Das AUTOSAR XML Schema und seine Bedeutung für die Implementierung von AUTOSAR Werkzeugen

Uwe Honekamp

PND – Tools for Networks and distributed Systems
Vector Informatik GmbH
Ingersheimer Straße 24
70499 Stuttgart
uwe.honekamp@vector-informatik.de

Abstract: Das AUTOSAR XML Schema wird als das Ergebnis eines komplexen Engineering-Prozesses aus dem sog. AUTOSAR Metamodell erzeugt. Seine Hauptaufgabe besteht in der Unterstützung des Datenaustauschs zwischen AUTOSAR Werkzeugen. Das Schema deckt den gesamten Modellierungskontext (die sogenannten Templates) von AUTOSAR ab.
Dieser Beitrag diskutiert auf der Basis von Erfahrungen aus realen Softwareprojeten die Bedeutung des AUTOSAR XML Schemas unter besonderer Berücksichtigung der Implementierung von AUTOSAR Werkzeugen.

1 Einleitung

Eines der wesentlichen Paradigmen der AUTOSAR Initiative [AR06] [AR08] beruht auf der Maßgabe, dass AUTOSAR Werkzeuge jeglicher Art auf der Basis des AUTOSAR XML [W3C06] Schemas [W3C04] beliebig austauschbar und interoperabel gestaltet werden können. Auf diese Weise soll die vor der Einführung von AUTOSAR bestehende Abhängigkeit von einzelnen Herstellern erfolgreich vermieden werden.

Das primäre Ziel des AUTOSAR Schema [PB06] [BK07] ist somit die Definition eines Datenaustauschformats für automotive Software. Dabei erstreckt sich der Gültigkeitsbereich im Gegensatz zu den früheren Ausgaben des Schemas mittlerweile bis in die formale Beschreibung der AUTOSAR-Basissoftware hinein.

Das AUTOSAR XML Schema entsteht als Ergebnis eines komplexen Generierungsprozesses an dessen Anfang das sog. AUTOSAR Metamodell steht. Letzteres definiert die zur Modellierung zur Verfügung stehenden Metaklassen sowie deren Beziehungen zueinander. Mit anderen Worten: die konkrete Formulierung des AUTOSAR Metamodells wird in das AUTOSAR XML Schema überführt.

Im Metamodell wird somit beispielsweise festgelegt, dass eine AUTOSAR Software-Komponente über Ports verfügt, die (entsprechend der Art des verwendeten Port-Interfaces) entweder eine Sender/Receiver- oder eine Client/Server-Kommunikation unterstützen.

Dabei deckt das XML Schema im Wesentlichen den syntaktischen Teil des Datenaustauschs ab. Der semantische Teil, d.h. die Definition von **semantischen Constraints** kann vom Schema naturgemäß nicht abgedeckt werde, ist aber **mindestens von ebenso großer Bedeutung** für den eigentlichen Datenaustausch.

Im Rahmen einer weitergehenden Standardisierung könnte seitens AUTOSAR der semantische Aspekt formal unter Verwendung der *Object Constraint Language* (OCL) [OMG05] im AUTOSAR Metamodell verbindlich definiert werden, dies ist aber bisher nicht erfolgt[1].

