System Architecture Validation with UML

André Pflüger, Wolfgang Golubski
Westsächsische Hochschule Zwickau
Fachgruppe Informatik
Dr.-Friedrichs-Ring 2A
08056 Zwickau
{Andre.Pflueger|Golubski}@fh-zwickau.de

Stefan Queins
SOPHIST GmbH
Vordere Cramergasse 13
90478 Nürnberg

Stefan.Queins@sophist.de

Abstract: The foundation of every system is the architecture. It has to be designed according to the corresponding system requirements by the system architect who is also responsible for ensuring that it fits to the system requirements even if these change due to new conditions during development process. Our approach defines a model driven process for the architect to validate system architecture against system requirements based on UML. It supports the architect in designing the architecture and in analysing the impacts of requirements changes.

1. Motivation

Designing and validating system architectures are time- and work-consuming tasks. The responsible person for those tasks, the system architect, must have special expertise in this area to create the basic foundation of the system, experience and social skills are also appreciated. The number of requirements is growing with the systems complexity which also increases the number of dependencies between them. The system architect does not only have to look at all these requirements, he also has to handle requirements changes and their impacts on the system architecture. Due to customer's desire for high flexibility, requirement changes are no longer limited to analysis phase and early design phase. They occur up to the end of implementation phase making architect's work a big challenge. A common proceeding can be described as follows: The architect creates an architecture draft and improves it iteratively. The validation is done manually by stepping through the architecture-relevant requirements. There is no defined validation process which might cause unintentional omitting of important aspects. The validation is based on estimations due to the architect's experiences and only in special cases, mostly for critical requirements, it is based on verifiable results. According to the required work effort documentation and validation are only done completely for important, e.g. review appointments. We think that the process described is not sufficient for developing complex systems because there is no defined validation process to achieve repeatable results. Figure 1 shows the four levels of our approach to support the architect in his work. The top-level defines our model-based validation process which is independent from any specific modelling language. According to apply this process we choose the Unified modelling Language (UML) and develop UML-specific support tool approaches in levels two and three. The implementation of these tool approaches are part of the fourth level. The validation process is used in conjunction with the iterative design of the system architecture which can be partly automated. So-called validation targets sum up architecture-relevant requirements on an abstract level suitable for architecture design.

207

Figure 1: different levels of the approach

This level gives the architect a better view on the system architecture according to the architecture-relevant requirements and makes complex correlations visible. For every validation target the architect creates a process to check if the associated requirements are fulfilled by the system architecture. The architecture is valid if all validation targets are valid. Requirements can be associated to more than one validation target supporting the architect during impact analysis after requirement changes. This approach depends on a model-based system documentation used as data source for the validation target processes and for a seamless documentation of system architecture design. For every validation target the architect may create a specific notation allowing him to model all necessary information for the validation target process. Based on the choice of UML information not available is added by using the UML profile mechanism [UML10].

In section 2 we will discuss related work to our approach. Section 3 describes the validation process in detail and section 4 deals with the validation target specific notations using the UML profile mechanism and the separating of development and validation model. In section 5 we apply the introduced approach to an example from the area of embedded systems. Section 6 lists up benefits and our experiences before section 7 concludes the paper and overviews future work.

2. Related Work

According to our research there are no approaches defining a model driven process for the architect to validate system architectures against requirements which includes the support for the architect for designing the architecture and analysing the impacts of requirement changes. The project System Architecture Virtual Integration (SAVI) [FEI10] initiated by the Aerospace Vehicle Systems Institutes is based on three key concepts: a reference model, a model repository with model bus and a model based development process. Models using i.e. UML, AADL or SysML and mapped to the reference model can be compared with the content of a requirements repository. In theory a validation of system architectures against requirements would be possible but the overhead would be high because this is not the projects intention. Furthermore, the project only exists in the description of a feasibility study. There are modelling-language-specifications available for special topics like the System Modelling Language (SysML) [OMG10a], the Architecture Analysis and Design Language (AADL) [CMU11] or the Modelling and Analysis of Real-Time and Embedded Systems (MARTE) profile [OMG10b] which could be used but they are limited to a specific domain and they cannot be extended to add validation specific data.

208

The tool PREEvision of aquintos [Aqu11] supports a model based development and evaluation of electric and electronic architectures. The modelling language is based on a graphical notation using an own domain specific meta-model, architecture relevant information are provided by a data model. Our approach is focused on the whole system architecture including software components and uses an extension of the domain specific notation by using the profile mechanism of the UML.

3. Validation process

The first step towards a model driven validation of system architectures is defining a process for the architect. This process should be independent from any special system or modeling language. It describes the necessary steps initially building up the validation and how to react on requirement changes. First the validation targets are detected from the requirements, especially non-functional requirements. The architect creates a validation process for every validation target and a validation target specific notation based upon the domain notation to add validation specific information to the validation model. By running all validation processes, manual or automatic, the architecture is validated. In failure case the architecture has to be adjusted and revalidated. These modifications are transformed into development model in case of a successful validation. The validation process restarts after requirements change. [Pfl10] describes this process in detail.

