ARCA - Automated Analysis of AUTOSAR Meta-Model Changes

Darko Durisic
Dept. of Electrical Systems Design
Volvo Car Corporation
Gothenburg, Sweden
Darko.Durisic@volvocars.com

Miroslaw Staron
Software Engineering Division
Chalmers | University of Gothenburg
Gothenburg, Sweden
Miroslaw.Staron@cse.gu.se

Matthias Tichy
Software Engineering Division
Chalmers | University of Gothenburg
Gothenburg, Sweden
Matthias.Tichy@cse.gu.se

***Abstract*—The software architecture of automotive software systems on the European market and wider is designed following the AUTOSAR standard. This requires continuous adoption of new AUTOSAR releases in the development projects in order to enable new innovative solutions in cars. Under these circumstances, the analysis of impact of the AUTOSAR meta-model changes on the modeling tools used in the development is crucial for avoiding delays and increased cost. However due to tens of new features combined with thousands of meta-model changes between consecutive releases of AUTOSAR, tool support is needed for such analysis. In this paper we present a systematic method and a tool - ARCA - for automated analysis of the AUTOSAR meta-model changes. The tool is able to identify relevant changes affecting modeling tools used by different roles in the development process and present the optimal set of new features to be adopted in the projects. The goal of the tool is to enable faster and cheaper software innovation cycles in cars.**

I. Introduction

The development of automotive software systems and their architectures on the European market and wider is mostly based on the AUTOSAR [1] (AUTomotive Open System ARchitecture) standard [2]. One reason is the possibility to re-use the existing architectural components and their implementations (e.g. related to middleware and hardware [3]) but also to more easily exchange the architectural models between the tools of different software vendors. In order to facilitate the exchange of these models, AUTOSAR defines a meta-model and requires full compliance of the system models to the AUTOSAR meta-model. We consider a model as an abstract representation of a software system and a meta-model as a model which defines the syntax and the semantics of a particular domain-specific modeling environment [4], [5].

Development based on the standardized meta-model requires constant adoption of new meta-model releases in the development in order to enable new innovative solutions in car projects. A good example of such a solution is *Ethernet* as a communication medium between different Electronic Control Units (ECUs) responsible for one or more vehicle functions (e.g. engine control) in a distributed automotive system. However as new releases of AUTOSAR usually bring thousands of changes to the AUTOSAR meta-model (e.g. more than 33 000 changes between two consecutive releases *4.1.3* and *4.2.1* as shown later), careful analysis of the impact of these changes on different modeling tools supporting these solutions is required before their implementation [6].

In particular, automotive software designers are often confronted with decisions about which newer AUTOSAR release or subsets of new AUTOSAR features to adopt in the development projects. They also want to know which roles in the development process will be mostly affected by the changes. Due to the constant increase in the size and complexity of the AUTOSAR meta-model related to new features in cars [7] (e.g. *R4.2.1* is 4-5 times more complex than *R3.2.3* for different roles as shown later), tool support is needed to quickly identify relevant changes for the most critical roles and facilitate the cost-benefit analysis of adopting different features.

In this paper we present a systematic method and a tool - *ARCA* (AutosaR Change Analyzer) - for automated analysis of AUTOSAR meta-model changes affecting modeling tools used by a set of defined roles. The tool has 3 main functionalities:

1) Quantifying and presenting the changes between different versions of the AUTOSAR meta-model.
2) Presenting the results of a number of software metrics characterizing the AUTOSAR meta-model evolution.
3) Quantifying and presenting the changes caused by specific features of a new AUTOSAR release.

Based on the first two functionalities, a decision about the adoption of a specific new AUTOSAR release can be made. This includes the identification of the most critical roles affected by the changes. We studied in [6] the evolution of the AUTOSAR meta-model for different roles and assessed the applicability of the used metrics for monitoring its evolution in [8]. In this paper we focus on the utilization of the results from these studies in car projects. Using the third functionality, a decision about the adoption of the optimal number of features from a new AUTOSAR release can be made based on their prioritization and estimated impact on the modeling tools.

This paper is structured as follows: Section 2 describes the AUTOSAR based development of automotive software systems. Section 3 describes the related work. Section 4 describes the *ARCA* tool. Finally section 5 presents our conclusions.

II. AUTOSAR Based Software Development

The development of the automotive software systems is distributed as they are developed in a collaborative environment which involves a number of actors. On one side we have car manufacturers (OEMs - Original Equipment Manufacturers) responsible for designing and verifying the architecture of the system. On the other side we have different layers of suppliers (e.g. application software suppliers, tool suppliers, hardware suppliers) responsible for design, implementation and verification of the specific architectural components [9]. As each party in the development process may use their own tools for working with the architectural models, the exchange of these models between different actors is quite challenging.

