This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

Tools for Forging the Functional Architecture

Andreas Korff[1], Jesko G. Lamm[2], Tim Weilkiens[3]

[1]Atego Systems GmbH, Major-Hirst-Str. 11, 38442 Wolfsburg, Germany,
andreas.korff <atsign>atego.com

[2]Bernafon AG, Morgenstrasse 131, 3018 Bern, Switzerland,
jla <atsign> bernafon.ch

[3]oose Innovative Informatik GmbH, Straßenbahnring 7, 20251 Hamburg, Germany,
tim.weilkiens <atsign> oose.de

Abstract: Functional Architectures enable modeling systems independent of their target technology. Their systematic modeling requires several steps, some of them being trivial, because their result is determined by general rules. If it is possible to automate these steps, the Architect will be able to stay focused on those activities that require his expertise. This paper uses SysML models in Artisan Studio® as an example for showing how certain steps in the creation of functional architectures can be automated or facilitated. These steps do not only involve creating functional blocks from activity trees, but also establishing links between the SysML model and existing models in the enterprise, which may for example have been created with the Simulink® software.

1 Introduction

System architects shape the system and establish links between the different involved technical domains. To describe the system in a way that is independent of its target technology and thus easy to re-use, they may use functional Architectures [LW10]. These require abstract thinking skills [Wei11] and, as a result, the architect has to translate the functional Architecture into the views of the different technical domains.

It is a known approach to create subsystems from certain technical domain with model-based tools like e.g. the MATLAB®/Simulink® [VD06] software, which can be used for functional modeling of systems. The challenge now is to find tools that support the Architect with his holistic task, allowing him to stay focused on his translator role instead of being distracted by trivial modeling activities. Tool support should relief the Architect from recurring, monotonous activities and facilitate the transition into models of the different technical domains.

To explain the issue with recurring, monotonous activities, let us report an observation from a project dealing with hearing instrument technology: while the method from [LW10] was applied, it was noticed that there was a 4 man-hours effort for carrying out and reviewing activities with a predefined flow that was given by the methodology. The stated effort was more than 5% of the project's total architectural modeling effort (the data is based on a retrospective analysis of the project by two of the authors who were

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

part of it). Accordingly, we expect that time can be saved with an approach of automated modeling, not only because model creation will be faster, but also because – once the tool has been verified – it should be possible to reduce reviewing effort due to fewer manual steps involved.

In this paper, models of functional architecture in SysML [OMG10, Kor08a, Wei08] and functional models based on the Simulink® software are used to demonstrate the use of modeling tools in assisting the creation of functional architectures and their linking with existing models. Instead of emphasizing the methodology behind the modeling of functional architecture [LW10], this paper focuses on the efficient use of the methodology with modeling tools. We describe a new concept for automating certain activities of the architect, but also base our work on existing ways of interlinking SysML modeling tools with the Simulink® software [VD06, Kor08b].

2 Method for creating functional architectures

This section introduces an essential concept behind this paper: the method for creating functional architectures from [LW10].

Use cases are the key to obtaining the functional architecture. They represent the services the system offers as seen from outside. Within a framework of functional requirements analysis like SYSMOD [Wei08], one can identify all use cases necessarily needed around a product. Then the details of each use case can be defined by an activity diagram (figure 1). The actions within an activity are the functions of the system. The decomposition of an activity into called activities and the outgoing and incoming objects (object flow) determine the functional architecture.

From the activities, one can obtain activity trees (also called "function trees" in the literature – [Wei08] [1], for example). The main activity of each use case is the respective tree's root, and the hierarchy within the tree (figure 2) has call semantics, meaning that a node or leaf in the tree represents an activity that is called within the context of its parent node. Furthermore, each activity has associations with the types of its incoming and outgoing objects.

Multiple activity trees result from the complete set of use cases for the system. The architect decides, based on his expertise and on Heuristics [LW10], how activities can be assigned to clusters of high cohesion. These are functional groups to be represented in SysML by functional blocks. The functional blocks together represent the function structure (figure 3), whose hierarchy has containment semantics. This means that a functional block is a part of its parent if its functions are subfunctions of its parent. At the root of the function structure, there is a special functional block, which represents the whole functional architecture.

