IoNAS: Connecting AUTOSAR and GENIVI Systems Using Model Transformations

Marco Eilers, AUTOSAR Open 2014
Overview

1. Introduction
2. Concept
3. Prototype
4. Availability
5. Future work
History

- AUTOSAR Concept 610: Integration of Non-AUTOSAR Systems (IoNAS)
- GENIVI chosen as first example
- Common AUTOSAR/GENIVI concept group
History

- AUTOSAR Concept 610: Integration of Non-AUTOSAR Systems (IoNAS)
- GENIVI chosen as first example
- Common AUTOSAR/GENIVI concept group
- Initial project commissioned by AUTOSAR:
  - Prototype development: Martin Grosse-Rhode (Fraunhofer FOKUS)
  - Validation: Klaus Birken, Marco Eilers (itemis)
- Second project commissioned by GENIVI
History

- **AUTOSAR Concept 610: Integration of Non-AUTOSAR Systems (IoNAS)**
- GENIVI chosen as first example
- Common AUTOSAR/GENIVI concept group
- Initial project commissioned by AUTOSAR:
  - Prototype development: Martin Grosse-Rhode (Fraunhofer FOKUS)
  - Validation: Klaus Birken, Marco Eilers (itemis)
- Second project commissioned by GENIVI
- Implementation in two phases:
  1. Model transformation
  2. IPC implementation
The Concept Group consolidates GENIVI and AUTOSAR requirements.
GENIVI Alliance

- Alliance of OEMs and manufacturers of IVI products
- Aiming toward a broadly adopted open source development platform

**About the Alliance**

GENIVI® is a non-profit industry alliance committed to driving the broad adoption of an In-Vehicle Infotainment (IVI) open-source development platform.

The alliance aims to align requirements, deliver reference implementations, offer certification programs, and foster a vibrant open-source IVI community.

Our work will result in shortened development cycles, faster time-to-market, and reduced costs for companies developing IVI equipment and software.
Franca IDL

- Interface definition language
Franca IDL

- Interface definition language
  - Methods, attributes, broadcasts
  - Data types
  - Optional deployment models
Franca IDL

- Interface definition language
  - Methods, attributes, broadcasts
  - Data types
  - Optional deployment models
- Generators for different target environments
- Support for constraint checking etc.
Franca IDL

- Interface definition language
  - Methods, attributes, broadcasts
  - Data types
  - Optional deployment models
- Generators for different target environments
- Support for constraint checking etc.

```java
package org.genivi.

interface Player {
  attribute UInt16 currentTrack
  method play {
    in {
      UInt16 trackId
    }
  }
  method nextTrack {
  }
  method previousTrack {
  }
  broadcast endOfPlaylist {
  }
}
```
Model scope

- Different model scopes
  - No components, connectors in Franca

![Diagram showing model scopes in Franca]

- data types
- interfaces
- component instances
- intra-connections

Franca IDL

AUTOSAR SWC Description
Model scope

- Different model scopes
  - No components, connectors in Franca
- Solution: Franca connector DSL
Franca connector DSL

General structure

```franca
*connector1.fcon

connector Connector1 {
    import_franca "francaInput1.fidl";
    import_autosar "autosarInput1.arxml";
    export_franca "result/francaOutput1.fidl";
    export_autosar "output/autosarOutput1.arxml";

    instances {
        franca_instance instance1 implements org.genivi.ionas.testProject.testInterface1;
    }

    connections {
        c1 franca_instance instance1 : testInterface1
            -> autosar_port ARRoot.aComposition : aComponentPrototype:ARRoot.aComponentType.rSR;

        c2 franca_instance instance1
            -> autosar_port ARRoot.aComposition:aComponentPrototype:ARRoot.aComponentType.pCS;

        c3 autosar_port ARRoot.aComposition : aComponentPrototype :ARRoot.aComponentType.rCS
            -> franca_instance instance1 : testInterface1;
    }
}
```
Franca connector

Connection examples

```plaintext
c1 franca_instance instance1 : testInterface1
    -> autosar_port ARRoot.aComposition : aComponentPrototype:ARRoot.aComponentType.rSR;
```
Franca connector

Connection examples

c1 franca_instance instance1 : testInterface1
   -> autosar_port ARRoot.aComposition : aComponentPrototype:ARRoot.aComponentType.rSR;

- testInterface1 has attribute and/or broadcast
- Port rSR is an RPortPrototype with SenderReceiverInterface
- instance1 sends data to aComponentPrototype
Franca connector

Connection examples

testInterface1 has attribute and/or broadcast
Port rSR is an RPortPrototype with SenderReceiverInterface
instance1 sends data to aComponentPrototype

```
c1 franca_instance instance1 : testInterface1
   -> autosar_port ARRoot.aComposition : aComponentPrototype:ARRoot.aComponentType.rSR;
```

```
c3 autosar_port ARRoot.aComposition : aComponentPrototype :ARRoot.aComponentType.rCS ->
   franca_instance instance1 : testInterface1;
```
Franca connector

Connection examples

c1 franca_instance instance1 : testInterface1
   -> autosar_port ARRoot.aComposition : aComponentPrototype:ARRoot.aComponentType.rSR;

- testInterface1 has attribute and/or broadcast
- Port rSR is an RPortPrototype with SenderReceiverInterface
- instance1 sends data to aComponentPrototype

c3 autosar_port ARRoot.aComposition : aComponentPrototype :ARRoot.aComponentType.rCS ->
   franca_instance instance1 : testInterface1;

- testInterface1 has method
- Port rCS is an RPortPrototype with ClientServerInterface
- aComponentPrototype calls method on instance1
Concept mapping

Franca methods, attributes, broadcasts have to be mapped to AUTOSAR interfaces.
Concept mapping

Franca methods, attributes, broadcasts have to be mapped to AUTOSAR interfaces.

