Behavioral Modeling in System Engineering

RADEK SILHAVY, PETR SILHAVY, ZDENKA PROKOPOVA
Department of Computer and Communication Systems
Faculty of Applied Informatics
Tomas Bata University in Zlin
nám. T. G. Masaryka 5555, 760 01 Zlín
Czech Republic
rsilhavy@fai.utb.cz, psilhavy@fai.utb.cz, prokopova@fai.utb.cz http://fai.utb.cz

Abstract: - This contribution focuses on the behavioral modeling of the systems in the system engineering. The system engineering process is described and overview of the system modeling language (SysML) is presented. The SysML basic principles and diagrams are discussed. Practical part of the paper discusses behavioral modeling process, which is illustrated by the set of the example models.

Key-Words: - System engineering, system modeling, behavioral modeling, sysml, uml.

1 Introduction

The System engineering [1] is understood as complex discipline for the system design and analysis of the system. The system is for the purpose of the system engineering defined as a set of the components which are interconnected and provides the group of emergent properties [1]. These emergent properties are derivate from the properties of the system components, but new dimension is added by the system integration.

A system engineering processes are about preparing [2] generic stakeholder goals, requirements, system design, evaluation of alternative system designs, allocation of functional requirements, system verification. All of these activities lead to creation of the balanced system.

The system development generic process is described by waterfall model [1]. The process should be composed of:

1) Requirements Engineering Definition and Elicitation.
2) System Design.
3) Sub-system Development.
4) System Integration.
5) System Installation.
6) System Evolution.
7) System Decommissioning.

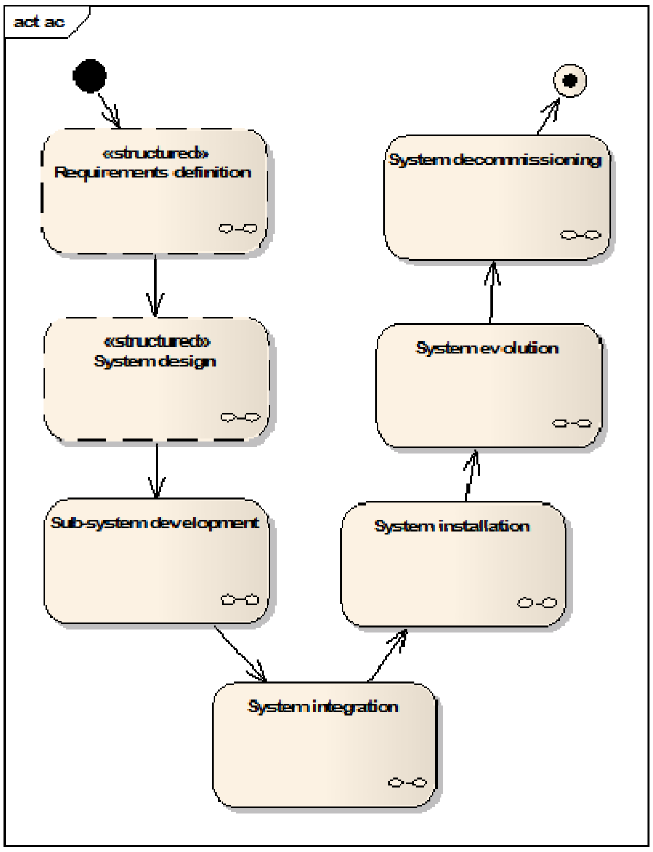


Fig 1: Waterfall Model of the System Engineering

The system requirements engineering general name for the specific sets of the software engineering techniques, which is used at the beginning of the software cycle. The purpose of these techniques is to discover stakeholder's needs.

The requirements gathering process in system engineering has these basics steps, which are graphically described on the next figure.