[1] [remark of the translator: the English term "function tree" was taken from an English edition of the referenced book: Weilkiens, T. Sytems Engineering with SysML / UML, Morgan Kaufmann OMG Press, 2007]

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

The function structure defines the elements of the functional Architecture. To describe the inner structure of a functional architecture, internal block diagrams are used (figure 4). Two functional entities are connected in an internal block diagram if their functions exchange data. The data flow can be specified in more detail by ports (figure 4).

3 Example

To demonstrate how a tool can support carrying out the described method, we use a sample system from [LW10], a hearing instrument that is fictitious, because it is simplified compared to the state-of-the-art. It is subject to two use cases (figure 1):

- "Listen to Amplified Signal". The hearing instrument user has the hearing instrument amplify sounds from the environment in order to listen to the result.
- "Adjust Volume". The hearing instrument user changes the volume, eventually more than once, until the perceived level is comfortable.

In the following, a demonstration of tool use will show an application example: the creation of the system's functional architecture based on the two described use cases with support by the tool.

4 Automatic creation of functional architectures

Here is our proposed procedure for the automatic creation of functional architectures:

- Activity diagrams are used for modeling the different activities within each use case. A partition with the stereotype "«I/O»" (figure 1) takes those activities that involve interfaces with connections outside the system (according to the heuristic *one functional group takes the functions that are related to system actors* [LW10]). From a formal point-of-view, activities are only allowed in the I/O partition if they have input and / or output pins providing object flows with external actors. A resulting recommendation is to divide an activity into several ones if it models more than just the interaction via an interface. Then, the interface-related part should be separated from the rest (e.g. divide an eventual activity "Amplify Sound" into the ones we chose here: "Apply Amplification" and "Output Amplified Signal").

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

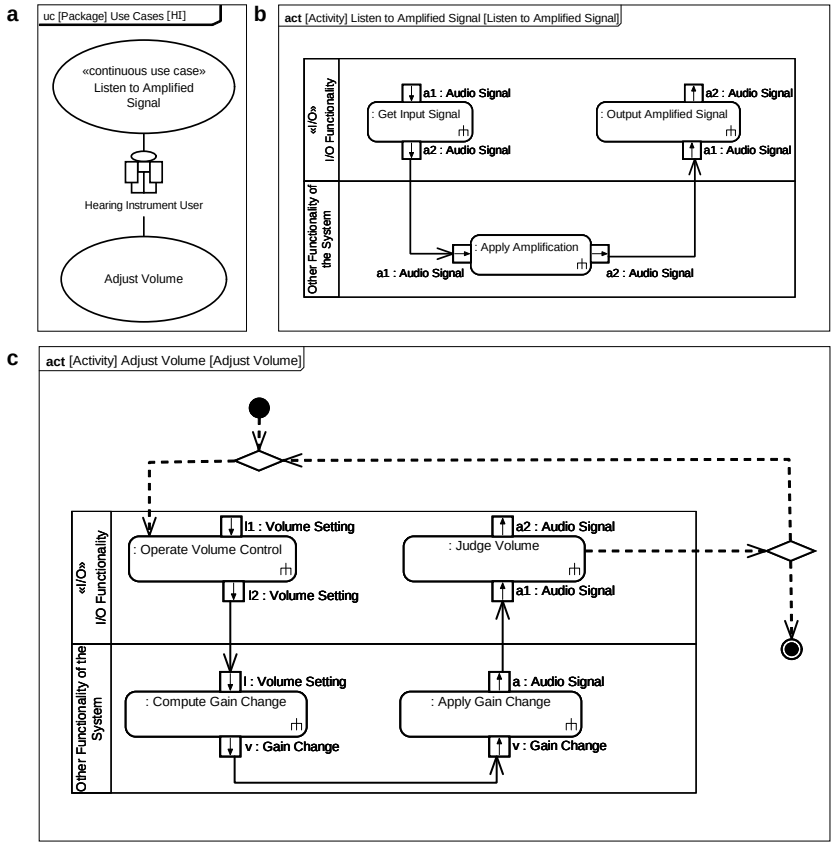

Figure 1. In refining use cases (a) by activities (b, c) these have to be placed in a partition with streotype «I/O» if they involve interfaces with connections outside the system

