Extensible Traceability Management with Eclipse Capra

Jan-Philipp Steghöfer
Chalmers University of Technology | University of Gothenburg
Traceability

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Obstacle Detection</td>
</tr>
</tbody>
</table>

Natural Language System Requirements

Model-based Systems Engineering (MBSysE)

Software Requirements Engineering (SWRE)

Example by David Schmelter, Fraunhofer IEM, Paderborn
Tool Solutions

- Holistic solution: full control over all artifacts and trace models; simplifies maintenance; almost impossible to achieve in practice
- Separate TM tool: more effort in maintenance required, easier to extend, easier to use in existing heterogeneous environments

Salome Maro, Anthony Anjorin, Rebekka Wohlrab, Jan-Philipp Steghöfer: Traceability maintenance: factors and guidelines. ASE 2016: 414-425
Eclipse Capra

Requirements

Variability

Source Code

Tests

Heterogeneous systems

AMALTHEA Contents Tree
Eclipse Capra

- Driven by requirements from industrial partners in a number of research projects
- Only (known, active) universal open source traceability tool
- Goals:
  - Test-bed for new traceability approaches and ideas
  - Enable studies and research on traceability in practical settings
  - Industrial strength: should be usable in production

11 Research Papers
15 Thesis Projects
5 Research Projects
Performing common traceability tasks with Eclipse Capra
Create trace links
Delete trace links
Extending Eclipse Capra
/*
 * Links Safety Case to FMEA
 */
class AddressedByFMEA extends DirectedTraceLink {
    refers Element [1] source // Safety Case
    refers FMEA [1] target // FMEA
}

/*
 * Links Safety Case to Amalthea Element
 */
class ReferencesInEvidences extends DirectedTraceLink {
    refers INamed [1..+] target // Amalthea Element
}

/*
 * Links the Fault Tree Analysis to a Hazard
 */
class DetailsCausesOf extends DirectedTraceLink {
    refers Element [1] source // Element -- Fault Tree Analysis
    refers Element [1..+] target // Element -- Hazard
}

/*
 * Links the Safety Case to the Fault Tree Analysis
 */
class DetailsHazardWith extends DirectedTraceLink {
    refers Element [1] target // Element -- Fault Tree Analysis
}
Handlers for non-EMF artifacts

```java
public class CDTHandler extends AbstractArtifactHandler<ICElement> implements IAnnotateArtifact {

    @Override
    public EObject createWrapper(ICElement element, EObject artifactModel) {
        ICompositeType type = (ICompositeType) element.getParent().getAncestor(ICElement.C_CLASS);
        if (type == null)
            type = (ICompositeType) element.getParent().getAncestor(ICElement.C_STRUCT);
        if (type == null)
            type = (ICompositeType) element.getParent().getAncestor(ICElement.C_UNION);

        String typePrefix = type == null ? "" : type.getName() + "/";
        String uri = new URIBuilder().setScheme("platform").setPath("/resource" + element.getPath())
                        .setFragment(typePrefix + element.getElementName()).toString();

        ArtifactMetaModelIAdapter adapter = ExtensionPointHelper.getArtifactWrapperMetaModelAdapter().orElseThrow();
        return adapter.createArtifact(artifactModel, this.getClass().getName(), uri, element.getHandleIdentifier(),
                                      element.getElementName(), element.getPath().toString());
    }

    @Override
    public ICElement resolveWrapper(EObject wrapper) {
        ArtifactMetaModelIAdapter adapter = ExtensionPointHelper.getArtifactWrapperMetaModelAdapter().orElseThrow();
        return CoreModel.create(adapter.getArtifactIdentifier(wrapper));
    }

    @Override
    public String getDisplayName(ICElement element) {
        return element.getElementName();
    }
}
```
Additional visualisations/analysis

```
visualise(object)

artifactModel = getArtifactWrappers()
    return

wrapper = createWrapper(object)
    return

traceModel = getTraceModel()
    return

traces = getConnectedElements(wrapper, traceModel)
    return

render(traces)
```
Eclipse Capra – Traceability management for the Eclipse platform

- Extensible and flexible – easy to adapt to your own needs
- Completely open source
- Compatible with many common artifacts out of the box
- Driven by research – driving research
- We are looking for additional committers!

- https://eclipse.org/capra