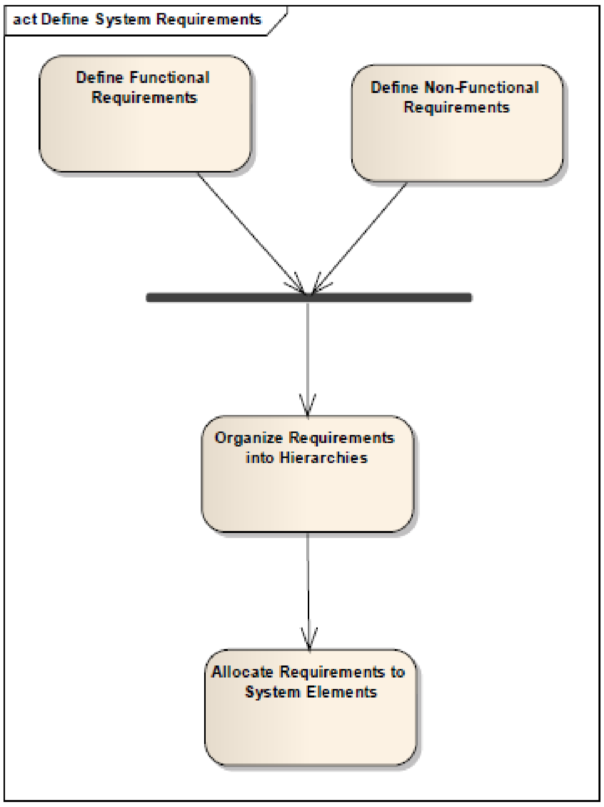

act Define System Requirements
Define Functional Requirements
Define Non-Functional Requirements
Organize Requirements into Hierarchies
Allocate Requirements to System Elements

Fig 2: Requirements Definition Model

The requirement [1] is a description of the functionality or condition which stakeholders define for the system. After the first round of requirements gathering is talked about the raw or abstract requirements. These raw requirements are list of functionality or condition for the proposed software, which is unanalyzed yet. The most important in this phase is to establish the project goals, which should be achieved.

The next group of the requirements is non-functional requirements. The purpose of these requirements is familiarized system designers with problem domain and conditions in the domain. Well-known examples of these are reliability, performance, safety or security. These non-functional requirements are critical in the system evaluation very often.

The last part of the requirements gathering is so-called system characteristics. The system characteristics are commonly prepared in negative way. It means, that system design specifies what irrelevant system behavior is.

The system design phase deals with association of the system requirements to individual sub-system, more specific to system components. The set of the system requirements is studied and individual requirement is associated to proposed component. One of the most common techniques, which could be applied in this process, is to sub-system design first. Than system designers is able to associated selected requirements to the specific subsystem. The process of the sub-systems identification and requirements association is interactive and is commonly modeled in form of spiral.

In the system modeling phase is a system architecture designed. This activity is based on sub-system (or component) design. The system is modeled as group of interconnected blocks, where connection indicates data flow or other form of dependency. The main task is to create more concrete definition of the sub-systems, which were set-up in the system design phase.

The sub-system development phase works with sub-system or system components implementation. The sub-systems are implemented in parallel. This is because of sub-systems in the system engineering is not only hardware/software, but also should be create by civil engineering.

Next step in the system engineering life-cycle is the system integration. The implemented subsystem are integrated into the system [1]. The integration is an incremental process. The sub-system are integrated, when their implementation is finished. This approach is time-less consuming, then legacy approach, when all components were implemented together.

The system evolution and system decommission are the final phase of the system lifecycle. A system designer should care about system improvements during its production time. They also should take care about times, when system should be prepared for out-of-service elaboration.

The main task of the software engineering is a system development from very beginning (requirements elicitation) to system decommission.

2 Problem Formulation

In the system engineering lifecycle, which were described above, basic ideas of the system engineering were concluded. In the system engineering are used well-known techniques for modeling. Probably the most common is block diagramming. This model is used for system design and for sub-system analyze. Secondly data flow diagrams are common for definition of data and process designing.

Today state-of-art in research of graphical documentation of the system development is System Modeling Language (SysML). The SysML is a graphical modeling language [2], which is derivate from Unified Modeling Language (UML). UML is an industry standard in the scope of the software engineering. SysML is a extension of UML, this two basic technique shared basic principles and some types of diagrams are used in both. The SysML take important role in the system engineering, because its usability in all phase of software engineering process.

3 System Modeling Language

SysML is relatively young modeling language. Its history is written from 2001, when Systems Engineering Domain Special Interest Group were set-up [3]. Today, there is 1.2 version from 2010 valid.