- The Architect defines the root activities of the activity tree (e.g. by dragging them into an empty diagram) and has the tool create the complete tree structure automatically. The tool also groups activities from an "«I/O»" partition automatically: it automatically connects them with a constraint (figure 2 without the elements marked by black ellipses). The SysML language lacks an element for grouping arbitrary model elements. OMG's SysML working group is currently working on a solution. Our favorite solution from the current discussion is a specialization of the *constraint* element [OMG09]. Even if there is no official standardization, this solution is formally correct and can be used right now. Therefore, we propose to use constraints as a means for grouping activities in our procedure. The tool needs to know which activities to take into account. This could be solved by having the Architect apply a special stereotype to those activities to be accounted for. In the example given here, however, the tool worked according to a more simple approach: it accounted for all activities within the current diagram.

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

- By continuing the grouping of activities that has been initialized automatically according to the previous step, the architect manually completes the assignment of activities to functional groups by connecting all activities that should belong to one functional group with the same constraint object, for each group respectively (figure 2, including the elements that are marked by black ellipses). In this step, only leaves of the activity trees are assigned to different groups. The remaining nodes, including the root of the tree, are assigned to one "system" group. At the end, the assignment of activities with constraints shall define a disjoint decomposition of the set of activities from all trees. According to section 2, also associated data types are visible in the activity tree. These have been omitted in figure 2, with benefits regarding the figure's clarity. It should be noted that it is not possible in all tools to display associated data types.

- The tool automatically creates one functional block per constraint, as the basis for functional architecture (figure 3). This can be done via the use of model transformation languages (e.g. according to [Alt09]) or via proprietary automation means of the tool (which is the approach we used here). The tool automatically connects each new functional block with the constraint from which it has been derived. This is done with a "«trace»" relationship. To carry out the described step, the tool needs to know which constraints to account for and below which element to create the new blocks. While the latter information should be entered by the Architect based on a user prompt appearing at the beginning of the transformation, the constraints to account for should already have been marked during modeling: a special stereotype can be assigned to them. In the example shown here, however, a straightforward approach was used for the sake of simplicity: a constraint has been accounted for, if it was visible in the current diagram.

