Modeling Comparison Of WebGME With AtomPM

Henry Tirla[1]

Belgium, Antwerp

University of Antwerp[1,1]

Abstract

Webgme is a web and cloud based multi-paradigm modeling collaborative tool that support the creation of domain specific languages and their corresponding domain models .Using this tool unique prototypical inheritance hierarchy scheme I will attempt to design a domain specific language for an evacuation system. Merits and demerits of this tool will be discussed , additionally with its comparison to another model driven engineering software AtomPM.

Keywords: Evacuation System, Domain Specific Language, Webgme

1. Introduction

Webgme is a web-based, collaborative meta-modeling environment tool with a centralized version controlled system. Hierarchical decomposition is the basic structure of models in this tool.It's unique prototypical inheritance feature makes each model a prototype that can be derived to create instances of other sub models. This variant of inheritance is a very powerful way to handle the inherent complexity in large models and complex domain specific model languages. I will be discussing subsequently the Bmod evacuation system requirements and how I used webgme to model it

System Requirements. The evacuation system involves the following entities Floor ,Person , Door , Cell , Dangerous condition and Emergency Exit Sign . The system requirements are as follows;

- Rooms and Doors should be represented and their connection clearly define
- Each Person Should have a perception and action attributes and their behaviour in case of an emergency is defined based on these attributes
- Cell should be connected to each other
- Concrete and Abstract Syntax of System should be defined
- Only one dangerous condition should be model in our case this will be fire.
- Emergency sign is to be model as fluorescent arrows indicating direction to exit.
- Containment relationship between cell to person and cell to fire should be define
- Movement of People should be modeled
- Transformation rules should be define
- Transform model into a petrinet model

In this paper subsequent sections I will discuss my approach towards the implementation of these requirements as follows ; section two will describe my techincal experience while developing the domain specific language , section three important features of Webgme ,section four will discuss about webgme in comparison to AtomPM and I will conclude with my recommendations in regards to both tools.

2. Domain-specific language modeling

Abstract Syntax. Webgme does not support the classic association between entities of a system , what it does is support containment relationship between them with a default cardinality of zero to many.I design an abstract syntax of the evacuation system based on this relationships. This form of association can also be used as a form of constraint while instantiating child nodes within an entity. For example in my abstract syntax i design there can be only one instance of a dangerous condition within a room and subsequently only one instance of fire can be instantiated. All entities of this system are objects of the floor entity define in our language model.

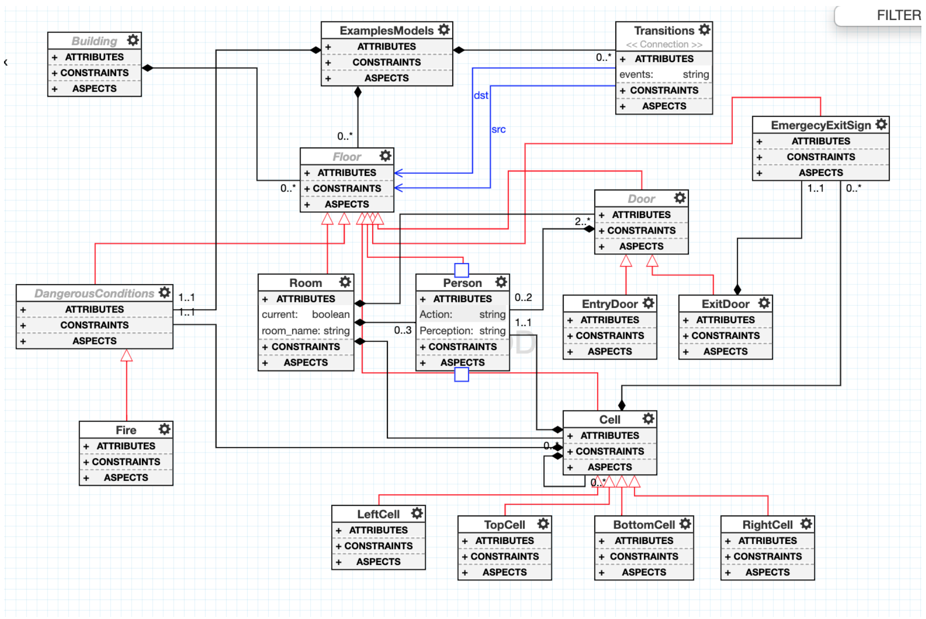

Figure 1: Abstract Syntax

Concrete Syntax. Webgme has a decorator tool which allows specific models to be given a form of concrete syntax ,its main limitation is not allowing the end user to upload his chosen graphical symbols into webgme, also it is not as graphical as AtompM and most importantly expressive. Though this can me defined for each node object it wouldn't give much meaning to the evacuation system . It lack of expressiveness make it irrelevant trying to define a concrete syntax , persisting to do this with webgme can lead to an overly complex visual model with no concrete meaning.