3.1 Language Overview

The modeling project support analysis, specification, design, verification and validation of systems. SysML therefore support all phases of the system engineering lifecycle.

The system components should be described by structural composition, interconnection and classification. Secondly by function-based approach which is based on messages between objects. This aspect should respect to constrains, which are derived from physical system structure or from performance properties. Important role takes association of system functionality to each of the system components.

In the SysML nine of diagrams are of recognized [2].

Requirements diagram is used for graphical interpretation of requirements and their connection to other requirements and to other elements and entities in project – such test cases, use cases. For those how are familiars to UML, this diagram is new in SysML and have never be used in UML.

Activity diagram is based on UML activity diagram; there is slightly different usage of it. Basically is used for modeling of flows of actions based on the availability of inputs, outputs or control. Transformation of actions is also modeled by activity diagram in SysML.

Sequence diagram is used for representing of message flow between objects.

State machine diagram presents set of state of the modeled entity and events which generated message upon which entity state is switched.

Use Case diagram is used for the system function description. This model is composed of the actors, which are external entity and of the use cases. The use cases represent system functions or algorithms. Each of the use case has to realize one of the requirements as minimum.

Block definition diagram is used instead of the class model in UML. The purpose of the block definition is to model system structure.

Internal block diagram is similar to UML composite structure. The main idea of this diagram is to model internal structure of the each individual part of the proposed block. Very important here is modeling of interface and communication between block´s entities.

Parametric diagram is SysML origin and have no predecessor in UML principles. Is used for modeling system parameters and constrains, should be used for critical – hazard system states.

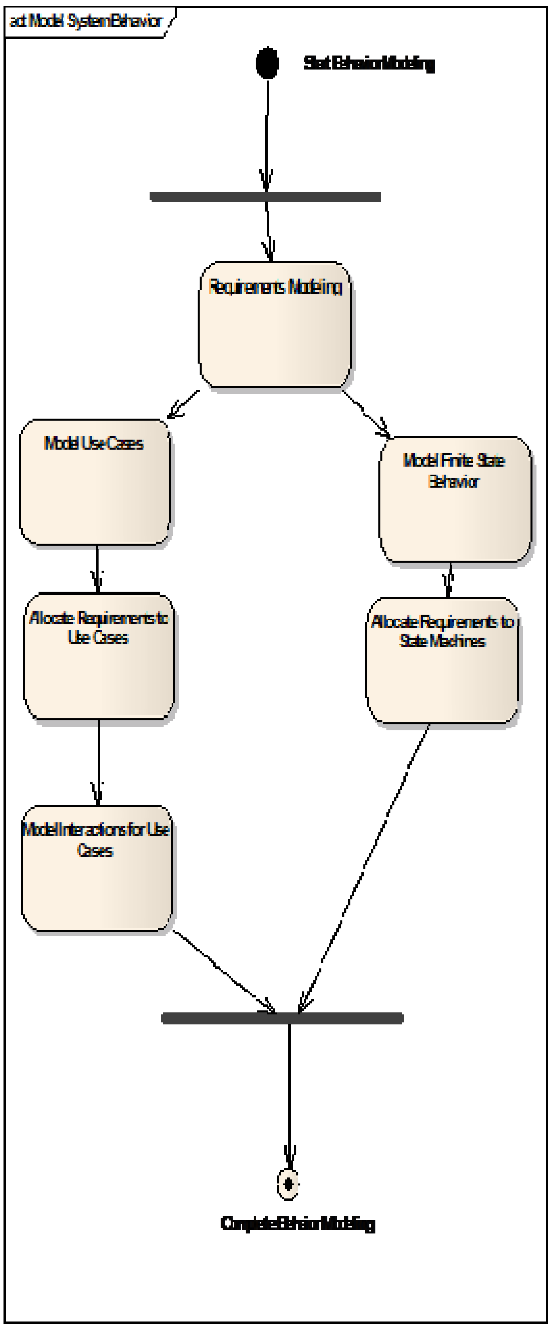
Package diagram is useful for model organizing. Model elements should clustered by its stereotypes. Packages also should be used for creation of a large project structure.