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

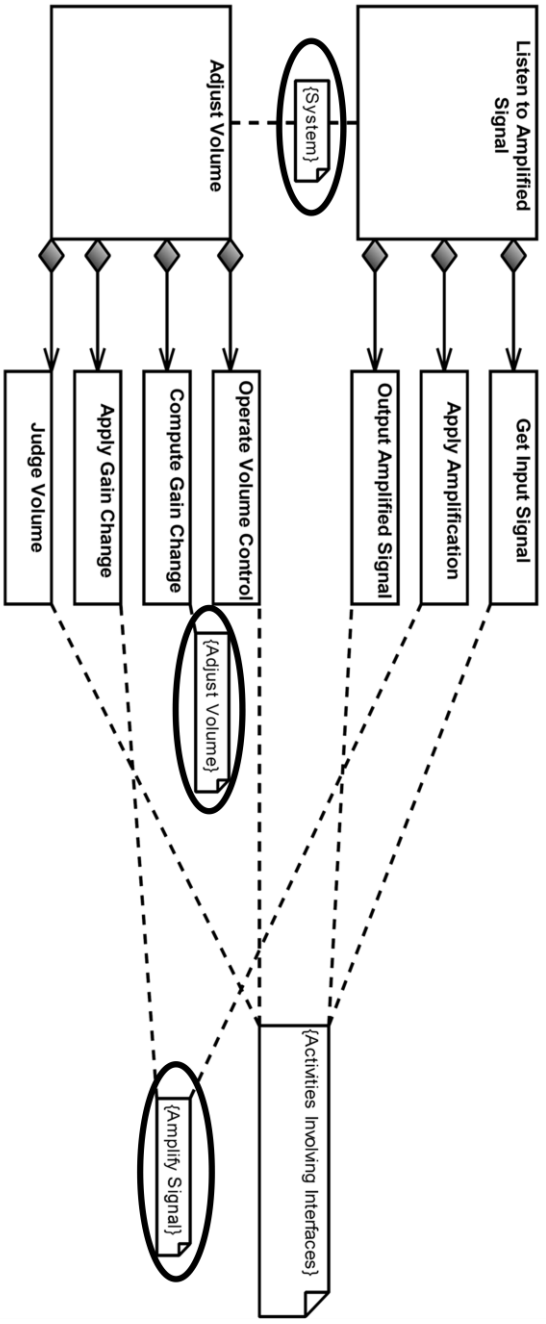

Figure 2. Constraints define the functional groups in an activity diagram

Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

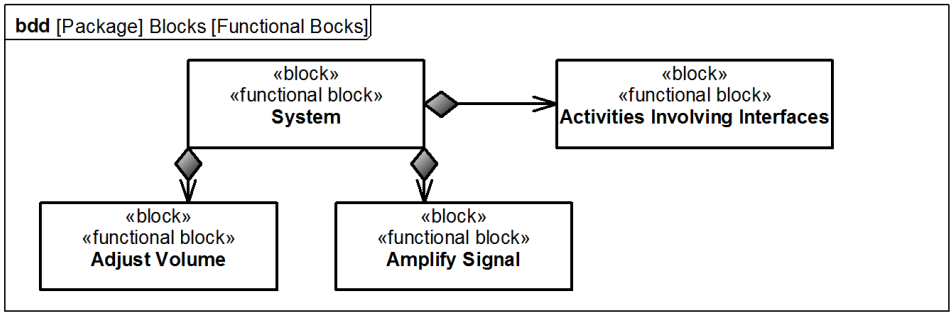

Figure 3. Automatically created blocks of functional architecture and (here: manually created) composition relationships in a block definition diagram

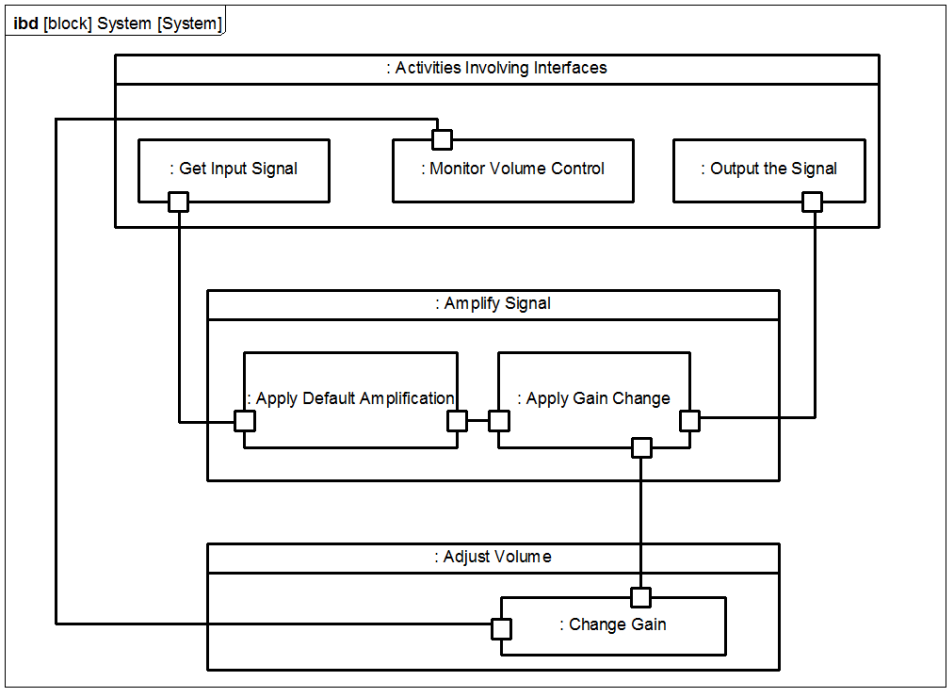

Figure 4. Manually post-processed functional architecture in an internal block diagram

Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

- The architect manually post-processes the functional architecture and refines it (figure 4). While doing so, he can update the constraints that have been used for grouping activities, but he can also delete them ("Discard Temporary Models" [Amb02]).