In order to facilitate this distributed development, the AUTOSAR standard was introduced [10] as a partnership of OEMs and their suppliers. One of the main goals of AUTOSAR is to standardize the exchange format for the architectural models of the system. This is done by defining a meta-model which specifies the syntax and the semantics of the automotive modeling environment [4], [5] and serves as a basis for the development of the modeling tools.

Figure 1 shows a simplified example of the usage of the AUTOSAR meta-model to allocate software components onto ECUs. The meta-model to the left defines how to map software components to ECUs while the model to the right instantiates this meta-model by mapping the actual *EnginePowerUnit* software components to *EngineControlModule* ECU.

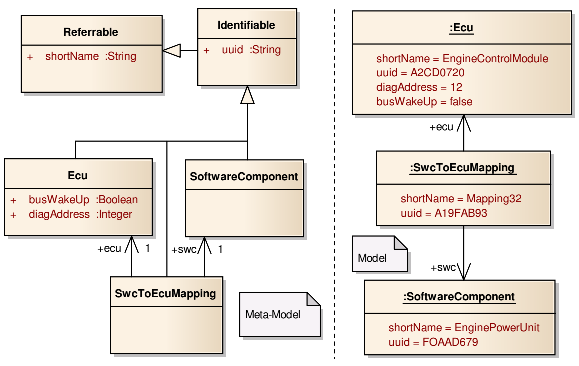


Fig. 1. AUTOSAR Meta-Model example

The development of the AUTOSAR meta-model is done with the help of two tools - a change management tool *Bugzilla* and an *SVN* repository. For implementing changes related to the new features, the process depicted in figure 2 is followed.

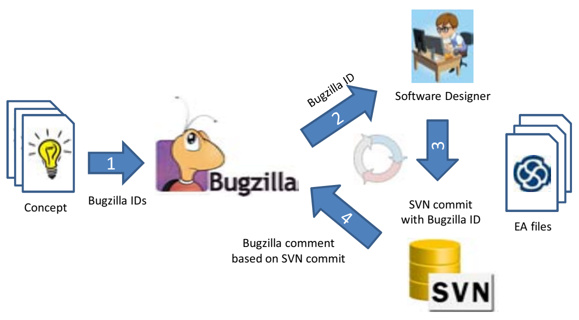


Fig. 2. Linking meta-model changes to AUTOSAR features

For each new feature to be implemented in the standard, an entry in *Bugzilla* is created with a unique identifier (1) which contains the description of the feature and the agreed solution. Software designers implementing the changes in the AUTOSAR meta-model use this identifier (2) in the commit message when committing the new version of the meta-model to the *SVN* (3). Several *SVN* commits of the meta-model may be related to one feature. Every time a commit is made, a comment is added to the *Bugzilla* entry with the identifier from the commit message (4). To assure that no links are omitted by the change implementers, the *SVN* repository shall be configured to accept only certain structure of commit messages, e.g. a regular expression starting with the unique identifier of the *Bugzilla* entry (e.g. #12345).

III. Related Work

A number of studies focus on the coupled evolution of models and meta-models and assessing the impact of meta-model changes on different artifacts. For example Ruscio et al. [11] address the impact of meta-model evolution on the entire meta-modeling eco-system, e.g. models, transformations and modeling tools and Mendez et al. show how to perform the automated transformation of models according to the meta-model changes [12]. Our paper contributes to these studies by analyzing the impact of meta-model changes on the modeling tools with the focus on supporting new features.

A number of software tools exist today for supporting the analysis of repository changes for different software artifacts such as VCS-Analyzer presented by Fontana et al. [13] or the change management tool presented by Li et al. [14]. However most of these tools are not tailored to the analysis of meta-model evolution and they are not capable of linking meta-model changes to different system features.

Additionally Poncin et al. present FLASR - a framework for analyzing software repositories by combining different repositories and matching related software development events [15]. We utilize a part of this framework (*SVN-Bugzilla* links) for linking AUTOSAR meta-model changes to features.

IV. *ARCA* Tool

The description of the *ARCA* tool is organized in 5 subsections. The first one contains a description of the architecture of the tool while the latter 3 contain the definition of the main functionalities supported by the tool. The last section shows an example of how to combine all functionalities in car projects.

A. *Architecture of the ARCA Tool*

For the analysis of changes between different versions of the AUTOSAR meta-model, the *ARCA* tool uses the meta-data model presented in figure 3 which represents a simplified version of the MOF meta-model [16].

