on top of **CAEX**
- AML docs/models are also CAEX doc/models serialized as **XML based document** compliant with CAEX.xsd
CAEX is a language that is specified in XSD...

```
<xs:schema xmlns:xsi="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
            attributeFormDefault="unqualified">
  <xs:simpleType name="ChangeMode">
    <xs:annotation>
      <xs:documentation>Optionally describes the change state of a CAEX object. If used, the ChangeMode shall have the following value range: state, create, delete and change. This information should be used for further change management applications.</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="state"/>
      <xs:enumeration value="create"/>
      <xs:enumeration value="delete"/>
      <xs:enumeration value="change"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="Header">
    <xs:annotation>
      <xs:documentation>Defines a group of organizational information, like description, version, revision, copyright, etc.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="Description" minOccurs="0">
        <xs:annotation>
          <xs:documentation>Textual description for CAEX objects.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:simpleType>
</xs:schema>
```
CAEX: Language Realization (2/2)

...or in Ecore, enabler for MDE

```
  <modelElements/>
</model>
```

https://eclipse.org/modeling/emf/
CAEX: Language Overview

CAEX in 10 secs slide?

Model (hierarchical graph based structure)

Hierachy

prototypes

roles

interfaces

CAEX in 10 secs slide?

Ecore

CAEX.ecore

«conforms to»

XML

eexample.aml

https://eclipse.org/modeling/emf/
Languages have to evolve!

How to manage co-evolution?
Theory and Practice
Language Evolution: Initial Setting

What is the impact of the evolution dimension on the co-evolution dimension?
Language Evolution: Setting Formalized

\[ \Delta_L := \{ \delta \} \]

\[ \text{«conforms to»} \quad := \mathcal{f}(L,S):\text{Bool} \]

\[ \text{Version } N \quad \text{Ecore} \]

\[ \text{Version } N+1 \quad \text{Ecore} \]

\[ \text{Co-Evolution} \]

\[ \text{Evolution} \]
Language Evolution: Non-Breaking Case

\[ \Delta_L := \{ \delta \} \rightarrow \forall \delta \in \text{NBC} \]

Evolution

Co-Evolution

Version N

Ecore

L

S

\(<\text{conforms to}>\):= f(L,S):\text{Bool}

Version N+1

Ecore

L'

\(<\text{conforms to}>\):

XML

\(<\text{conforms to}>\):= \&\& (L,S):\text{Bool}
Language Evolution: Breaking Case

\[ \Delta_L := \{\delta\} \rightarrow \exists \ \delta \in \text{BC} \]

\[ \text{«conforms to»} \]

\[ L := f(L,S):\text{Bool} \]

\[ \text{«conforms to»} \]

\[ \text{«conforms to»} \]

\[ \text{«conforms to»} \]
Language Evolution: Breaking Case

\[ \Delta_L := \{\delta\} \rightarrow \exists \delta \in BC \]

\[ \Delta_s := \{\delta\} \]

1. \( \Delta_s(S) \in L' \)
2. \( \Delta(I(S),I(S')) \rightarrow \text{min} \)

- **Evolution**
  - Version N
    - Ecore
    - XML
    - «conforms to»
    - \( \Delta_L \)

- **Co-Evolution**
  - Version N+1
    - Ecore
    - XML
    - «conforms to»
    - \( \Delta_s \)

**Formulation**

\[ (L,S):\text{Bool} \]

**Example**

0x505 to 109x540
CAEX: Co-Evolution Methodology

- Represent **languages/models** in a **technology independent** manner
- **Process** consists of **two parts**
  1. Represent **schema delta** and **classify changes**: *non-breaking, breaking*
  2. Develop a **transformation chain** for **data migration**: *copy, check, convert, improve*

---

**Diagram:**
- **CAEX 2.15**
  - «metamodel» Schema
  - «xml» Data
    - «conformsTo»
  - △_Schema
- **CAEX 3.0**
  - «metamodel» Schema'
    - «conformsTo»

**Transformation Chain:**
- **no change or NBC**
  - copy
  - improve
- **BC**
  - check
  - convert

**Impact:**
- no impact
1: SchemaML - A Kernel Schema Language

SchemaML comes with DataML
Changes are derived from the SchemaML specification and classified w.r.t. their impact on existing data

- Set of atomic changes (+, ~, -)
- NBC extend languages
- BC restrict languages or invalidate used identifiers
- Atomic changes may be composed

