Using XPath to Define Design Metrics

Jan Söderberg, Ali Shahrokni
Systemite AB
Gothenburg, Sweden
e-mail: {jan.soderberg, ali.shahrokni}@systemite.se

Bashar Nassar
Chalmers University of Technology
Gothenburg, Sweden
e-mail: bashar.h.nassar@gmail.com

***Abstract*—Architecture description formats like EAST-ADL and automotive open system architecture (AUTOSAR) use an extensible markup language (XML) based file representation. The complexity of the systems based on these architecture description languages often call for metrics definitions for the purpose of complexity or completeness management. The Swedish research project *Synligare* deals with improved management of complex systems based on EAST-ADL. One result from the project was that XPath could be used as a basis for the definition of design metrics, offering several advantages. XPath has further been demonstrated in the project to offer sufficient expressiveness and usability for the purpose.**

Keywords-Metrics; XPath; EAST-ADL; AUTOSAR; Exchange metrics.

I. Introduction

The evolution rate of automotive electric/electronic(E/E) systems has increased exponentially during the last decade, and the number of electronic control units now typically amounts to 50-100 [1].New and complex functionalities and technologies are emerging, making the prospect of autonomous driving within reach [2]. A consequence of the higher complexity is that the classical document and file based methods are no longer sufficient to manage the product and process data. We have seen that the Software specification of a single Electronic Control Unit (ECU)can be in excess of 8.000 pages. Meanwhile, there is an increased demand for reduced development cycles and product costs.

Synligare[1] is a Swedish industrial research project that aims to improve methods and tool support for model-based development of automotive E/E systems within and between organizations [7]. The members of the Synligare project include Volvo AB, ArcCore AB, Autoliv AB, Semcon and Systemite AB. The parties represent the different roles in a typical E/E development project, including Volvo as a manufacturer and integrator ("OEM" in current automotive terminology), Autoliv as a Tier 1 supplier, ArcCore as a Tier 2 supplier, Systemite as a high level modeling tool supplier on high levels of abstraction, ArcCore as low level modeling tool supplier, and Semcon as a specialist engineering service supplier.

The project uses the EAST-ADL language [8] as a common specification for exchanging developed data within and between organizations. EAST-ADL is an adaptation of SysML[9] for automotive E/E systems. The language includes support for high level specifications of the system, for instance, vehicle features, down to the implementation level, based on AUTOSAR[10]. The language includes optional packages for modeling of variability, timing, safety, and more.

One of the main objectives of the Synligare project is to enable exchange of functional safety data inside and across organizations. ISO 26262 is a standard for functional safety that challenges the automotive industry. The data is produced on different location by different companies. However, the progress needs to be measured, updated, and consolidated in different companies and exchanged between suppliers and OEMs. Many process and products metrics in the ISO 26262 standard are valid across organization boundaries. Many of the progress metrics can be extracted from product data. For instance, one such metric is the state of progress of the verification process for all technical safety requirements, or the state of fulfillment of safety goals on different levels of abstractions.

The Synligare project specifically addresses data exchange challenges between OEMs and suppliers. When the exchange is based on a single formalized representation like EAST-ADL the efficiency and quality of the exchange can be significantly improved, since handover of development, tracing impact of changes and analysis of data can be automated.

A remaining challenge when information is shared and exchanged is to assure that all involved parties can interpret the information in the same way. Although the XML based exchange format for EAST-ADL provides a formalization of the information, the way this information is viewed by different parties is not specified; specifically, when it comes to design metrics. For instance, EAST-ADL does not include progress measurements such as completeness or complexity of the design. In the Synligare project, these metrics were originally specified in natural language, with references to the constructs of the language. For specifying the metrics, we used a more formal alternative, inspired by XPath expressions[11], to express the metrics. These metrics could then be shared between different tools at the OEM and supplier sides to calculate the metrics in a unified way. Using common metrics enables the different groups and organizations to share a common view of the progress of the project. In this paper, we introduce this method of sharing metrics on model-based development data.

The remainder of this paper is organized as follows. In Section II, definition of the EAST-ADL Language, while metrics using path queries defined in Section III. Section IV presents the implementation aspects of XPath, while Section

[1]Synligare means “more visible” in Swedish.

V dissection and conclusion, and Section VI gives a vision for future work.

II. THE EAST-ADL LANGUAGE

EAST-ADL is a domain specific architecture description language specialized for describing automotive E/E systems. The language supports the use of different levels of abstraction with traceability between the levels. The logical structure of an architecture expressed in EAST-ADL is according to a structural component model where components are connected through ports.