We define a change as an atomic modification, addition or removal of the meta-data model elements and their properties (e.g. *Name*, *Note*). For example if an *Attribute* changed both its *Name* and *Type*, this represents two changes. Additionally when introducing or removing meta-data elements (e.g.

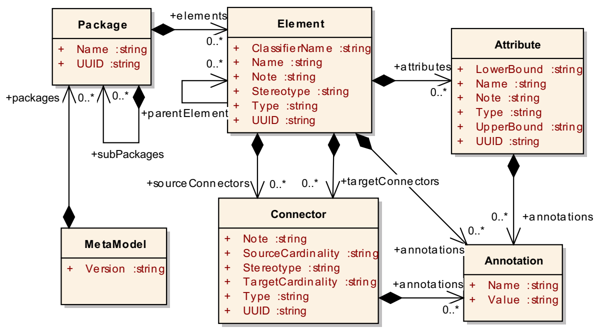


Fig. 3. Meta-data model

Attributes) containing other meta-data elements (e.g. *Annotations*), both changes to the containing and contained meta-data elements are considered. The comparison is done based on the unique element identifiers in the *Enterprise Architect* model.

Data-models instantiating this meta-data model (*.mod* files) are obtained by extracting the data from the *Enterprise Architect* (*.eap*) files corresponding to a specific AUTOSAR meta-model version (see 1 in figure 4). Two data-models need to be loaded into the tool (see 2 in figure 4) before further analysis of the changes is possible. The reason why the tool works only with its own data-models is to increase the performance, i.e. it is more than 100 times faster to load the extracted data-model in comparison to querying the *Enterprise Architect* file.

Before using the *ARCA* tool for different types of change analysis, two important aspects need to be configured: (i) which types of changes shall be considered and (ii) which roles shall be used in the analysis. The first aspect is important as certain changes may not require any implementation effort in the AUTOSAR meta-model based tools, e.g. changes in the format of the unique identifiers of the meta-elements or in their notes. The second aspect is important as the development of automotive software systems involves a number of actors so the role based analysis of the changes may indicate which teams (i.e. their modeling tools) will be mostly affected.

The configuration of these two aspects is done by importing the configuration file (*.xml*) before the analysis (see 5 in figure 4). Regarding the considered types of changes (referred to as relevant changes), it is possible to specify which meta-model packages shall be excluded from the analysis. It is also possible to specify whether the changes in the element notes shall be considered and which annotations shall be excluded. Consideration of the relevant changes only can be enabled and disabled via the 'Relevant only' check-box (see 3 in figure 4).

As the AUTOSAR meta-model is organized in logical packages, roles in the configuration file are defined as collections of packages affecting modeling tools used by the corresponding role. We defined in [6] 7 major roles in the automotive software development process but new roles can be defined.

The link between meta-model changes and AUTOSAR features is established by analyzing the AUTOSAR meta-model changes between the *SVN* commits referring to the *Bugzilla* entry which corresponds to the analyzed feature.

B. Quantifying and Presenting Changes Between Versions

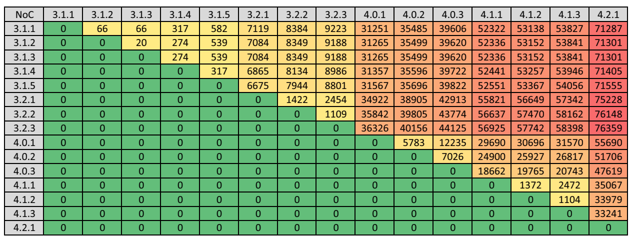
This functionality enables a comparison between two AUTOSAR meta-model versions (e.g. two different releases of the AUTOSAR meta-model) for a selected role (see 7 in figure 4) after the corresponding *.mod* files have been loaded. By selecting the 'Metrics' check-box in the 'Show changes' panel (see 9 in figure 4), we can see the total number of changes between the chosen meta-model versions together with the number of modified, added and removed elements, attributes and packages (see 11 in figure 4). In addition to the quantitative analysis of the changes, it is also possible to list all changes and modified, added and removed elements, attributes and packages by selecting the corresponding check-boxes in the 'Show changes' panel (see 12 in figure 4).

There are two main use-cases for this functionality:

1) Analyzing the impact of switching from one AUTOSAR release to another on the tools used by different roles in the development process.
2) Constant follow-up of the changes in the AUTOSAR meta-model between two releases in order to influence their standardization (e.g. prevent the removal of an element which is planned to be used in future).

