



Interoperability with AUTOSAR

Establishing interoperability of GENIVI CommonAPI-based systems and Adaptive AUTOSAR systems by model-to-model transformations

Torsten GÖRG (itemis AG)

GENIVI Technical Summit, 12 November 2019



Agenda

1. Introduction / Motivation
2. Conceptual mapping
3. The FARACON tool
4. Q&A

INTRODUCTION / MOTIVATION

Introduction

SOA (service oriented architecture) is a clear SW design trend in automotive.

SOA in Automotive can be represented in 3 layers

Interface Description Language
(IDL)



Define the contract between services and clients

Middleware



Generates code (from IDL) and provides libraries for SOA services and clients to communicate
Abstracts the underlying transport protocol

Transport protocol



Provide rules to transport the messages
(message identification, serialization, ...)

Problem statement

GENIVI study demonstrated that 2 SOA technologies are under the Automotive spotlights today.

- COMMONAPI + FIDL : strongly adopted in infotainment domain
- ARA::COM : AUTOSAR emerging technology

⇒ **How to ensure compatibility ?**

IDL

Middleware

Transport protocol



FRANCA IDL 

COMMONAPI

SOME/IP



ARXML

ARA::COM

SOME/IP

Solution approach

Two major ideas:

1. Mapping of communication **concepts** of the IDL models

- a. Choose one IDL as reference and translate it
 - i. Either FRANCA IDL (better human readability) or ARXML
 - ii. Goal is the 2 ways translation
- b. IDL brings a set of communication concepts :
 - i. Message types : Event, RPC calls, ...
 - ii. Data types : unitary data (UInt8, Float, ...) and composed data (Struct, ...)

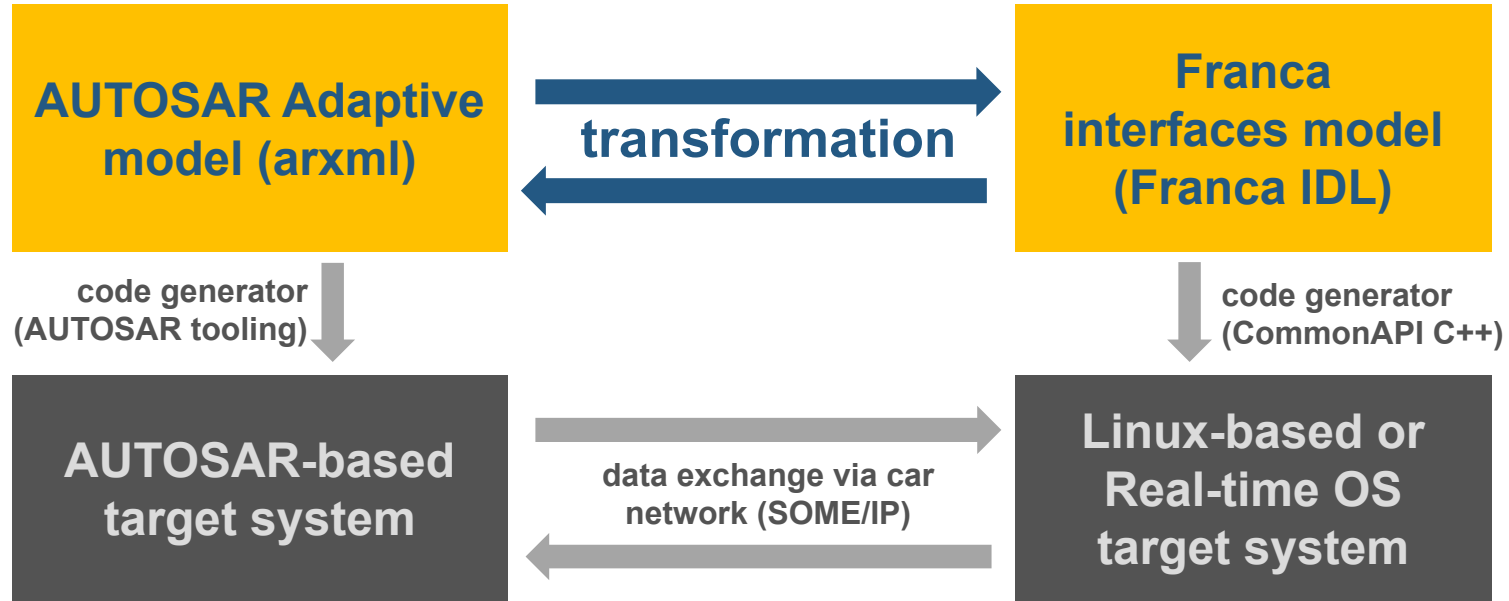
2. Transport the information in the **same format**