4. Modeling language UML

According to the four levels presented in figure 1 our approach does not depend on any particular modeling language but for practical usage a choice is necessary. We decided for the UML because it is a widespread and accepted standard in research and industry. It provides two possibilities to extend its meta-model: meta-model extension and the profile mechanism [UML10]. By creating and using a profile composed of stereotypes, tagged values and constraints for every validation target the architect can add the relevant information to UML elements without much effort. An UML element can own several stereotypes from different profiles, i.e. from different validation targets, thus there is no need for modeling dependencies between different validation targets just because they add information to the same UML element easing profile reuse. The UML like other meta-model based languages supports the exchange of data between different models. For this purpose the specification defines the XML Metadata Interchange (XMI) format. We use the Eclipse Modeling Framework (EMF) [EMF10] for implementing tools supporting the system architect in creating the validation model by transferring selected UML elements from development model to validation model and in synchronizing the two models after system architecture modifications. More details about this topic can be found in [Pfl10].

5. Radar system example

A radar system receives echoes of electro-magnetic signals from an antenna, processes these data closed to real-time and provides tracks which are the desired objects the radar engineer wants to see on his radar display. The data processing in a radar system is almost sequential. It is just separated by data processing variants due to different types of signals which are in our example: sea targets, air targets and near range targets.

These signals are characterized by different electro-magnetic wave forms and processing algorithms allowing the radar engineer to detect objects in a short or far away distance with different resolutions. The system processes a continuous stream of data so that the computing performance for data of a specific point in time is influenced by processing of past and future signals. For this example we consider the following requirement:

- The system shall provide tracks for fed in test data after 320ms. (A1)
- The system shall provide plots for fed in test data after 280ms. (A2)
- The system shall be able to compute 3500 MBit/s data received from antenna. (A3)
- The system shall consume less than 220Watt/h. (A4)
- The temperature of each FPGA shall be less than 50°C. (A5)
- The temperature of each microprocessor shall be less than 70°C. (A6)

The system architecture describes software and hardware components, their connections and behaviour. For this example we focus on the components and their connections. Each software component has to have a dependency connection with stereotype *executedOn* to a hardware component indicating that it is executed on it. There are three hardware boards with two processing units, one microprocessor and one field programmable gate array (FPGA), each.

According to the validation process the architect detects validation targets from system requirements in the first step. A1 and A2 are assigned to validation-specific aspect processing time (VT1), A3 to communication infrastructure (VT2), A4 to energy consumption (VT3) and A5 and A6 to system temperature (VT4). For the validation target specific notations the stereotype software has the tagged value *mflops* specifying needed floating point operations for the software component and the stereotype *hardware* has the tagged value *performance* providing the performance of processing units in MFlop/s. If we assume linear energy consumption and linear temperature rising for processing units in combination with a basic value, these data can be calculated by using processing unit load deduced by VT1's examination process. We add the tagged values *energyConsumption* and *basicEnergyConsumption* to the stereotype *hardware* as validation target specific notation for VT3 and *temperature* and *basicTemperature* for VT4 (see figure 3). We add the stereotype *communication* with tagged value *bandWidth* to the association representing physical link between the board and the hardware component executing the first software component in the processing chain. After determine the validation targets, their corresponding examination processes and notations the architect creates the validation model illustrated in figure 2.