Apart from the analysis of the changes between two AUTOSAR meta-model versions, the *ARCA* tool can be used for generating *.csv* reports from the comparison of multiple meta-model versions specified in the configuration file (i.e. paths to the corresponding *.mod* files are provided) for all defined roles (see 'Report changes' under 6 in figure 4). Based on these reports, heat-maps showing the number of changes (or the number of modified, added and removed elements, attributes and packages) which need to be implemented in the AUTOSAR based tools to switch from one AUTOSAR release to another can be created for each role.

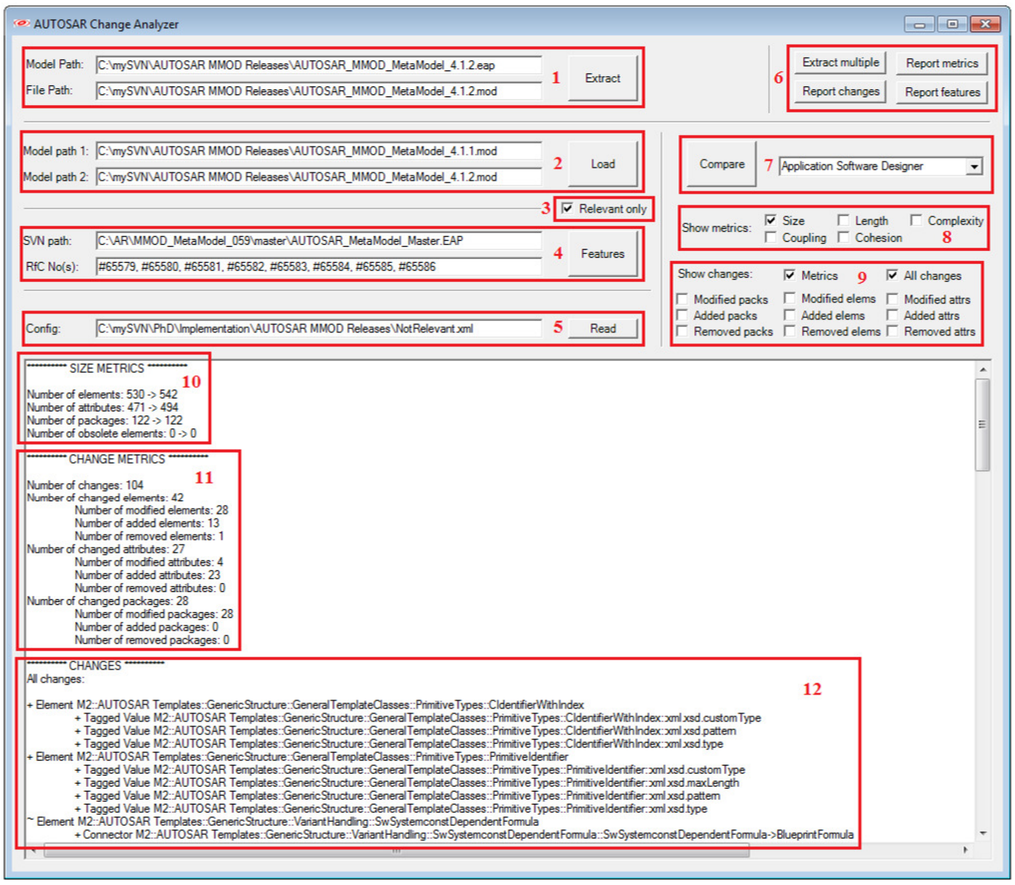
The main use-case for the heat-maps is during the decision making process of which new AUTOSAR release to adopt in the development process. This is because they can serve as a good indicator of potential cost needed to update the AUTOSAR based modeling tools used by different roles. An example of such a heat-map containing the number of relevant changes for the AUTOSAR releases *3.0.1 - 4.2.1* for the entire meta-model is shown in figure 5.