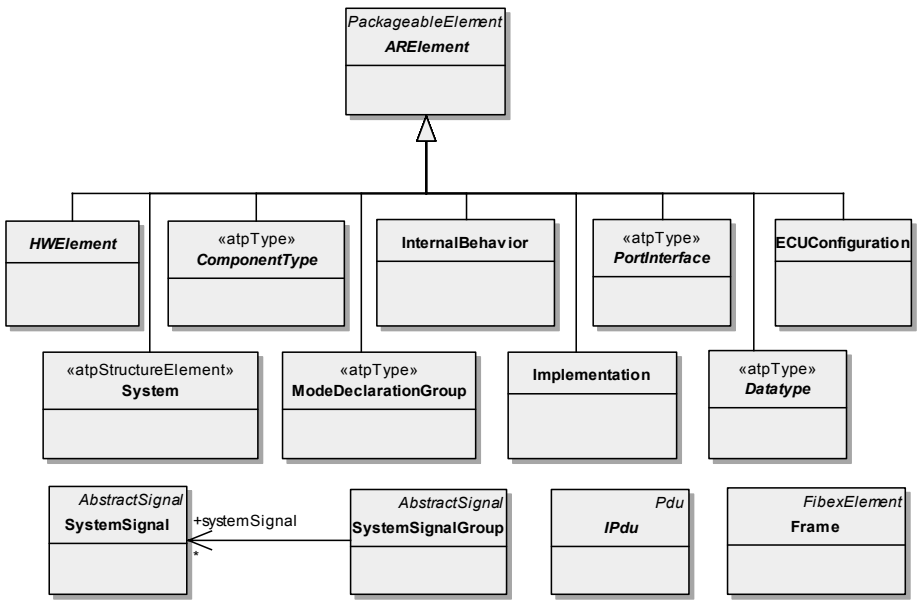

Abbildung 1: AUTOSAR ARElements

Eine im Rahmen der AUTOSAR Entwicklungspartnerschaft mehr oder weniger ungeklärte Fragestellung ist die konkrete Bedeutung des AUTOSAR XML Schemas für die Implementierung von AUTOSAR Werkzeugen.

[1] Dies würde nicht zwingend bedeuten, dass AUTOSAR Werkzeuge direkt OCL Code verarbeiten können müssen. Die Überprüfung der Constraints kann in beliebiger Weise implementiert werden, solange die zugrundeliegende Semantik erhalten bleibt.

2 Das AUTOSAR XML Schema

Wie bereits erwähnt, definiert das AUTOSAR XML Schema eine **Austauschplattform für AUTOSAR Werkzeuge**. Diese Festlegung bestimmt wesentliche Eigenschaften des AUTOSAR XML Schemas.

Eine Kenngröße für die Tauglichkeit eines Austauschformats ist die Granularität der Artefakte, d.h. Änderungen eines bestimmten Artefakts sollten möglichst nicht zur Notwendigkeit des Austauschs einer gesamten Systembeschreibung führen.

Besonderer Wert bei der Definition des Metamodells wurde und wird daher auf **maximale Wiederverwendbarkeit und Modularität** der Elemente des Metamodells und somit der auf der Basis des XML Schemas definierten XML Artefakte gelegt.

Das AUTOSAR Metamodell definiert zu diesem Zweck Artefakte mit globaler Sichtbarkeit (sogenannte ARElements, siehe Abbildung 1), die zusammen mit der Möglichkeit zur Definition von sogenannten ARPackages letztlich die minimale Granularität eines konkreten AUTOSAR XML Artefakts bestimmen.

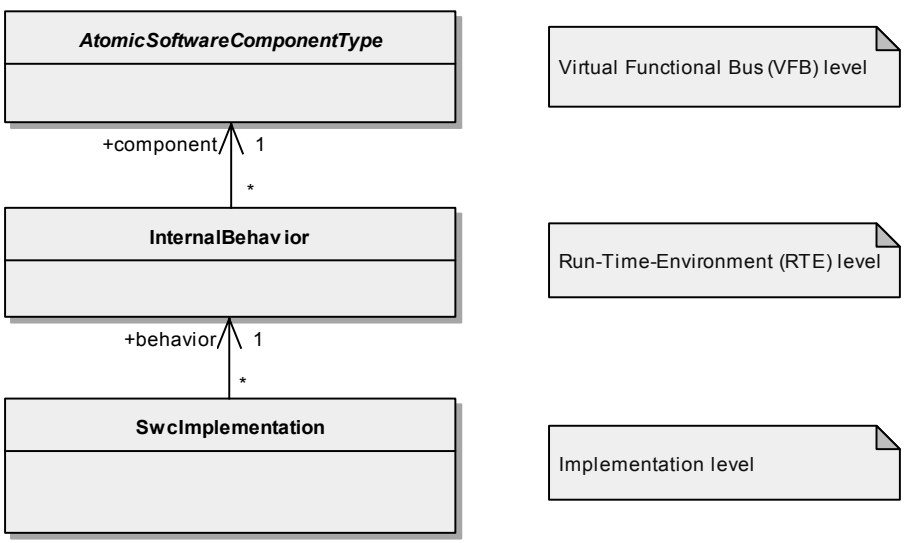

Abbildung 2: Drei-Schichten-Architektur im Software Component Template [SWC07]

Ein besonders eindrucksvolles Beispiel für die Anwendung dieses Grundgedankens ist die "Drei-Schichten-Architektur" des sog. Software Component Template [SWC07], siehe Abbildung 2.

Die oberste Schicht dieser Architektur repräsentiert die Sicht auf den sog. Virtual Function Bus (VFB) [VFB07], die zweite Schicht fügt einige für die Konfiguration des sog. Runtime Environment (RTE) [RTE07] notwendige Informationen hinzu und die dritte Schicht konzentriert sich schließlich auf implementierungsspezifische Inhalte.

Dabei erfolgt die Richtung der Referenzierung von "unten nach oben", d.h. die Metaklasse *SwcImplementation* referenziert die Metaklasse *InternalBehavior* und diese wiederum referenziert *AtomicSoftwareComponentType*.