- **Methods**
  - normal: ClientServerInterface
  - fireAndForget: SenderReceiverInterface
Concept mapping

Franca methods, attributes, broadcasts have to be mapped to AUTOSAR interfaces.

- **Methods**
  - normal: ClientServerInterface
  - fireAndForget: SenderReceiverInterface

- **Attributes**
  - Getter/Setter: ClientServerInterface
  - Update "listener": SenderReceiverInterface
Franca methods, attributes, broadcasts have to be mapped to AUTOSAR interfaces.

- **Methods**
  - normal: ClientServerInterface
  - fireAndForget: SenderReceiverInterface

- **Attributes**
  - Getter/Setter: ClientServerInterface
  - Update ”listener”: SenderReceiverInterface

- **Broadcasts**: SenderReceiverInterface
Concept mapping

Franca methods, attributes, broadcasts have to be mapped to AUTOSAR interfaces.

- **Methods**
  - normal: ClientServerInterface
  - fireAndForget: SenderReceiverInterface

- **Attributes**
  - Getter/Setter: ClientServerInterface
  - Update "listener": SenderReceiverInterface

- **Broadcasts**: SenderReceiverInterface

Connector DSL includes validation for connections.
Translation process

Franca to AUTOSAR:
1. Translate used data types
2. Create component for every instance
3. Create ports and interfaces
4. Create connector for every connection

AUTOSAR to Franca:
1. Translate used data types
2. Create Franca interface for every used AR interface
Translation process

Franca to AUTOSAR:
1. Translate used data types
2. Create component for every instance
3. Create ports and interfaces
4. Create connector for every connection
Translation process

Franca to AUTOSAR:
1. Translate used data types
2. Create component for every instance
3. Create ports and interfaces
4. Create connector for every connection

AUTOSAR to Franca:
1. Translate used data types
2. Create Franca interface for every used AR interface
Translation result

Franca Model (.fidl)

interface F {
  method m ()
}

AUTOSAR SWC Description (.arxml)

Franca Connector

instances {
  franca_instance f implements F;
  franca_instance g;
}
connections {
  autosar_port AC:a:p -> franca_instance f:F;
  franca_instance g -> autosar_port BC:b:q;
}
Translation result

Franca Model (.fidl)

interface F {
  method m []
}

interface A {
}

interface B {
  method op []
}

AUTOSAR SWC Description (.arxml)

Franca Connector

instances {
  franca_instance f implements F;
  franca_instance g;
}
connections {
  autosar_port AC:a:p -> franca_instance f:F;
  franca_instance g -> autosar_port BC:b:q;
}
Prototype

- Based on Eclipse and Artop
- Two parts: Transformation and UI
Prototype
Prototype

---

```
connector Connector1 {
    import franca "francaInput1.fidl";
    import autosar "autosarInput1.xml";
    export franca "result/francaOutput1.fidl";
    export autosar "output/autosarOutput1.xml";

    instances {
        franca_instance instance1 implements org.genivi.ionas.testProject.testInterface1;
    }

    connections {
        connection1 franca_instance instance1 : testInterface1 -> autosar_port ARRoot.aComp;
        connection2 franca_instance instance1 -> autosar_port ARRoot.aComposition:aComp;
        connection3 autosar_port ARRoot.aComposition : aComponentPrototype : ARRoot.aComp;
    }
}
```

IonAS

Loading Franca deployment model 'platform:/resource/org.genivi.ionas.testProject/francaInput1.fcd'
Loading arxml-file with platform uri 'platform:/resource/org.genivi.ionas.testProject/autosarInput1.arxml'
Saved generated arxml-file as 'platform:/resource/org.genivi.ionas.testProject/output/autosarOutput1.arxml'

AUTOSAR model has been successfully created.

* Transformation summary:
  * Created AUTOSAR base types.
  * Created 0 new AUTOSAR types.
  * Created 1 new SwComponentPrototypes.
  * Created 3 new AUTOSAR interfaces.
  * Created 0 new CompositionSwComponentTypes and 1 new ApplicationSwComponentTypes.
  * Created 3 new AssemblySwConnectors.

---
Availability

- Licensing situation still unclear
- Translation binaries freely available
- Translation source available to AUTOSAR members
  - AUTOSAR SVN
- UI source & binaries freely available

Please get it and try it when available!
(But keep in mind that this is still a prototype.)

http://projects.genivi.org/

Feedback:
marco.eilers@itemis.de
klaus.birken@itemis.de
Availability

- Licensing situation still unclear
- Translation binaries freely available
- Translation source available to AUTOSAR members
  - AUTOSAR SVN
- UI source & binaries freely available

Please get it and try it when available!
(But keep in mind that this is still a prototype.)

http://projects.genivi.org/
Feedback: marco.eilers@itemis.de
klaus.birken@itemis.de
Future work

Actual IPC implementation is the goal of a future project.
Future work

Actual IPC implementation is the goal of a future project.
Thank you for your attention!

Franca connector

Translation scope

- data types
- interfaces
- component instances
- intra-connections
- inter-connections

Translation result

IPC implementation

© itemis AG