NoC	3.1.1	3.1.2	3.1.3	3.1.4	3.1.5	3.2.1	3.2.2	3.2.3	4.0.1	4.0.2	4.0.3	4.1.1	4.1.2	4.1.3	4.2.1
3.1.1	0	66	66	317	582	7119	8384	9223	31251	35485	39606	52322	53138	53827	71287
3.1.2	0	0	20	274	539	7084	8349	9188	31265	35499	39620	52336	53152	53841	71301
3.1.3	0	0	0	274	539	7084	8349	9188	31265	35499	39620	52336	53152	53841	71301
3.1.4	0	0	0	0	317	6865	8134	8986	31357	35596	39722	52441	53257	53946	71405
3.1.5	0	0	0	0	0	6675	7944	8801	31567	35696	39822	52551	53367	54056	71555
3.2.1	0	0	0	0	0	0	1422	2454	34922	38905	42913	55821	56649	57342	75228
3.2.2	0	0	0	0	0	0	0	1109	35842	39805	43774	56637	57470	58162	76148
3.2.3	0	0	0	0	0	0	0	0	36326	40156	44125	56925	57742	58398	76359
4.0.1	0	0	0	0	0	0	0	0	0	5783	12235	29690	30696	31570	55690
4.0.2	0	0	0	0	0	0	0	0	0	0	7026	24900	25927	26817	51706
4.0.3	0	0	0	0	0	0	0	0	0	0	0	18662	19765	20743	47619
4.1.1	0	0	0	0	0	0	0	0	0	0	0	0	1372	2472	35067
4.1.2	0	0	0	0	0	0	0	0	0	0	0	0	0	1104	33979
4.1.3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	33241
4.2.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 5. Heatmap example

We can see that most changes are needed when switching from an AUTOSAR release *3.x* to a release *4.x* as these branches have been developed in parallel for some time leading to their divergence.



Model Path: C:\mySVN\AUTOSAR MMOD Releases\AUTOSAR_MMOD_MetaModel_4.1.2.eap
File Path: C:\mySVN\AUTOSAR MMOD Releases\AUTOSAR_MMOD_MetaModel_4.1.2.mod

Model path 1: C:\mySVN\AUTOSAR MMOD Releases\AUTOSAR_MMOD_MetaModel_4.1.1.mod
Model path 2: C:\mySVN\AUTOSAR MMOD Releases\AUTOSAR_MMOD_MetaModel_4.1.2.mod

[✓ Relevant only]

SVN path: C:\AR\MMOD_MetaModel_059\master\AUTOSAR_MetaModel_Master.EAP
RfC No(s): #65579, #65580, #65581, #65582, #65583, #65584, #65585, #65586

Config: C:\mySVN\PhD\Implementation\AUTOSAR MMOD Releases\NotRelevant.xml

Application Software Designer

Show metrics: [✓ Size] Length Complexity Coupling Cohesion

Show changes: [✓ Metrics] [✓ All changes] Modified packs Added packs Removed packs Modified elems Added elems Removed elems Modified attrs Added attrs Removed attrs

```
*********** SIZE METRICS ***********

Number of elements: 530 -> 542
Number of attributes: 471 -> 494
Number of packages: 122 -> 122
Number of obsolete elements: 0 -> 0
*********** CHANGE METRICS ***********

Number of changes: 104
Number of changed elements: 42
            Number of modified elements: 28
            Number of added elements: 13
            Number of removed elements: 1
Number of changed attributes: 27
            Number of modified attributes: 4
            Number of added attributes: 23
            Number of removed attributes: 0
Number of changed packages: 28
            Number of modified packages: 28
            Number of added packages: 0
            Number of removed packages: 0
*********** CHANGES ***********
All changes:

+ Element M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::CIdentifierWithIndex
            + Tagged Value M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::CIdentifierWithIndex::xml.xsd.customType
            + Tagged Value M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::CIdentifierWithIndex::xml.xsd.pattern
            + Tagged Value M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::CIdentifierWithIndex::xml.xsd.type
+ Element M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::PrimitiveIdentifier
            + Tagged Value M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::PrimitiveIdentifier::xml.xsd.customType
            + Tagged Value M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::PrimitiveIdentifier::xml.xsd.maxLength
            + Tagged Value M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::PrimitiveIdentifier::xml.xsd.pattern
            + Tagged Value M2::AUTOSAR Templates::GenericStructure::GeneralTemplateClasses::PrimitiveTypes::PrimitiveIdentifier::xml.xsd.type
~ Element M2::AUTOSAR Templates::GenericStructure::VariantHandling::SwSystemconstDependentFormula
            + Connector M2::AUTOSAR Templates::GenericStructure::VariantHandling::SwSystemconstDependentFormula::SwSystemconstDependentFormula->BlueprintFormula
```

Fig. 4. ARCA tool

C. *Presenting Results of Software Metrics*

This functionality enables a comparison between results of a number of software metrics applied on the chosen two AUTOSAR meta-model versions for a selected role (see 7 in figure 4) after the corresponding *.mod* files have been loaded. However this time the presentation of the results (see 10 in figure 4) depends on the selected check-boxes of the 'Show metrics' panel (see 8 in figure 4). The metrics are divided into 5 categories according to the properties defined by Briand et al. [17] and include the following metrics:

- **Size:** Number of elements, Number of attributes
- **Length:** Dept of inheritance
- **Complex:** Fan-in, Fan-out and Fan-IO (Fan-in + Fan-out)
- **Coupling:** Coupling between obj. and Package coupling
- **Cohesion:** Package cohesion and Cohesion ration