4 Behavioral Modeling Process

In this chapter will be introduced sample projects, which contains example model of each type of the diagrams. For purpose of this article authors adopted project which was prepared by SparxSystems as sample project in SysML language [4].

In this sample is described development of the audioPlayer. The main task of the project team was to offer a solution which was successfully in usability and which offers appropriate functionality.

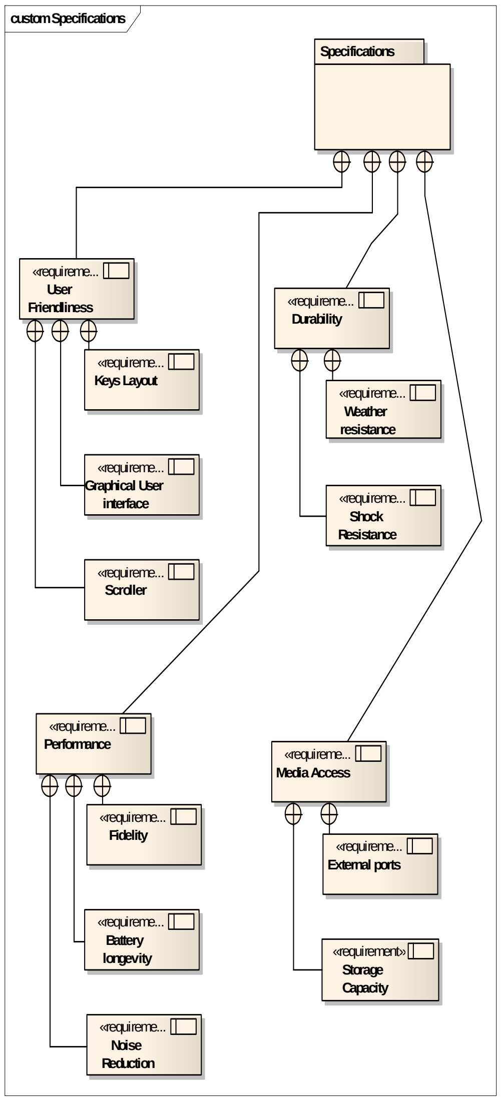
For purpose of this description authors used modified version of behavioral modeling process [4].


act Model System Behavior
Start Behavior Modeling
Requirements Modeling
Model Use Cases
Model Finite State Behavior
Allocate Requirements to Use Cases
Allocate Requirements to State Machines
Model Interactions for Use Cases
Complete Behavior Modeling
Fig 3: Behavioral Modeling Process

The behavioral modeling is a description (Figure 3) how the proposed system will interact with the actors and with entitles which is out of boundary of the system.

The first step in the creation of the system behavioral model is the requirements gathering.

In the figure can be seen model of the requirements. The diagram illustrates hierarchical structure of the requirements. On the top of the tree the Specification package can be seen. Other parts are interconnected by containment association. The specification is clustered into sever groups. Groups are User Friendliness, Durability, Performance and Media Capacity.


custom Specifications
Specifications
«requireme... User Friendliness
«requireme... Keys Layout
«requireme... Graphical User interface
«requireme... Scroller
«requireme... Durability
«requireme... Weather resistance
«requireme... Shock Resistance
«requireme... Performance
«requireme... Fidelity
«requireme... Battery longevity
«requireme... Noise Reduction
«requireme... Media Access
«requireme... External ports
«requirement» Storage Capacity
Fig 4: Requirements Model

The user friendliness group defines set of the requirements, which deal with quality of service of the audio player. Keys Layout, Graphical User Interface and Scroller are primary requirements, which take important role in user satisfaction.

When the requirements model is finished, next step is use case model. In the use case model there is an algorithmic definition of the actors' activity in the system. In the figure system boundary, actor and use cases can be seen. The boundary is named as Playlist Maintenance, because this model describes only activities of the listener, which are available as maintain the playlist. Inside of the each individual use case is scenario description. The scenario is a sequence of steps, which describes a track, should be downloaded. In this Use Case model include association is used. The "include" association is used for situation, when the use case contains other use case for achieving its goal.