<table>
<thead>
<tr>
<th>Non-Breaking Changes (NBC)</th>
<th>Breaking Changes (BC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 + Class</td>
<td>1 - Class</td>
</tr>
<tr>
<td>2 + Property [optional]</td>
<td>2 - Property</td>
</tr>
<tr>
<td>3 - Property.upperBound [increase]</td>
<td>3 + Property [mandatory]</td>
</tr>
<tr>
<td>4 ~ Property.lowerBound [decrease]</td>
<td>4 ~ Class.name</td>
</tr>
<tr>
<td>5 - Property.type [extension]</td>
<td>5 ~ Property.name</td>
</tr>
<tr>
<td>6 Property.isContainment [false]</td>
<td>6 ~ Property.lowerBound [increase]</td>
</tr>
<tr>
<td></td>
<td>7 ~ Property.upperBound [decrease]</td>
</tr>
<tr>
<td></td>
<td>8 ~ Property.type [restriction</td>
</tr>
<tr>
<td></td>
<td>9 ~ Property.isContainment [true]</td>
</tr>
</tbody>
</table>
1: Delta Evaluation for CAEX 2.15->3.0

- **Delta model** excerpt for CAEX 2.15 and CAEX 3.0

<table>
<thead>
<tr>
<th>CAEX Evolutions ($\delta_i$) from 2.15 to 3.0</th>
<th>Atomic Changes</th>
<th>Type / Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta_0$ Mandatory RCs for RRIs</td>
<td>AC1</td>
<td>BC6 / Reference RRRefBaseRoleClassPath.lowerBound increased.</td>
</tr>
<tr>
<td>$\delta_1$ Multiple RRIs for IEs</td>
<td>AC2</td>
<td>BC2 / Reference IE.mappingObject to RR removed.</td>
</tr>
<tr>
<td></td>
<td>AC3</td>
<td>NBC2 / Reference RR.mappingObject to M added.</td>
</tr>
<tr>
<td></td>
<td>AC4</td>
<td>NBC3 / Reference RR.mappingObject.upperbound to M increased from optional, single valued (0..1) to optional, multi valued (0..*).</td>
</tr>
<tr>
<td>$\delta_2$ Nested ExtI</td>
<td>AC5</td>
<td>NBC2 / Reference ExtI.externalInterfaces, optional, multi valued (0..*) to ExtI added.</td>
</tr>
<tr>
<td>$\delta_3$ Mandatory ends for ILS</td>
<td>AC6</td>
<td>BC6 / Reference IL.refPartnerSideA.lowerBound increased from optional (0) to mandatory (1).</td>
</tr>
<tr>
<td></td>
<td>AC7</td>
<td>BC6 / Reference IL.refPartnerSideB.lowerBound increased from optional (0) to mandatory (1).</td>
</tr>
<tr>
<td>$\delta_4$ ExtI mappings by identifiers</td>
<td>AC8</td>
<td>BC4 / Class INameM.name renamed to IIIdM.</td>
</tr>
<tr>
<td></td>
<td>AC9</td>
<td>BC5 / Reference INameM.roleInterfaceName renamed to IIIdM.roleInterfaceID.</td>
</tr>
<tr>
<td></td>
<td>AC10</td>
<td>BC5 / Reference INameM.systemUnitInterfaceName renamed to IIIdM.systemUnitInterfaceID.</td>
</tr>
</tbody>
</table>
2: Technologies for Migration Chain

Given the theoretical background...

...we realize a co-evolution framework with MDE technologies.
2: Checking for Missing Data in EVL

check

RR relationship has both ends defined

context caex215!RoleRequirements{
    constraint refBaseRoleClassPathGetsMandatory{
        check { return self.refBaseRoleClassPath <> '' and self.refBaseRoleClassPath <> null; } message { return 'The RoleRequirements of ' + self.eContainer().name + ' needs a valid reference to the RoleClass.';}
    }
}

IE  RR  RC

Resource
2: Copying and Converting Data with ATL

```
rule ConvertRoleRequirements extends CAEXBasicObject {
    from
    src : caex215!RoleRequirements
    to
    trgt : caex300!RoleRequirements ( 
        attribute <- src.attribute,
        externalInterface <- src.externalInterface,
        refBaseRoleClassPath <- src.refBaseRoleClassPath,
        -- moving mapping objects from IE to its inner RR
        mappingObject <- src.refImmediateComposite().mappingObject
    )
}
```
Conclusion

• We presented a **model-driven language engineering perspective** to cope with **CAEX language evolution**

• We presented a **technology independent definition** of the co-evolution problem by devising SchemaML and DataML

• We **classified CAEX evolution** changes in non-breaking and breaking changes as well as how to deal with breaking ones

• We provide a **technical solution** based on **MDE technologies** for our co-evolution methodology: *copy, check, convert, improve*

http://www.sysml4industry.org
DISCLAIMER

These slides are made by authors to help readers quick and better understanding of the subject. The opinions made or information presented here belong to the sole responsibility of the authors. IE Tech News (ITeN) neither endorses nor denounces any opinions made or information presented. Part or full of these material may be used only with written permission from the authors.