EAST-ADL defines an exchange format in XML, called EAXML [12]. The schema of the EAXML is the most precise definition of the language, although the underlying meta-model is defined in UML. The mapping between the meta-model and the XML schema is according to patterns defined in the AUTOSAR community. According to these patterns the schema becomes a reflection of the meta-model, and the schema will only include elements according to the meta-model.

Note that the principles behind the EAXML and ARXML (AUTOSAR xml) schemas differ from the schema of the XMI format, used for the representation of UML models; XMI is based on the more generic MOF (Meta Object Facility) framework [13]. This means that the schema of XMI will not reflect the used meta-model, but rather the meta-meta-model according to MOF. A consequence of importance to the use of XPath is that the element structure of an EAXML file is a direct reflection of the corresponding EAST-ADL model.

III. METRICS DEFINITIONS USING PATH QUERIES

XPath 2.0 became a W3C recommendation 2007. XPath is a specialized query language that can express selection criteria of nodes of an XML document, typically from within an XML style sheet. The selection criteria include the path to traverse in the structure of the document, and additional tests and predicates that must be fulfilled for the selected nodes.

The way XPath is used is by 1) selecting the sets of nodes in the XML document that are relevant for the specific metrics, and 2) performing arithmetic operations on the quantities defined by the sets.

In this section, we present two types of metrics that we have specified with path queries and shared between object model tools. The metrics are inspired by the XPath query language for XML files. The first type of metrics calculates the progress of the development process using the product data. The second type of metrics calculated the complexity of the product components.

A. Progress metrics

One type of the metrics that we defined and shared between tools extracts the state of the project from the development data specified in different tools. The underlying specification of the tools is EAST-ADL, which enables us to create generic metrics and share them between tools. One such metric describes the completeness of the allocation of requirements.

The metrics value was originally expressed in the Synligare project as: "Progress of requirement allocation is measured as the fraction of requirements allocated to architectural elements"

The two sets of elements involved in this calculation are 1) the set of all requirements, and 2)the set of allocated requirements.

The first set can be expressed as the path expression (1) below, which is assumed to start from a "EA-PACKAGE" context node of the EAXML document. Definition for different elements of the XML representation of the meta-model such as EA-PACKAGE is available on EAST-ADL's language specification documentation [8].

Note that since the EA-PACKAGE structure in an EAXML document is an arbitrary packaging structure, it is suitable to exclude this part from the definition, and define the part on a case to case basis.

```
/ELEMENTS/REQUIREMENTS-
MODEL/REQUIREMENTS/REQUIREMENT          (1)
```

The set of allocated requirements is a subset of the set described above, with the additional constraint that the requirement must be included in a so called "Satisfy" relationship:

```
/ELEMENTS/REQUIREMENTS-MODEL/OWNED-
RELATIONSHIPS/SATISFY/SATISFIED-
REQUIREMENT-REFS/SATISFIED-REQUIREMENT-
REF                                     (2)
```

The set of unallocated requirements can be defined as the difference between the two sets, using the "except" operation:

```
/ELEMENTS/REQUIREMENTS-
MODEL/REQUIREMENTS/REQUIREMENT except
/ELEMENTS/REQUIREMENTS-
MODEL/REQUIREMENTS/REQUIREMENT          (3)
```