- a. Uniquely identify the messages
- b. Serialize the data in the same order and the same encoding
⇒ SOME/IP is the solution to align transport format.



CONCEPTUAL MAPPING

Model transformation tooling



Major requirement: Transform models such that the resulting code on both sides will be compatible wrt. its IPC properties.

Definition of mapping



- AUTOSAR metamodel defined by Artop metamodel (artop.org)
- Franca IDL metamodel defined by Franca Eclipse project
- mapping between both domains is defined
 - each language concept is implemented by a metaclass
 - mapping is defined on metaclasses and their attributes
 - e.g., ServiceInterface \Leftrightarrow FInterface, ClientServerOperation \Leftrightarrow FMethod
- the metamodel level mapping is the starting point for tool implementation
- detailed mapping table (GoogleDocs format)

Detailed mapping table (aka specification)

Definition of Mapping Franca/AUTOSAR

This is the specification for the transformation AUTOSAR Adaptive <=> Franca.
We disregard all concepts of AUTOSAR which purely belong to AUTOSAR Classic

Completed: 80,1%

Group	Franca concept or one of (IGNORE ERROR EMULATE) (see A10)	Franca metamodel export ID	Franca metamodel classifier	Franca metamodel attribute	AUTOSAR concept or one of (IGNORE ERROR EMULATE) (see A10)	Artop metamodel classifier	Artop metamodel attribute	Detail level (IDL, Serialization, CommonAPI, SOMEIP)
structure	version of type collection	IDL1190	Class FTypeCollection	FVersion version (optional)	see IDL1490	see IDL1490	see IDL1490	IDL
structure	list of types (all with visibility public)	IDL1200	Class FTypeCollection	List<FType> types	package contents	ARPackage	List<PackageableElement> getElements()	IDL
structure	list of constants (all with visibility public)	IDL1210	Class FTypeCollection	List<FConstantDefinition> constants	asked MBR...			IDL
structure	interface definition	IDL1220	Class FInterface	n/a	interface definition	ServiceInterface		IDL
structure	list of attributes	IDL1230	Class FInterface	List<FAttribute> attributes	fields of a service	ServiceInterface	List<Field> getFields()	
structure	list of methods	IDL1240	Class FInterface	List<FMethod> methods	client server operations of a service	ServiceInterface	List<ClientServerOperations> on the service interface	IDL
structure	list of broadcasts	IDL1250	Class FInterface	List<FBroadcast> broadcasts	events of a service	ServiceInterface	List<VariableDataPrototype> getEvents()	IDL
structure	optional interface contract	IDL1260	Class FInterface	FContract contract (optional)	IGNORE	n/a	n/a	n/a
structure	inheritance for interfaces	IDL1270	Class FInterface	FInterface base (optional)	EMULATE	n/a	n/a	IDL
structure	manages-relation for interfaces	IDL1280	Class FInterface	List<FInterface> managedInterfaces	ERROR	n/a	n/a	IDL
comm primitives	method	IDL1290	Class FMethod	n/a	operation	ClientServerOperation	n/a	IDL
comm primitives	fire-and-forget flag	IDL1300	Class FMethod	EBoolean fireAndForget (optional)	fire-and-forget flag	ClientServerOperation	boolean isSetFireAndForget()	

- read-only link to document on GoogleDocs: [mapping table](#)

Main concept mappings

- interfaces, type collections
- namespaces, packages
- methods, broadcasts, attributes
- primitive types (booleans, integers, strings, ...)
- struct/union types, array types (implicit/explicit), map types
- annotations/structured comments

See detailed mapping table for a complete list.