If functional models have already been created in some of the different technical domains, these can be integrated into the model. This will be described in the next section.

5 Linking to existing functional models

The functional block "Apply Default Amplification" could typically exist in the case in which it would be known as a separate entity from earlier versions of the system (heuristic *use grouping criteria of existing groups* [LW10]). In such cases, the practitioner sometimes has functional models of such blocks in place, e.g. within the Simulink® software. Figure 5 shows a corresponding model of the block "Apply Default Amplification". The system architect should re-use such models instead of again modeling the functionality. Figure 6 shows an according way of linking the example model from figure 5 with the SysML model we have described here.

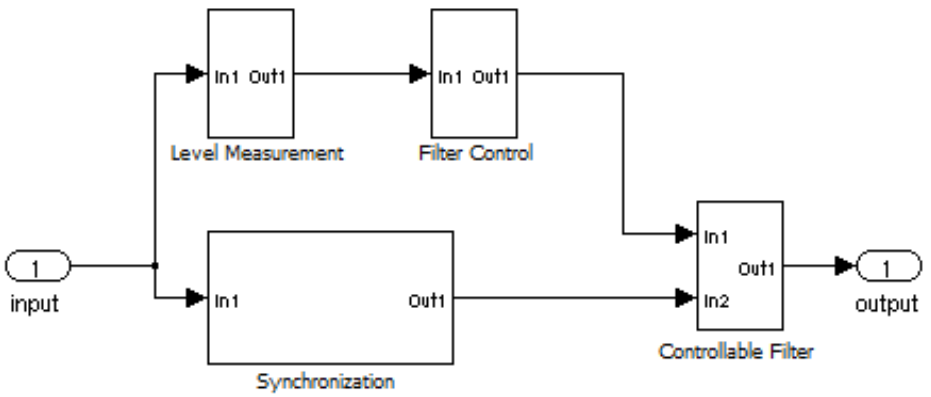

Figure 5: A functional model of the block „Apply Default Amplification", designed according to [Sch05] and created based on the Simulink® software.

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

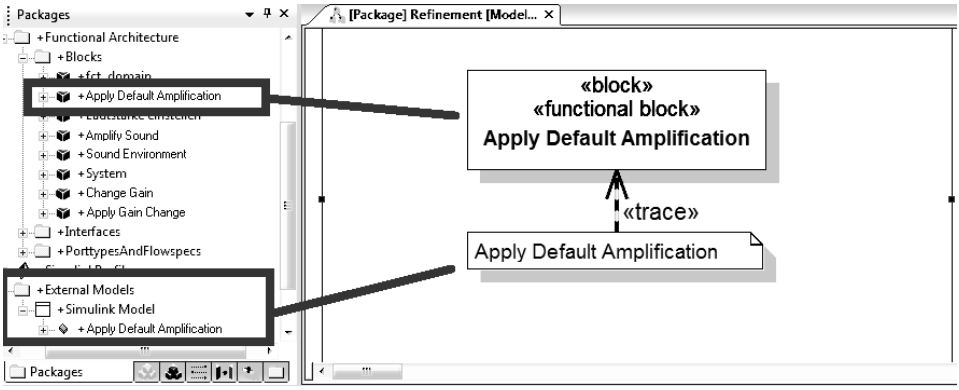

Bild 6: Linking a model originating from the Simulink® software with the functional block "Apply Default Amplification" of the SysML model

6 Discussion, conclusions and outlook

Like the blacksmith, forging the steel in the first place instead of putting focus on lighting the fire in the smithy, the architect should forge the system and needs tools that relief him of trivial but error-prone steps of modeling. UML/SysML tools like Artisan Studio® enable an implementation that carries them out automatically, based on extending the range of offered functionality with appropriate features that provide more than just the automatic creation of diagrams. Care has to be taken that the justification of automation is not in the automation itself, but that automation helps saving time and avoiding errors that result from error-prone modeling steps. The more experienced the architect is with the modeling methodology, the more steps in modeling he will perceive as an undemanding, industrial activity that lacks the need for creativity. As a consequence, the amount of automation should increase with growing experience of the architect. Therefore the following paragraphs will outline a way of automating the creation of architecture more than shown so far.