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-522-7

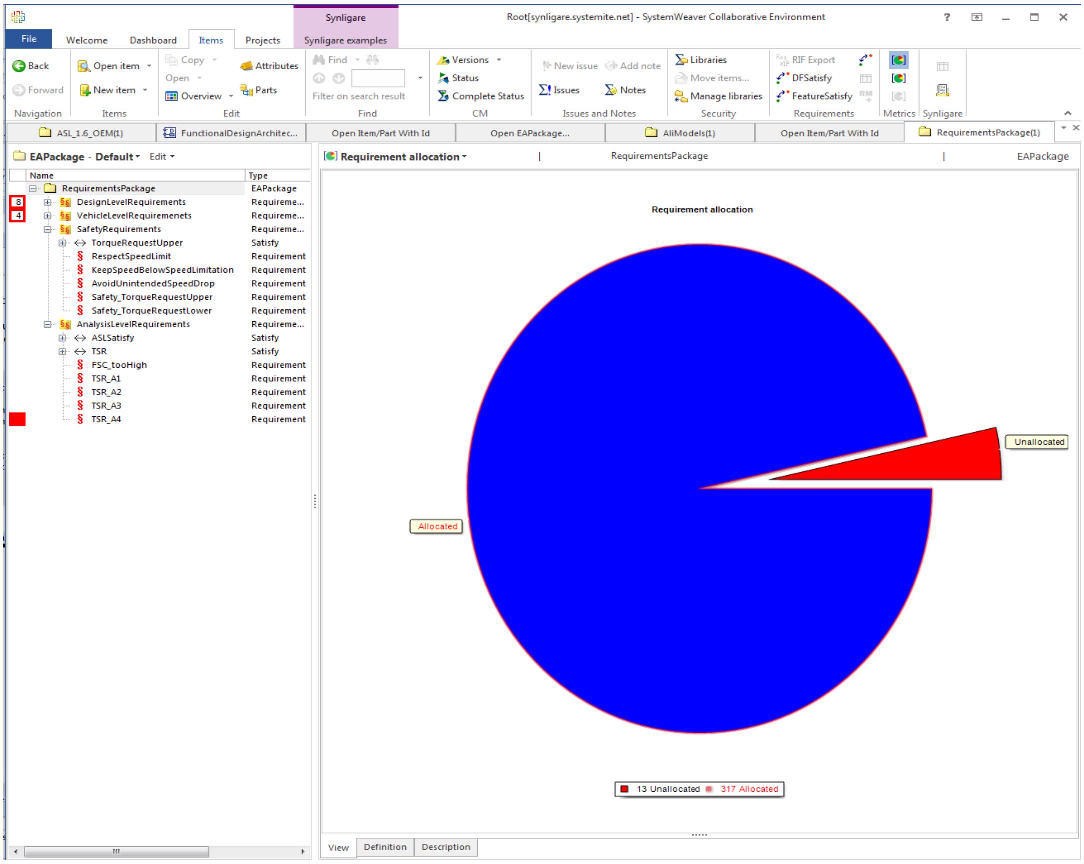

Figure 1 Completeness of allocated requirements

The fraction of the sets can be calculated using the XPath count function and div operator:

```
count(/ELEMENTS/REQUIREMENTS-MODEL/OWNED-RELATIONSHIPS/SATISFY/SATISFIED-REQUIREMENT-REFS/SATISFIED-REQUIREMENT-REF) div count(/ELEMENTS/REQUIREMENTS-MODEL/REQUIREMENTS/REQUIREMENT)          (4)
```

The real underlying need behind this metric is the need for traceability to the set of unallocated requirements. This traceability can be performed interactively using a pie chart representation of the set (3) in the SystemWeaver tool [14]. We see the evaluated system in the tree view to the left in Figure 1. The system is the reference system of the Synligare project, supplied by Volvo. The package "RequirementsPackage" has been selected, thereby selecting the context of the evaluation. The "Requirements allocation" view to the right displays a pie chart, where the two slices represent allocated requirements (in blue) and unallocated requirements (in red). By selecting the *Unallocated* slice, the set of model elements according to the XPath expression (3) become highlighted in the tree view.

B. *Complexity of component models*

Another type of metric that we investigated in this paper is the metrics concerning complexity of component models. One such complexity metric is cyclomaticcomplexity [5], calculated for a component model.

```
count(/CONNECTORS/FUNCTION-CONNECTOR)–count(/PARTS/DESIGN-FUNCTION-PROTOTYPE) + 2          (5)
```

Another component complexity metric uses couplings between objects [6]

```
count(/CONNECTORS/FUNCTION-CONNECTOR) div count(/PARTS/DESIGN-FUNCTION-PROTOTYPE)          (6)
```

IV. IMPLEMENTATION ASPECTS OF XPATH

In the Synligare project, support for metrics definitions expressed by the path query language was implemented in the SystemWeaver tool. SystemWeaver has a programmable meta-model and constitutes an internal database that can manage and integrate the content of multiple EAXML files. The constructs supported by the meta modeling framework in the tool supports the patterns used in EAST-ADL, like the type/prototype pattern. This means that the internal representation in SystemWeaver to a high degree conforms to the EAXML file format. A database like the one in SystemWeaver is not limited to managing the content corresponding to a single system, but can manage any number of systems, and content shared between the systems.

SystemWeaver supports dimensions of data that is not supported by EAST-ADL, like versioning and management of contexts that go beyond the scope of a single system. Such dimensions correspond to additional axes of the XPath expressions that cannot be derived from the specific meta-model.

A specific challenge is the way references are expressed according to EAST-ADL and AUTOSAR. Instead of common XML ID/IDREF to express references, EAST-ADL and AUTOSAR uses element paths of the XML file to reference elements, e.g.,"/DesignLevelElements/ FCN/GlobalBrakeController/BrakeTorqueFL".

References like the one described above are common in the AUTOSAR/EAST-ADL models and means that the XPath expressions cannot be evaluated against a DOM (Document Object Model). Instead, the XML file has to be parsed and transformed into a custom object model where references have been replaced by object links. SystemWeaver for example represents the references as bi-directional object links. During an import of an EAXML file into SystemWeaver all path strings are parsed and replaced with object links.