While the size and length metrics are based on the commonly used UML metrics [18], the complexity, coupling and cohesion metrics are based on the interaction between different elements (meta-classes) in the meta-model, i.e. based on *Connector*s (associations). Even though in the modeling world associations can be considered as attributes of the classes, in case of industrial meta-models they may have slightly different semantic as meta-classes there represent logical entities whose instances may be modeled by separate teams. Therefore the introduction / removal of one association may have a wider impact than the introduction / removal of one attribute which describes only one meta-class. For this reason, we analyzed the connectors in the context of complexity, coupling and cohesion rather than in the context of size. A detailed definition of all presented metrics can be found in [8].

The main use cases for this functionality is to analyze the impact of the AUTOSAR meta-model changes on different roles and their interaction (model exchange). Therefore the distinction between the results of coupling and cohesion metrics is especially important as high cohesion increase of one role indicates high re-work of the modeling tools used by this role while high coupling increase indicates possible interoperability issues between the tools used by different roles.

Apart from the comparison of the results of metrics between two AUTOSAR meta-model versions, the *ARCA* tool can be used for generating *.csv* reports from the calculation of metrics on multiple meta-model versions specified in the configuration file (i.e. paths to the corresponding *.mod* files are provided) for all defined roles (see 'Report metrics' under 6 in figure 4). Based on these reports, histograms and trend charts showing the evolution of the AUTOSAR meta-model with respect to the analyzed properties can be created for each role.

The main use-case for these charts is during planning activities in order to identify places in the development (i.e. exchange of models between two roles) where additional resources, money or integration activities should take place. An example of such a chart showing the complexity evolution of the AUTOSAR meta-model between releases *3.0.1 - 4.2.1* for 4 roles defined in the configuration file (see the description of roles and their mapping to different packages in [6]) based on the results of the *Fan-IO* metric is shown in figure 6.

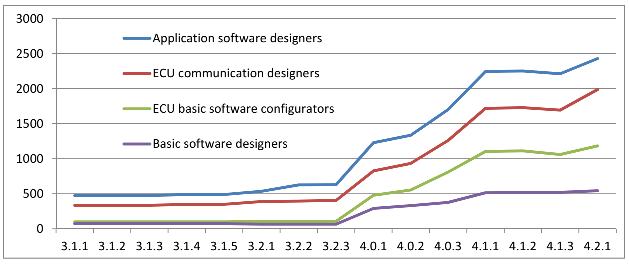


Fig. 6. Complexity evolution

We can see a big increase in the complexity between AUTOSAR releases *4.0.1* and *4.1.1* for all roles indicating higher risk of faults. We can also see that the role of the *Application software designers* is considered as the most complex.

D. Presenting and Quantifying Feature Related Changes

This functionality is similar to the first functionality, however this time it is possible to present the results of changes for a selected role related to specific feature. This is done utilizing the process presented in figure 2. Based on the provided *SVN* path to the AUTOSAR meta-model and a list of *Bugzilla* entries relevant for the analyzed feature (see 4 in figure 4), the tool will analyze the changes between meta-model commits where the commit message references at least one *Bugzilla* entry from the list. In case the message references *Bugzilla* entries related to more than one feature, the user is asked to discard the changes not relevant for the analyzed feature.

The main use-case for this functionality is the impact assessment of adopting only certain features from new AUTOSAR releases on the modeling tools used in the development. This is possible as AUTOSAR features are usually loosely coupled and the implementation of changes is always done in a backwards compatible way so it is not very likely that changes caused by one feature will affect other existing and/or new features. This functionality is important as it is often the case that only certain features from new AUTOSAR releases are actually needed so there is no need for adopting an entire new release with thousands of changes to the meta-model.

Apart from the analysis of changes related to a specific AUTOSAR feature, the *ARCA* tool can be used for generating *.csv* reports from the analysis of multiple features specified in the configuration file (i.e. relevant *Bugzilla* entries for each feature are provided) for all defined roles (see 'Report features' under 6 in figure 4). Based on these reports, histograms showing the number of changes (or the number of modified, added and removed elements, attributes and packages) needed to be implemented in the AUTOSAR based tools to adopt each feature can be created for each role.

The example of such a histogram containing the number of relevant changes for 14 new features (referred to as concepts according to the AUTOSAR terminology) of the AUTOSAR release *4.2.1* is shown in figure 7.