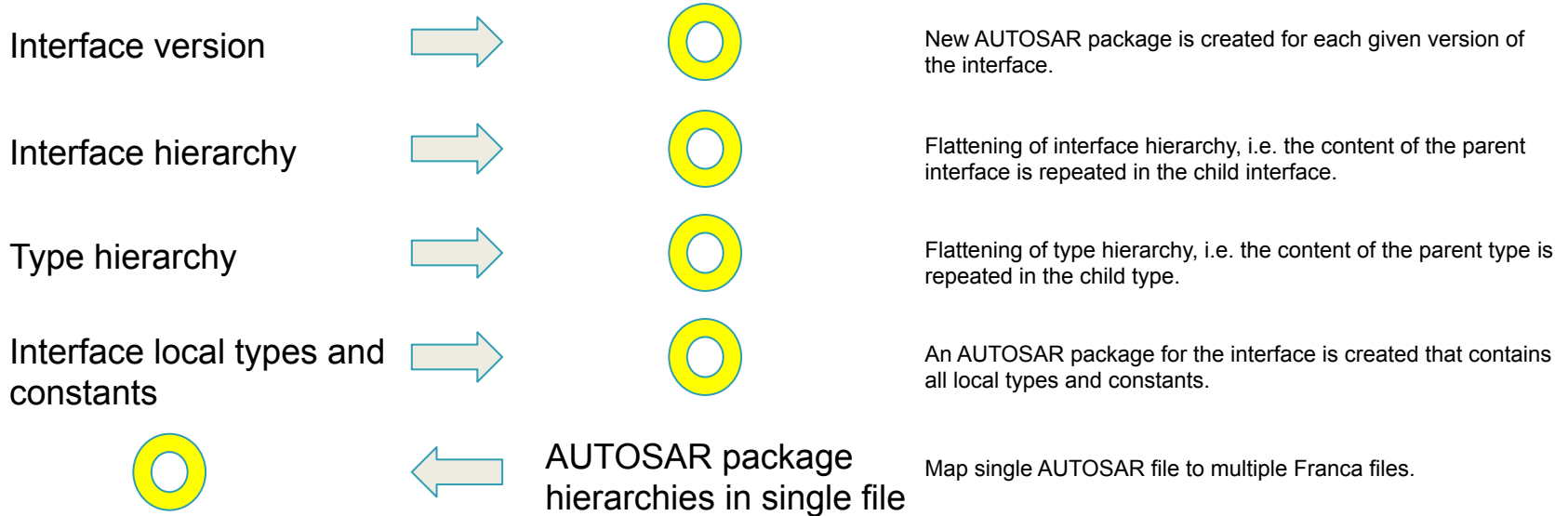
Guidelines for resolving mapping problems

- objective: generated code is compatible
- principal reasons for incompatibilities:
 - a. no corresponding concept on metamodel level (e.g., inheritance)
 - b. generated code shows different behavior (e.g., error handling)
- options for resolving incompatibilities:
 - a. check if concept can be “emulated” (e.g., flattening inheritance)
 - b. check if code generation can be fixed (either by adapting the code generator or indirectly by changing the mapping)
 - c. if all else fails: make user aware that concept cannot be mapped (e.g., by attaching warnings/errors from validation)

Main concept emulation mapping

GENIVI

AUTOSAR



See detailed mapping table for a complete list.

Main concept mapping issues

GENIVI

AUTOSAR

Selective Broadcast



Allows to send broadcast to dedicated clients but against SOA paradigm where only the middleware knows the registered clients.

Polymorphic structures



CommonAPI uses a tagged-value serialization with hash value for the tag. But not defined in SOME/IP specification.