Auf diese Weise ist es möglich, einen bestimmten *AtonicSoftwareComponentType* per Referenzierung mit mehreren *InternalBehaviors* auszustatten, die ihrerseits wiederum jeweils von mehreren *SwcImplementations* referenziert werden können.

Entsprechend dem Grundgedanken der maximalen Modularität können somit mehr oder weniger beliebige Kombination von *AtomicSoftwareComponentType*, *InternalBehavior* und *SwcImplementation* hergestellt und ausgetauscht werden. Eine Änderung von beispielsweise einer bestimmten *SwcImplementation* erfordert lediglich den Austausch dieser *SwcImplementation*, aber nicht des referenzierten *InternalBehavior*.

3 Relevante Eigenschaften von AUTOSAR Werkzeugen

Ein AUTOSAR Werkzeug muss in der Lage sein, AUTOSAR XML Artefakte (ggf. unter Bezug auf unterschiedliche Versionen des AUTOSAR XML Schemas) lesen und auswerten sowie (sofern as sich um ein Autorentool handelt) aus Benutzereingaben AUTOSAR XML Artefakte erzeugen zu können [IA07].

Ein AUTOSAR Werkzeug ist allerdings nicht verpflichtet, relevante Modellinhalte in der vom AUTOSAR Schema implizierten Form zu präsentieren, es kann das Modell so aufbereiten, dass eine optimierte und möglichst performante Benutzerführung ermöglicht wird[2].

Insbesondere sind AUTOSAR Werkzeuge nicht verpflichtet, graphische Inhalte in einer standardisierten Form anzubieten oder auszutauschen. Die von AUTOSAR zu diesem Thema veröffentlichte "Graphical Notation" [GN07] besitzt nur Vorschlagscharakter.

4 Verwendung des AUTOSAR XML Schemas für die Implementierung von AUTOSAR Werkzeugen

Obwohl – wie bereits dargelegt – das AUTOSAR XML Schema als Austauschdatenformat ausgelegt wurde, wird vielerorts die direkte Verwendung des XML Schemas für die Implementierung von AUTOSAR Werkzeugen propagiert.

---

[2] Diese Aussage steht ausdrücklich nicht im Widerspruch zu dem von AUTOSAR ausgegebenem Motto " "Cooperate on standards, compete on implementation" [AR08]

Diese Vorgehensweise erscheint insbesondere unter Verwendung softwaretechnischer Unterstützung zur direkten Umsetzung eines XML Schemas in die Implementierung von Quellcode attraktiv. Mit vergleichsweise geringem Aufwand kann bereits ein wesentlicher Teil der Implementierung eines AUTOSAR Werkzeugs erledigt werden.

Der beschriebene Ansatz wird beispielsweise durch das "Eclipse Modeling Framework" [BUD03] (Eclipse EMF) unterstützt. Als Ergebnis der Transformation des XML Schemas entsteht ein Satz von Java-Klassen [SUN08], deren Methoden direkt aus den Beziehungen der AUTOSAR Modellelemente untereinander abgeleitet werden können. Mit anderen Worten: ein AUTOSAR XML Parser kann die XML Elemente direkt in Instanzen der erwähnten Java-Klassen umformen.

Als weiteren Optimierungsschritt kann man den erwähnten Transformationsvorgang auch mehr oder weniger direkt auf dem AUTOSAR Metamodell aufsetzten und kann somit den bereits verlustbehafteten Transformationsschritt vom Metamodell zum Schema umgehen. Das Generierungsverfahren unterstützt außerdem in gewissen Grenzen zusätzlich die Erzeugung von graphischen Editoren.

Trotz der vordergründig attraktiven Möglichkeiten dieser Werkzeuge kann nicht entgehen, dass dabei die auf maximale Wiederverwendbarkeit ausgelegte Grundarchitektur des AUTOSAR Metamodells/Schemas direkt als Grundlage für die Werkzeugimplementierung verwendet wird.

Maximale Modularität auf der einen Seite sowie die Forderung nach einer kompakten, performanten, und an den Anforderungen einer sinnvollen Benutzerführung orientierten Werkzeugimplementierung auf der anderen Seite sind allerdings nicht in jedem Fall in Übereinstimmung zu bringen. Beispiel (mit Bezug auf die in Abbildung 2 diskutierte "Drei-Schichten-Architektur"):

Bei Bearbeitung eines *AtomicSoftwareComponentType* in einem AUTOSAR Werkzeug ist es erforderlich, auch Eigenschaften der "mittleren" Ebene, beispielsweise die Definition von *RunnableEntities*, zu berücksichtigen. Bei direkter Umsetzung des AUTOSAR Metamodells müssten dazu erst einmal vergleichsweise umständlich alle vorhandenen *InternalBehaviors* durch Iteration und Vergleich über alle vorhandenen *InternalBehaviors* identifiziert werden.