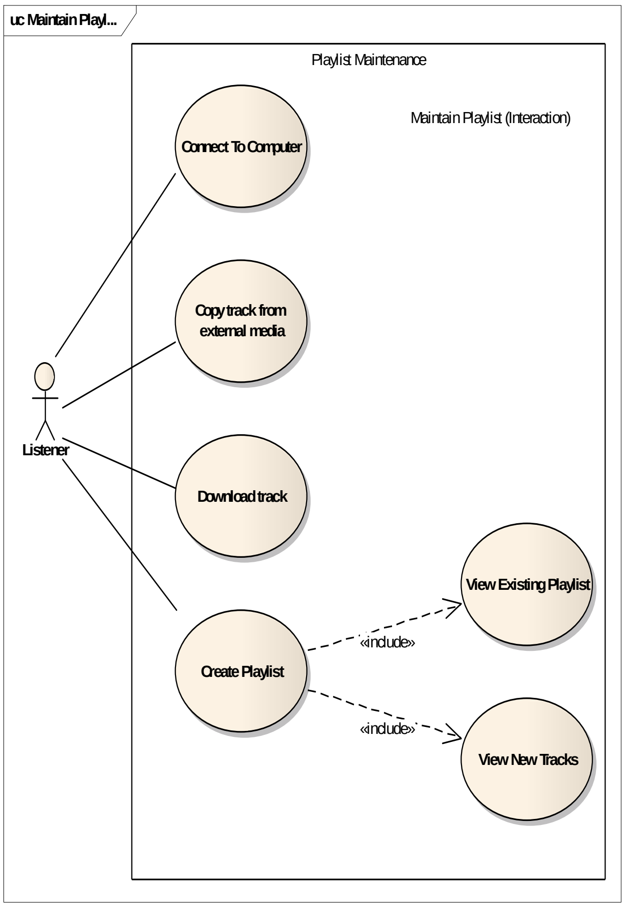


Fig 5: Use Case Model

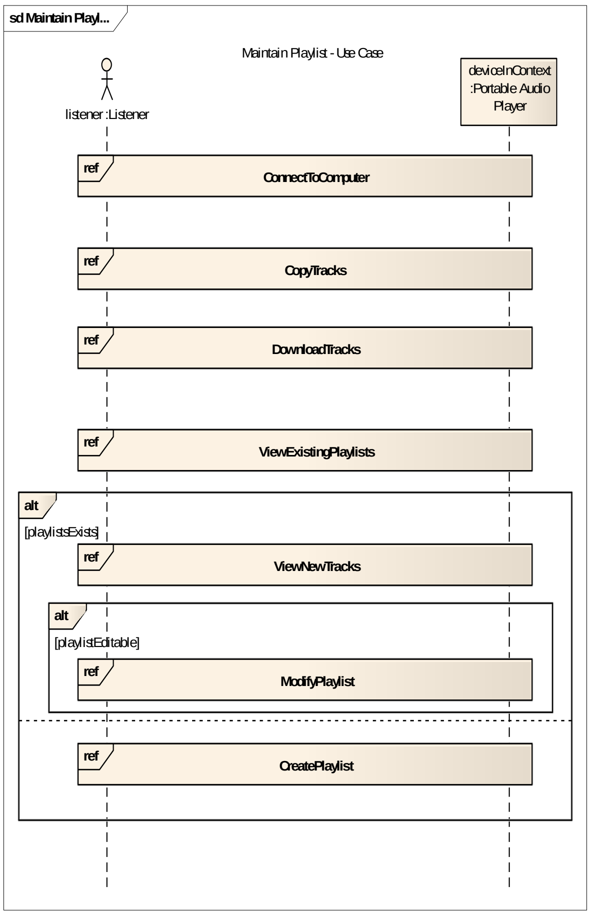


Fig 6: Sequence Diagram for Playlist Maintaining

The Use Case models (Figure. 5) are derivate from requirements. The Use Cases are commonly prepared for individual system blocks. This is rigorous connection between the requirement and the use case. This connection should be modeled as realization in the use case model or can be documented in form of the responsibility matrix.