Optional fields

Introduced in AUTOSAR Adaptive 18.10. Field is present if tag is present in TLV serialization format. CommonAPI serializes structs with LV (length presence and width is configurable in FDEPL).



Data semantic

Used to define the content of an unitary data (unit, max value...) Not defined in FIDL.

Method errors



Method errors

Application errors are not transported on the same way. AUTOSAR uses SOME/IP error code to transport applicative errors. CommonAPI generated code defines a mandatory error status.

See detailed mapping table for a complete list.

THE FARACON TOOL

Technology Demonstrators in 2019/Q1



CES, Las Vegas, January 2019



European R-CAR Consortium Forum, Düsseldorf, March 2019

Development cooperation

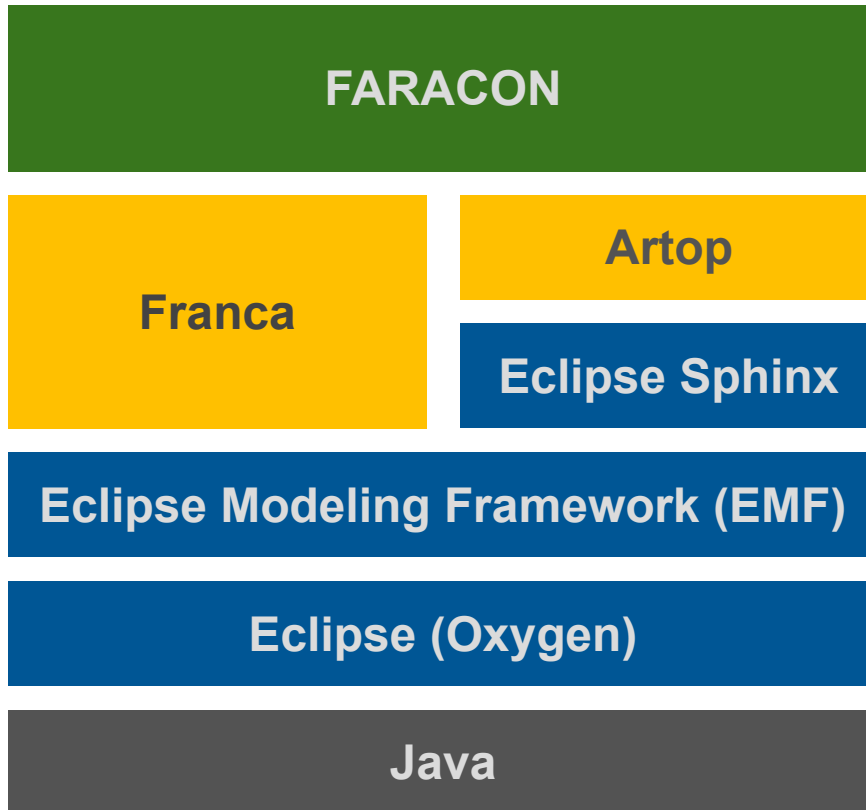
- cooperation partners:
 - Renault
 - Continental
 - Visteon
- ensure to meet the really existing needs
- feedback from testing with real world system

The FARACON tool

- transform Franca IDL models to Artop models and vice versa
- scope for Artop models is limited to AUTOSAR adaptive
- supported versions
 - Franca 0.13.1
 - Artop 4.12 (AUTOSAR Adaptive Platform R19-03)
- no full roundtrip (information loss due to mapping limitations)
- planned: mapping of deployment data for SOME/IP