It can be assumed that any tool that supports EAST-ADL or AUTOSAR will have an efficient internal representation of such references. We have seen that a real life AUTOSAR XML file can be of the size of 10 Mbyte or more, including more than 100,000 elements. A corresponding EAST-ADL model would include even more aspects, and thereby more elements. This means that efficiency becomes a real concern, especially when the evaluation of metrics is done interactively, or when the complexity of XPath expressions are $O(n^2)$ or higher, for instance, when set operations are used.

V. CONCLUSION AND DISCUSSION

In this paper, we presented a generic method to formalize metrics and share them between model-based data management tools. In the Synligare project, metrics originally expressed in natural language have been re-expressed in an XPath-like format and executed in different tools with identical results.

Being XML based, Xpath is intended for use with XML based representations. Since XPath is implementation independent it can work as a formal definition of the metrics, while also being executable.

Elwakil et al. [4] identified a number of advantages of using XQuery in metrics definitions for XMI based representations. These advantages have been found to hold also for XPath, being a subset of XQuery, for the case that data is represented in the more basic XML representations used for AUTOSAR or EAST-ADL:

- The XPath expressions can be expressed according to the meta-model of the used architecture language, meaning that the correctness of the expressions can be validated statically.
- The XPath language is standardized, technology independent, mature and wide spread.
- A tool implementation of the method may directly interpret and execute the XPath expressions. This makes it easy to try different metrics expressions in the tool implementation, without changing the tool itself.

In addition to these findings, the implementation of the support for XPath has taken benefit from the fact that XPath supports the selection of sets of elements, thus making it suitable for interactive analysis and traceabilitybetween the visualization of the metrics and the underlying data.

The solution has been demonstrated using industrial examples, with satisfactory performance.

There are some natural limitations and disadvantages of using XPath:

- The approach is likely feasible only for those cases where the language is expressed as XML; specifically, that the schema is a reflection of the used meta-model.
- Given the declarative characteristics of the language it is likely that not all types of metrics can be defined easily in the language. The use of XQuery as described in [3] has not been investigated for the type of representation used in the project, but may be an alternative for more complex types of metrics.

VI. FUTURE WORK

The evaluation of XPath for metrics definitions described in this paper was limited to the use cases of the Synligare project. It remains to evaluate the suitability of the approach for other types of metrics.

ACKNOWLEDGEMENTS

This work was supported by VINNOVA under the FFI Programme, Grant 2013-2196 Synligare.

REFERENCES

[1] D. Goswami et al., “Challenges in automotive cyber-physical systems design.” In Embedded Computer Systems (SAMOS), International Conference on 2012,pp. 346-354. IEEE.

[2] R. Okuda, Y. Kajiwara, and K. Terashima, “A survey of technical trend of ADAS and autonomous driving.” In VLSI Technology, Systems and Application (VLSI-TSA), Proceedings of Technical Program-International Symposium on 2014, pp. 1-4. IEEE.

[3] M. Sharma, N. S. Gill, and S. Sikka,“Survey of object-oriented metrics: Focusing on validation and formal specification.” ACM SIGSOFT Software Engineering Notes, 2012, 37.6: 1-5.

[4] M. El-Wakil, A. El-Bastawisi, M. Riad, and A. Fahmy, “A novel approach to formalize and collect Object-Oriented Design-Metrics.” 9th International Conference on Empirical Assessment in Software Engineering, 2005.

[5] Y.Jiang, B. Cuki, T. Menzies, andN. Bartlow,“Comparing design and code metrics for software quality prediction.” In Proceedings of the 4th international workshop on Predictor models in software engineering, 2008, pp. 11-18. ACM.
[6] S. S.Rathore and A. Gupta, (2012, September). “Investigating object-oriented design metrics to predict fault-proneness of software modules.” In Software Engineering (CONSEG), CSI Sixth International Conference on2012, pp. 1-10. IEEE.
[7] Synligare Consortium: Synligare Project website. Available from: http://www.synligare.eu/ [retrieved: Dec, 2015].
[8] EAST-ADL Association: EAST-ADL web site. Available from: http://www.east-adl.info/ [retrieved: Dec, 2015].
[9] SysML. Available from: http://www.omgsysml.org/ [retrieved: Dec, 2015].
[10] AUTOSAR Development Partnership: AUTOSAR web site. Available from: http://www.autosar.org/ [retrieved: Dec, 2015].
[11] XPath. Available from: http://www.w3.org/TR/xpath/ [retrieved: Dec, 2015].
[12] EAXML specification. Available from: http://www.east-adl.info/2.1.12/eastadl_2-1-12.xsd [retrieved: Dec, 2015].
[13] MOF. Available from: http://www.omg.org/mof/ [retrieved: Dec, 2015].
[14] SystemWeaver. Available from: www.systemweaver.se [retrieved: Dec, 2015].