The interaction in form of sequence diagram is useful for modeling overview of operations. In the Fig 7 can be seen overview of diagram, which describes all possible activity of the actor, called Listener. There are used fragments, which used for referencing individual activity descriptions. These are named as ref. The alt abbreviation is used for conditions. In this context the viewnewtracks can be executed only if the playlist exists.

The last view of the system can be done by state machine diagram. The most important noticeable fact here is, that by state machine diagram is modeled same system in the different view only.

The Figure 8 shows modified version of the state machine diagram. It does not present only two basic states – idle and connected. The state connected is describes in form of the activity diagram. In the connected state tracks could be copied and downloaded. These two operations are in a parallel section, which is created by using fork/join artifacts.

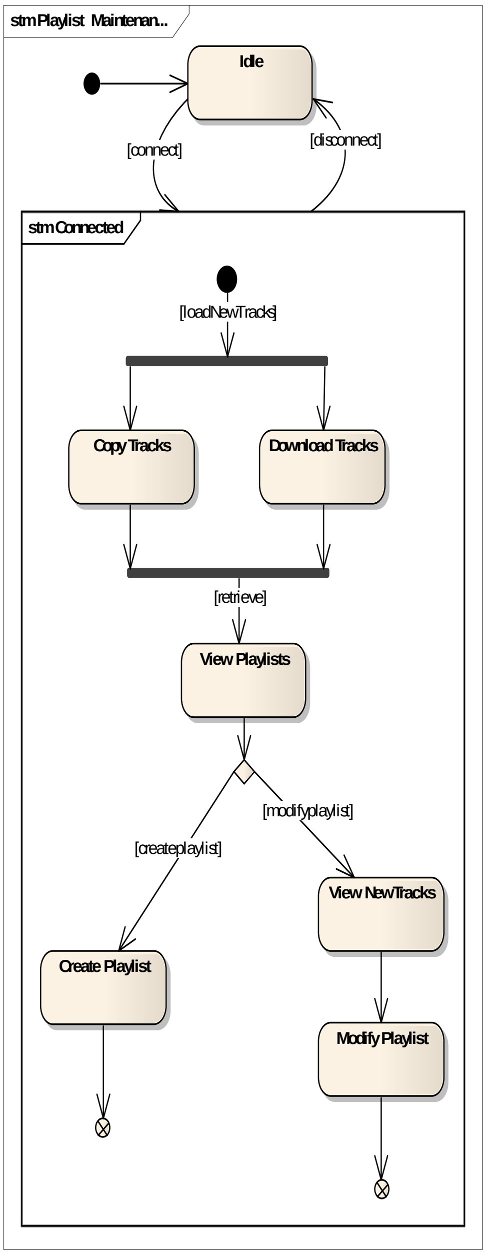


Fig 7: State Machine Diagram with Activity

4 Conclusion

The idea of the contribution was to introduce System Modeling Language for modeling system behavior. The system engineering were described and connection between system modeling language diagrams and the system engineering phases were illustrated.

For the example purpose part of the the audio player model were presented.

The modeling project support analysis, specification, design, verification and validation of systems. SysML therefore support all phases of the system engineering lifecycle.

The SysML uses number of diagram, which allow to a system designer model the proposed system in many views.

The system behavioral modeling deals with requirements engineering, use cases elicitation and state modeling of the system.

Further research in system modeling is focused on the improvement of the simulation environment and to the model based development, which probably the most important in the development of the system engineering.

Acknowledgments
This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic under the Research Plan No. MSM 7088352102.

References:

[1] SOMMERVILLE, Ian. Software Engineering. Eight Edition. Harlow : Pearson Education Limited, 2007. 824 s. ISBN 978-0-321-31379-9.
[2] FRIENDENTHAL, Sanford; MOORE, Alan; STEINER, Rick. A Practical Guide to SysML : The Systems Modeling Language. USA : Morgan Kaufmann, 2009. 5769 s. ISBN 978-0123786074.
[3] Object Management Group. Systems Engineering Domain Special Interest Group [online]. 2007-2011 [cit. 2011-03-20]. Available on WWW: <http://syseng.omg.org/>.
[4] ROSENBERG, Doug; MANCARELLA, Sam. Embedded Systems Development using SysML. [s.l.] : Sparx Systems Ply Ltd, ICONIYX, 2010. 68 s.