arKItect

Reading part report of MDE project

By

Mohammmad Amin Sepasian

Model-Driven Engineering

Held by

Prof. Hans Van Gheluwe

Antwerp University

Computer Science Department

Table of content

Table of figures

1. Introduction

The under study tool, **arKItect**, is a commercial DSM tool to design large scale complex systems potentially interact with physical world. **arKItect** provides the possibility to create meta-model and graphical representation of desired DSM systems origin from different natures. Furthermore, the graphical representation is enabled to interact with other domains such as software domain, including *block diagram, data flow, state diagram* and etc, resulting in changes in one of interacting domain upon a change occurs in the other interacting domain. Having different variants or configurations of a same system as well as being able to generating documents and reports are other features of **arKItect**.

1.2. arKItect Designer

The most useful part of arKItect tool for the goal of this project is *arKItect Designer* in which it is possible to define a Domain specific model or a meta-model of a specific criterion. It is worthy to mention that the tool is a DSM tool which mainly focuses on the domains produce, consume and flows of data or processes. Moreover, its main use of arKItect is for system architecture and system requirement designing. Therefore, it is challenge to match such tool for different domain such as a game. Meta-Modeling in architect exploits particular notions of itself which will be explained in the following part.

2. Meta-modeling - arKItect concepts

In arKItect Designer, users, or better say designers, define the types, the flows, the views and the attributes. arKItect tool calls this the meta-model. The main features to define a meta-model are Rules and Filters. Along with these notions, there are other notions such types, attribute, relation and requirement which more and less close what it is been using in other modeling criteria, with some face lift to fit to the context.

2.1. Rules

“Rules define the object types as well as their relations. A Rule links two types together in a type relation. The term "type" denotes the abstract type of an object whereas a "relation" represents the nature of a parental connection between two types” [1]. In arKItect *types* could be seen as class of objects or the entities of the system categorize other entities. In figure 1, the panel of Rules where the system model and its contained entities, such as “Stakeholder” with a branch of attribute and requirement, is demonstrated consider

Remark. The terms rule and type are often used interchangeably and may be mistaken one to another. For example, "defining a type", in fact, is equal to create a new rule for a type and similarly "deleting a type" is deleting a Rule. Although, “defining a new rule does not always involve the creation of a new type (while the opposite is true) and the removal of a rule does not always lead to the complete removal of a type” [1].

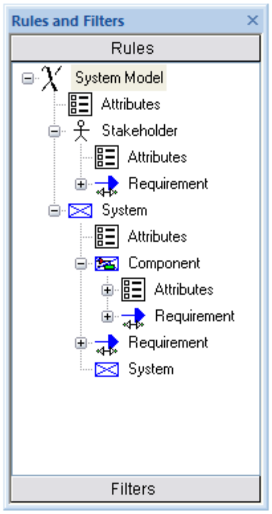

Figure 1. Rules panel

2.2. Filters

The next principle of arKItect meta-modeling is Filters which in each project is necessary to create at least one Filter. And in fact, it is possible to have number of filter over a project. “Projections are actually the result of filters applied to the rules of the project; they permit to show only chosen aspects of the project by creating specific views on it. For instance, in a system engineering project your projections could include System Architecture, Functional Architecture and Physical Architecture as well as Use Cases”. In a nutshell, tt allows stressing in specific parts of a project.

Filters are actually applied to rules and not to types. Consequently, the possibility to project a type in a given context while excluding other contexts: “for instance, it is possible to display all Cars that are children of the Transport System but to exclude the ones that are children of Consumer Goods” [1]. In figure 2, the filter panel is depicted, where the system model consists of two views *Requirement Statement* and *System Architecture*, while it could be a choice to have extra views for *Control System Architecture* or *Functional Architecture* by another filter.

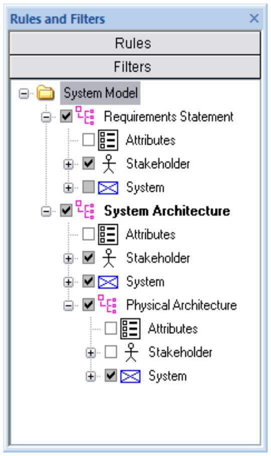

Figure 2. Filters panel

2.3. Other concepts

Relations. There two kinds of relation in arKItect, *hierarchal relation* and *flow relation,* which are divided to three different relation variants:

- An element relation, the child type being a simple component of its parent.
- A generic flow, either an input or an output or both. However, flow object instances must be defined as either input or output (they cannot be both except in the case of bidirectional flows).
- A bidirectional flow, both an input and an output

Types and Attributes. These two notions tend to mimic notions class and class attributes is present in some famous modeling languages like UML and of course their semantic is the same as their correspondent in UML

Requirements. It is obvious that each system has its own specific requirement, so this notion does not come as a surprise and would be a good choice to constraint the system.

3. Discussion and features

After having the overview of arKItect, now it is time to evaluate the power this tool to see how it could fit to the goal of the project.

As it is discussed already, the tool is specifically emphasis on flowing systems where relations are mostly dependant on flow of data or processes such a production line of a factory. Therefore, in the first contact to the tool, it is seen that the context is somehow irrelevant to the goal of Project, namely RPG formalism which is a game framework. However, this is not a deadlock to say no to project question. The idea is that to look at the RPG in same viewpoint of the architect, then good things come up. In other words, the challenge is to treat the problem of RPG in a sense that is more compatible with the arKItect context.

Besides this, a good choice would be to utilize the other features of the in such ways like exporting semantic data of the system to other platform, where applicable. It is remarkable that architect claims to provide such features for *Doors* requirement interface (Dynamic Object Oriented Requirements System) tool, *Simulink* or *Excel*.

4. Project Plan

As it is mentioned before, the main goal of this project is to define DSM framework for the RPG formalism using the under study tool. Secondly, the effort is to exploit as much power as the tool to investigate all the features and possible features. However, as discussed above, due to different context, the challenge of the project is to fit RPG context to the arKItect context. And definitely doing analysis and draw comparison to what it is done already bt AToMPM tools is one of super important concern of this project.

References

[1] http://support.k-inside.com/display/ARKI23/arKItect+2.3.x+documentation+home

[2] http://www.k-inside.com/web/