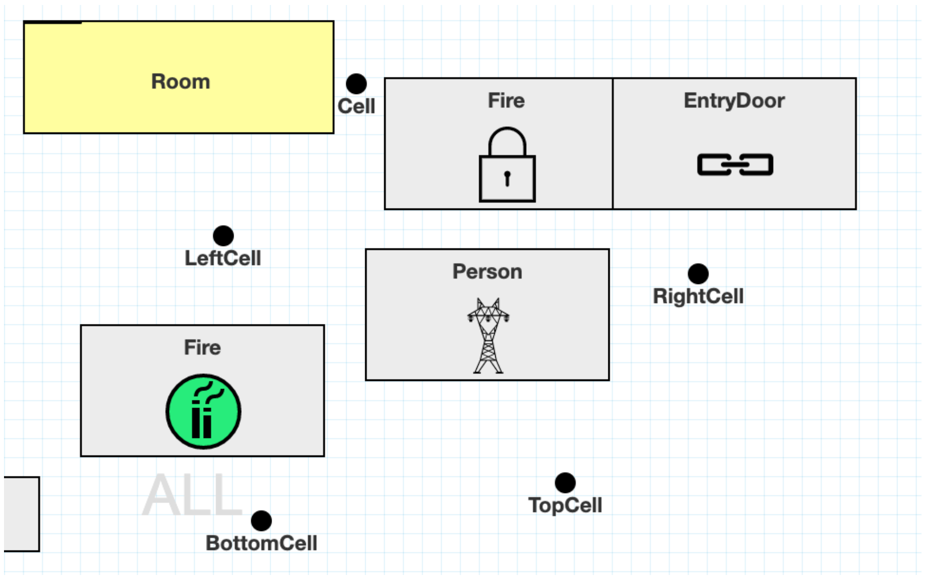

Figure 2: Concrete Syntax

Operational Semantics. Webgme does not support this .Though it can be argued the language for rules definition can be defined .This will have no impact on the behavior of the model since at its core webgme does not support graphical complex model simulations. In an attempt to describe the process of an evacuation , i included transitions in my model which can be use to describe connection between node objects ,and i realise even with these it's not expressive enough because an evacuation system cannot be abstracted to a simple process it relinquish its meaning and practicality. Also in comparison to AtomPM base on requirements from assignment four where model translation into petrinet was required to be automatically generated base on certain predefined transformation rules. In webgme my model can be translated into a petrinet without any transformation predefined rules but simply by re-using the petrinet model built within Webgme , by considering each cell to be a place ,fire and person to be tokens and perception or action as inhibitor arcs, still yet even if i give my domain this semantics, analysis on complex movement of people will not be satisfied because no dynamics rules are being defined.

Code Generation and Simulation. Webgme have a plugin frameworks that build up the model going through it's hierarchy and extracting the data needed for generating code which can be use to simulate the dynamic behaviour of a model. Webgme being mostly web-base there is no functionality to generate a custom plugin online hence the user have to do it within a local environment and the method of integrating it within the web-base workflow again require a high technical knowledge of the tool.Also there are available plugins created for default models already build within webgme but are only project specific and in our case are not much relevant.

3. Important Features of Webgme

The important features of webgme over AtomPM;

- Version control: It's in built version control system make it a unique model driven engineering collaborative tool
- Web Based : Being platform independent enhance it's collaborative ability.
- Documentation: In built documentation system which provide the documentation of a system within the system making it more knowledgeable to new users who are to work within a project.
- Dynamic: Changes done to a model are saved automatically in webgme.

4. WebGME Vs AtomPM

When it comes to design of graphical dynamic systems for an evacuation. Webgme isn't the most suitable tool firstly because the dynamics of the model cannot be implemented also predefined icons and decorators do not make it a much expressive tool hence when it comes to graphically representing a model AtomPM is a more expressive tool. The aim of an evacuation system is to be able to evaluate the dynamics of the system, providing data which can be implemented when building the system , having this mind it is very clear Webgme isn't the most suitable tool to evaluate that aspect of the system.The learning curve in terms of code(plugin) generation is high compared to AtomPM. Also Webgme has many online tutorials to help a user get started in comparison to AtomPM. In my opinion both tools are suitable for the design of different types of systems in this case webgme is not a suitable to design the concrete syntax and operational semantics of the Bmod evacuation system. Additionally webgme needs a variety of components to be install locally making it quite complex but i think its web base version resolves this and unlike webgme AtomPM is relatively easy to install and work with locally.Finally in terms of graphical user interface , WebGME is a more user friendly tool in comparison to AtomPM.

5. Conclusion

Webgme being an extension of GME it is a tool still being developed and continually improved .It is already a great collaborative tool which can design very complex systems. Giving it the ability for a user to upload his icons will make it a much more expressive tool.Also implementing a web-base and version control system within AtomPM and additionally improving it's graphical user interface will make it a much more better model driven engineering tool.

References

[1] M. Maroti, T. Kecskes, R. Kereskenyi, B. Broll, P. Volgyesi, L. Juracz, T. Levendoszky, A. Ledeczi "Next Generation (Meta)Modeling: Web- and Cloud-based Collaborative Tool Infrastructure". Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA.

[2] Tamas Kecskes Qishen Zhang Janos Sztipanovits "Bridging Engineering and Formal Modeling: WebGME and Formula Integration" Department of EECS, Vanderbilt University, Nashville, TN.

[3] Patrik Meijer, Anastasia Mavridou, "How to Build a Design Studio with WebGME," Institute for Software-Integrated Systems Technical Report

[4] WebGme Youtube Tutorials
`https://www.youtube.com/channel/UC1cPQP4jjsXRhpXUnoPZQWg/playlists`