Interfaces in functional architecture (ports and their connections in figure 4) can be created with tool support, because interfaces between functional blocks depend on the object flows in the underlying activity diagrams. The tool can verify by itself if there are object flows between activities behind different functional blocks and if there are, it can suggest creating an interface between the corresponding blocks. The architect should then verify if the object flows are actually within the scope of the considered system or if they are a side effect (e.g. resulting from the re-use of activities across multiple system models). Furthermore, the modeling of object flows within use cases but also inbetween them needs to be complete as a condition for the correct automatic creation of interfaces. Therefore another kind of automation may be considered: automatic reviews. Instead of providing scripts that change the model and leave the architect with changed reality he has to verify afterwards, the tool can automatically check the model and give the architect plausible advice for his next steps. For example, an automatic review could find activities that are located in «I/O» partitions but have neither input nor output pins. The existence of such pins would indicate that the model could not be complete enough for

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).

This is an English translation of a German conference paper presented at the 2011 conference of the German "Gesellschaft für Systems Engineering e.V.". Please cite this document as follows: "Korff, A.; Lamm, J. G.; Weilkiens, T., Werkzeuge für den Schmied funktionaler Architekturen, Tag des Systems Engineering, November 2011. English translation by J. Lamm."

automatic creation of the architecture model and that extensions of the activity model should be made to provide the missing input and output pins.

7 Acknowledgments

The authors like to thank The MathWorks, Inc. for supporting this work.

References

[Alt09] Alt, O. Car Multimedia Systeme Modell-basiert testen mit SysML. PhD thesis, TU Darmstadt, 2008. Vieweg+Teubner, 2009.

[Amb02] Ambler, S. W. Agile modeling. John Wiley, 2002.

[Kor08a] Korff, A. Modellierung von eingebetteten Systemen mit UML und SysML, Spektrum Akademischer Verlag, 2008.

[Kor08b] Korff, A. Drei Use Cases zur Kopplung funktionaler und Systemmodelle. Tag des Systems Engineering. Bremen, Germany, November 2008.

[LW10] Lamm, J. G. and Weilkiens, T. Funktionale Architekturen in SysML. In M. Maurer und S.-O. Schulze (eds.), Tag des Systems Engineering 2010 S. 109–118. Carl Hanser Verlag, München, Germany, November 2010.

[OMG09] OMG Issue Nr. 13928. http://www.omg.org/issues/issue13928.txt (visited May 18th, 2011)

[OMG10] Object Management Group (OMG): OMG Systems Modeling Language (OMG SysML™) Version 1.2. OMG Document Number formal/2010-06-01, 2010.

[Sch05] Schaub, A. Digitale Hörgeräte. Median-Verlag, Heidelberg, Germany, 2005.

[VD06] Vanderperren, Y. and Dehaene, W. From UML/SysML to Matlab/Simulink: Current state and future perspectives. In Design, Automation and Test in Europe, 2006. DATE '06. Proceedings, S. 1. 2006.

[Wei08] Weilkiens, T.: Systems Engineering mit SysML / UML. dpunkt.verlag, 2008.

[Wei11] Weilkiens, T. Zukunftsdisziplin Modellbasiertes Systems Engineering. in: 8. Paderborner Workshop "Entwurf mechatronischer Systeme", HNI-Verlagsschriftreihe, vol. 294, Heinz Nixdorf Institut, Paderborn, Germany, 2011.

© Copyrighted material. A transfer agreement with Gesellschaft für Systems Engineering has been made for the German source of this translation. This translation is provided with kind permission of Gesellschaft für Systems Engineering. When using this material, always make reference to "Tag des Systems Engineering 2011" (see box in header).