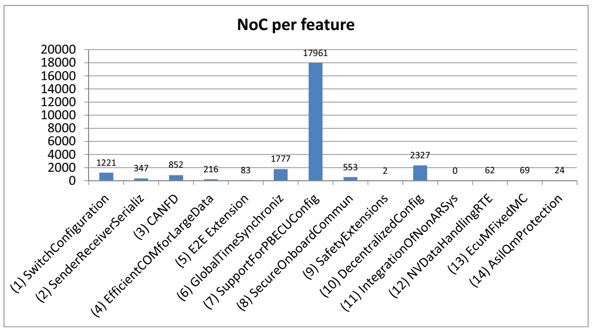


Fig. 7. Number of changes per concept

We can see that feature 7 (SupportForPBECUConfig) requires significantly more changes than all other features combined, i.e. it represents an outlier. Without this tool, the amount of work needed to adopt this feature in the development projects could be underestimated.

Together with the generated *.csv* reports from the analysis of multiple features, *ARCA* tool can used to identify and present the optimal sets of new features (solutions) to be adopted in the development projects based on two objectives [19]: (i) to maximize the weighted number of features to be adopted and (ii) to minimize the number of changes to the AUTOSAR meta-model caused by the features. The weight of each feature can be defined in the configuration file.

The optimal solutions are presented using the Pareto optimality chart which is created automatically by running the generated R script using the R tool. The script also shows which features are included in which solution. An example of the Pareto chart with 14 optimal solutions containing features of equal weight from the AUTOSAR release *4.2.1* is shown in figure 8 and the mapping of features to solutions in table I.

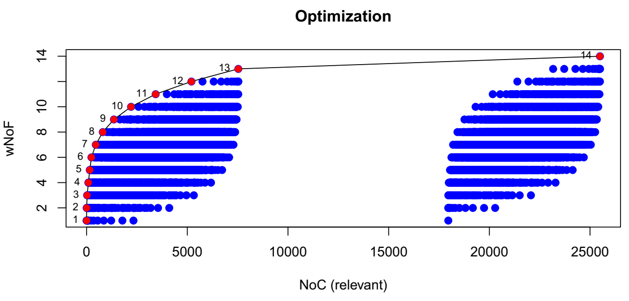


Fig. 8. Pareto front

We can see that with a relatively small increase in the number of changes, car manufacturers can adopt several additional

TABLE I
MAPPING OF FEATURES TO FEATURE SETS

Feature set	Features
1	11
2	9, 11
3	9, 11, 14
4	9, 11, 12, 14
5	9, 11, 12, 13, 14
6	5, 9, 11, 12, 13, 14
7	4, 5, 9, 11, 12, 13, 14
8	2, 4, 5, 9, 11, 12, 13, 14
9	2, 4, 5, 8, 9, 11, 12, 13, 14
10	2, 3, 4, 5, 8, 9, 11, 12, 13, 14
11	1, 2, 3, 4, 5, 8, 9, 11, 12, 13, 14
12	1, 2, 3, 4, 5, 6, 8, 9, 11, 12, 13, 14
13	1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14
14	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14

features (e.g. features 5, 9, 12, 13 and 14 from solution 6) in comparison to only one feature 11 from solution 1. However features 6, 10 and especially 7 from solutions 12, 13 and 14 respectively require significant re-work in the modeling tools and they should be considered only in case of high demand.

E. Combining All Functionalities of the Tool in Car Projects

There are still car manufacturers today developing automotive software systems based on an AUTOSAR release *3.x*. In order to decide which *4.x* release to adopt in the development, the results presented in figures 5 and 6 can be used. For example if *R4.1.1* contains the required features, we can see that the increase in the number of changes and complexity between this release and *R4.1.3* is not that high. Therefore it would make sense to adopt the newer *R4.1.3* as some of identified faults in previous releases have been fixed.

Due to its high increase in the number of changes and complexity, there is a risk of high number of faults in *R4.2.1* so it may be wise to wait for future *4.2.x* releases of stable complexity where most of them are fixed. However if there are some required features from this release, Pareto optimality chart can be used to identify which *R4.2.1* features can be adopted on top of *R4.1.3* without a high risk of late faults and increased cost. These kinds of analysis would not be feasible for the automotive software designers without the tool support.

V. CONCLUSION

Today automotive software designers need to know AUTOSAR in order to keep track of the change in its meta-model and analyze their impact on different modeling tools used in the development. With the *ARCA* tool, they can focus on the implementation of car functions and get a quick feedback about the cost of switching to a newer AUTOSAR release. Additionally linking meta-model changes to different features enables a quick overview of the optimal sets of AUTOSAR features to be adopted in the development projects.