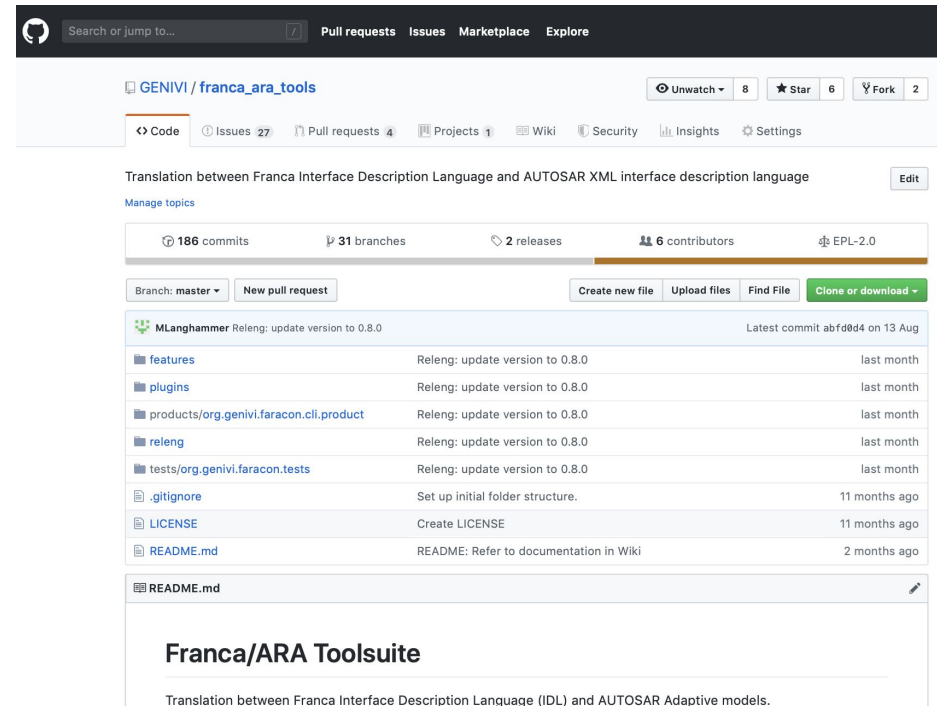
FARACON tool architecture



- based on the Eclipse ecosystem
- usable both from Eclipse IDE and command-line
- EMF as a common metamodel
- transformations implemented using Xtend language
- additional tools/frameworks: maven/Tycho, JUnit
- **Artop is available only to AUTOSAR members**

FARACON licensing and development

- FARACON is open-source (Eclipse Public License v2)
- FARACON development is funded by GENIVI Alliance
- public repository:
https://github.com/GENIVI/franca_ara_tools
- Artop plugins have to be downloaded from artop.org, AUTOSAR membership needed!



Search or jump to... Pull requests Issues Marketplace Explore

GENIVI / franca_ara_tools Unwatch 8 Star 6 Fork 2

Code Issues 27 Pull requests 4 Projects 1 Wiki Security Insights Settings

Translation between Franca Interface Description Language and AUTOSAR XML interface description language Edit

Manage topics

186 commits 31 branches 2 releases 6 contributors EPL-2.0

Branch: master New pull request Create new file Upload files Find File Clone or download

MLanghammer Releng: update version to 0.8.0	Latest commit abfd0d4 on 13 Aug
features	Releng: update version to 0.8.0 last month
plugins	Releng: update version to 0.8.0 last month
products/org.genivi.faracon.cli.product	Releng: update version to 0.8.0 last month
releng	Releng: update version to 0.8.0 last month
tests/org.genivi.faracon.tests	Releng: update version to 0.8.0 last month
.gitignore	Set up initial folder structure. 11 months ago
LICENSE	Create LICENSE 11 months ago
README.md	README: Refer to documentation in Wiki 2 months ago

README.md

Franca/ARA Toolsuite

Translation between Franca Interface Description Language (IDL) and AUTOSAR Adaptive models.

FARACON Release 0.9

- already available
- features:
 - nearly 100% of transformation logic (both directions)
 - IDE integration and command-line tool
- see [mapping table](#) on GoogleDocs for full details

FARACON Release 1.0

- available mid of November 2019
- additional features:
 - prototypical mapping for SOME/IP deployment data (i.e., to support fixed-sized arrays)
 - configurable/customizable AUTOSAR primitive types
- plus bugfixes from beta testing feedback

Q&A