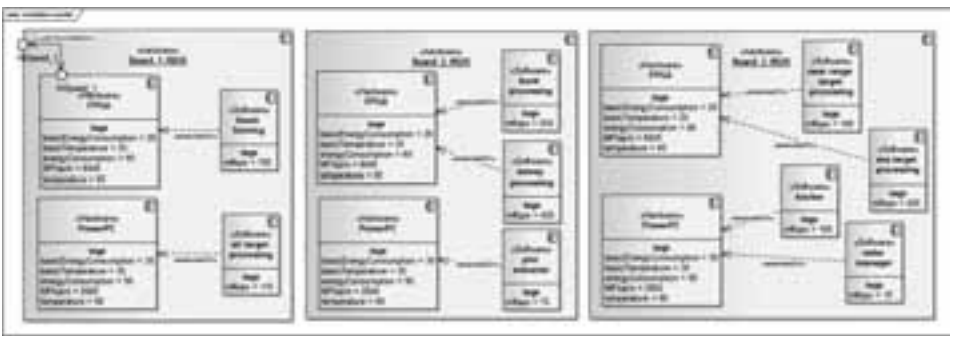

Figure 2: Validation model for radar system validation

210

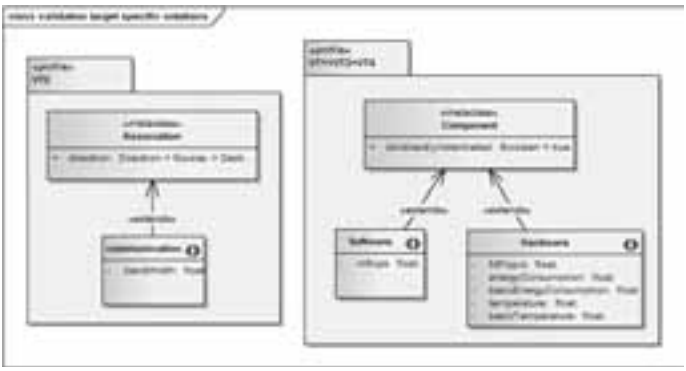

Figure 3: Validation target specific notations for VT1, VT2, VT3 and VT4

Table 1: result examination process VT1

Data	Processing time [ms]
tracks	313
plots	277

Table 2: result examination process VT2

Communication device	Band width [MBit/s]
Antenna – radar system	5200

Table 3: result examination process VT3 and VT4

Processing unit	Processor load [%]	Temperature [°C]	Energy [Watt/h]
B1_FPGA	26	48	35
B1_PowerPC	13	63	42
B2_FPGA	24	47	34
B2_PowerPC	8	60	37
B3_FPGA	14	43	28
B3_PowerPC	12	63	41
Total energy consumption			**218**

Table 4: result examination process VT1

Data	Processing time [ms]
tracks	306
plots	270

Table 5: result examination process VT2

Communication device	Band width [MBit/s]
Antenna – radar system	5200

Table 6: result examination process VT3 and VT4

Processing unit	Processor load [%]	Temperature [°C]	Energy [Watt/h]
B1_FPGA	25	47	34
B1_PowerPC	12	63	40
B2_FPGA	15	44	29
B2_PowerPC	28	74	55
B3_FPGA	13	43	27
B3_PowerPC	11	62	39
Total energy consumption			**221**

It contains validation specific information only and it is used as data source for automated validation. Some information of the development model has been transferred, some is not required and therefore has not been transferred and other has been added by the introduced profiles. The results of the system architecture validation, i.e. of the four validation targets, are shown in table 1 to 3.The system architecture passes the validation and therefore fulfils the regarded system requirements. There has been no architecture modification in the validation model so that synchronization of the models is not necessary. During development requirements are changing. In our example the customer changes requirement A1: Tracks shall be provided after 310ms instead of 320ms.

211

The revalidation after requirement change fails because VT1 cannot be validated: 313ms is greater than 310ms. The architect has to change the system architecture. He assigns software component *sweep processing* to the *PowerPC* of *Board_2*. This modification is memorized because it has to be transferred to the development model in case of a successful validation. Although VT1 is valid after modification, the whole validation fails due to energy consumption (see table 4 to 6). By analyzing validation results including former ones the architect is able to identify the problem. The processing unit load of the *PowerPC* on *Board_2* has been raised causing raising energy consumption which cannot be compensated by falling energy consumption of *FPGA* on *Board_2*. The architect restores the former architecture, exchanges the software components *burst processing* and *near range target processing* and reruns architecture validation. These modifications are transferred to development model because validations for all validation targets are successful.

6. Conclusion and future work

System development has to deal with evolving conditions challenging the system architect who has to design and document the architecture just as validate it against system requirements considering influencing parameter. The introduced approach defines a model-based validation process integrated into iterative development of system architectures. The process can be partly automated reducing time and effort for validation. The approach supports the architect in keeping track of architecture specific aspects and their dependencies such as analysing impacts after requirement changes. The approach has been developed with the help of projects of the SOPHIST GmbH and our experiences from system developments. Our experience shows that extra work caused by this validation process amortizes within several iterations of architecture design and requirement changes. Currently we are developing tools to support the architect in model-to-model transformation and using non-UML data sources during transfer process. In combination with the validation target management tool we try to provide a tool chain for the whole validation process increasing the usability. We also develop a validation target process simulation to gain experience with heavily dependent validation targets.

References

[Aqu11] PREEvision by aquintos, http://www.aquintos.com, last access on 2011-02-04

[CMU11] Architecture Analysis & Design Language (AADL), Carnegie Mellon University, http://www.aadl.info, last access on 2011-01-03

[EMF10] Eclipse Modeling Framework (EMF), Eclipse Foundation, http://www.eclipse.org/emf/, last access on 2010-11-28

[Fei10] P. Feiler, L. Wrage, J. Hansson, System Architecture Virtual Integration: A Case Study, Embedded Real-time Software and Systems Conference, May 2010

[OMG10a] System Modeling Language (SysML), Object Management Group, http://www.omgsysml.org/, last access on 2010-11-28

[OMG10b] UML-Profil MARTE, http://www.omgwiki.org/marte/, Object Management Group, last access on 2010-11-28

[Pfl10] Pflüger A., Golubski W., Queins S.: Validation Of System Architectures Against Requirements, International Joint Conferences on Computer, Information and System Sciences, Engineering (CISSE'2010), published by Springer in Summer 2011

[UML10] Unified Modeling Language (UML) Version 2.3, http://www.uml.org, 2010