As the trend in the size and complexity increase of the AUTOSAR meta-model is expected to continue in future, tools like this become even more important for the car manufacturers in order to assure high quality of the automotive electrical systems and lower their development cost.

The *ARCA* tool can be downloaded from the following link: *http://web.student.chalmers.se/~durisic/ARCA.zip*

ACKNOWLEDGMENT

The authors would like to thank Swedish Governmental Agency for Innovation Systems (VINNOVA) for funding this research (grant no. 2013-02630) and the AUTOSAR team at Volvo Car Corporation for contributing to the work.

REFERENCES

[1] *Automotive Open System Architecture*, www.autosar.org, 2003.

[2] M. Di Natale and A. L. Sangiovanni-Vincentelli, "Moving From Federated to Integrated Architectures in Automotive: The Role of Standards, Methods and Tools," *Proceedings of the IEEE*, vol. 98, no. 4, pp. 603–620, 2010.

[3] S. Gal-Oz, "Standard API Would Significantly Accelerate Embedded System Development," *Real-Time Magazine*, vol. 5, pp. 81–87, 1999.

[4] T. Kűhne, "Matters of (Meta-) Modeling," *Journal of Software and Systems Modeling*, vol. 5, no. 4, pp. 369–385, 2006.

[5] G. Nordstrom, B. Dawant, D. M. Wilkes, and G. Karsai, "Metamodeling - Rapid Design and Evolution of Domain-Specific Modeling Environments," in *Proceedings of the IEEE Conference on Engineering of Computer Based Systems*, 1999, pp. 68–74.

[6] D. Durisic, M. Staron, M. Tichy, and J. Hansson, "Evolution of Long-Term Industrial Meta-Models - A Case Study of AUTOSAR," in *Proceedings of the Euromicro Conference on Software Engineering and Advanced Applications*, 2014, pp. 141–148.

[7] M. Broy, I. Kruger, A. Pretschner, and C. Salzmann, "Engineering Automotive Software," in *Proceedings of the IEEE*, ser. 2, vol. 95, 2007.

[8] D. Durisic, M. Staron, and M. Tichy, "Quantifying Long-Term Evolution of Industrial Meta-Models - A Case Study," in *Proceedings of the International Conference on Software Process and Product Measurement*, 2014, pp. 104–113.

[9] B. Boss, "Architectural Aspects of Software Sharing and Standardization: AUTOSAR for Automotive Domain," in *Proceedings of the International Workshop on Software Engineering for Embedded Systems*, 2012, pp. 9–15.

[10] C. Wang, L. Ge, and T. Lee, "Automotive ECU Software Design Based on AUTOSAR," *Journal of Applied Mechanics and Materials*, vol. 577, pp. 1034–1037, 2014.

[11] D. Di Ruscio, L. Iovino, and A. Pierantonio, "Evolutionary Togetherness: How to Manage Coupled Evolution in Metamodeling Ecosystems," in *Proceedings of the 6th International Conference on Graph Transformations*, 2012, pp. 20–37.

[12] D. Mendez, A. Etien, A. Muller, and R. Casallas, "Towards Transformation Migration After Metamodel Evolution," in *Proceedings of the Model and Evolution Workshop*, 2010, pp. 20–37.

[13] F. A. Fontana, M. Rolla, and M. Zanoni, "Capturing Software Evolution and Change through Code Repository Smells," *Lecture Notes in Business Information Processing*, vol. 199, pp. 148–165, 2014.

[14] L. Li, L. Zhang, and Z. Fan, "The Measurement and Analysis of Software Change Based on Software Repository," in *Proceedings of the International Conference on Software Engineering and Data Mining*, 2010, pp. 289–294.

[15] W. Poncin, A. Serebrenik, and M. V. D. Brand, "Process Mining Software Repositories," in *Proceedings of the European Conference on Software Maintenance and Reengineering*, 2011, pp. 5–14.

[16] *OMG. MOF 2.0 Core Final Adopted Specification*, Object Management Group, www.omg.org, 2004.

[17] L. Briand, S. Morasca, and V. Basili, "Property-based Software Engineering Measurement," *IEEE Transactions on Software Engineering*, vol. 22, no. 1, pp. 68–86, 1996.

[18] K. Hyoseob and C. Boldyreff, "Developing Software Metrics Applicable to UML Models," in *Proceedings of the Workshop on Quantitative Approaches in Object-Oriented Software Engineering*, 2002.

[19] J. Branke, K. Deb, K. Miettinen, and R. Słowiński, "Interactive and Evolutionary Approaches," in *Multiobjective Optimization.* Springer Berlin Heidelberg, 2008.