Daher erscheint die Definition und Implementierung eines speziell auf die Bedürfnisse einer AUTOSAR Werkzeugimplementierung und vom AUTOSAR XML Schema abstrahierten Metamodells sinnvoll, auch wenn dies **vordergründig** mit einem höheren Aufwand verbunden ist.

Weitere wichtige Vorteile eines vom AUTOSAR XML Schema abstrahierten, selbstdefinierten Werkeug-Metamodells sind die Möglichkeit zum **Mischen von Inhalten aus unterschiedlichen Versionen des AUTOSAR XML Schemas** sowie die Möglichkeit zur **Migration von XML Content von einer auf eine andere Version** des Schemas.

Dennoch ist das AUTOSAR Metamodell bzw. XML Schema von direkter Bedeutung für die Implementierung von AUTOSAR Werkzeugen. Der Import/Export Mechanismus eines AUTOSAR Werkzeugs orientiert sich naturgemäß sehr viel stärker an der Definition des Schemas als am werkzeuginternen Metamodell.

Die Bedeutung dieser Aussage wird anhand einer exemplarischen Implementierung eines AUTOSAR-konformen Importmechanismus deutlich. Eine wesentliche Voraussetzung für einen fehlerfreien Importvorgang ist die Überprüfung der Konsistenz des importierten XML Contents.

Die Forderung nach Konsistenz bezieht sich primär auf syntaktische Korrektheit, beinhaltet aber auch die Überprüfung der semantischen Constraints. Der Import kann daher durchaus unter Verwendung des AUTOSAR Metamodells durchgeführt werden.

Dabei wird der XML Content zunächst vollständig in ein AUTOSAR-konformes Metamodell überführt und anschließend intensiv auf seine **semantische Konsistenz** geprüft. Verläuft diese Prüfung erfolgreich, können die importierten Artefakte den "Quarantänebereich" verlassen und in das werkzeuginterne Metamodell überführt werden.

Literaturverzeichnis

[AR08] AUTOSAR Entwicklungspartnerschaft, www.autosar.org, 2008

[AR06] Fennel, et. al.: Achievements and exploitation of the AUTOSAR development partnership, Convergence 2006, Cobo Center, Detroit, Michigan, 16.-18. Oktober 2006

[SWC07] AUTOSAR GbR: Software Component Template, V3.0.0, AUTOSAR Release 3.0, 2007.

[VFB07] AUTOSAR GbR: Specification of the virtual Functional Bus, V1.0.0, AUTOSAR Release 3.0, 2007.

[RTE07] AUTOSAR GbR: Specification of RTE, V2.0.0, AUTOSAR Release 3.0, 2007.

[GN07] AUTOSAR GbR: Specification of Graphical Notation, V1.0.4, AUTOSAR Release 3.0, 2007.

[IA07] AUTOSAR GbR: Interoperability of AUTOSAR Authoring Tools, V1.0.4, AUTOSAR Release 3.0, 2007.

[BK07] Brörkens, M.; Köster, M.: Improving the Interoperability of Automotive Tools by Raising the Abstraction from Legacy XML Formats to Standardized Metamodels, n D.H. Akehurst, R. Vogel, and R.F. Paige (Eds.): ECMDA-FA 2007, LNCS 4530, pp. 59–67, Springer-Verlag Berlin Heidelberg 2007..

[PB06] Pagel, M.; Brörkens, M.: Definition and Generation of Data Exchange Formats in AUTOSAR, in A. Rensink and J. Warmer (Eds.): ECMDA-FA 2006, LNCS 4066, pp. 52–65. Springer-Verlag Berlin Heidelberg 2006.

[W3C06] W3C: Extensible markup language (xml) 1.1 (2 edn.)., http://www.w3.org/TR/2006/REC-xml11-20060816/, 2006

[W3C04] W3C: Xml schema part1: Structures second edition, http://www.w3.org/TR/xmlschema-1/, 2004

[OMG05] OMG: UML OCL specification version 2.0., http://www.omg.org/cgi-bin/doc?ptc/2005-06-06, 2005

[BUD03] Budinski, F.; Steinberg, D.; Merks, E.; Ellersick, R.; Grose, T. J.: Eclipse Modeling Framework, Addison Wesley Professional, 2003

[SUN08] Sun Microsystems: java.sun.com