Detecting Model Inconsistencies in 4diac Models with OCL

STUDENT: Sándor Bácsi

MENTOR: Alois Zoitl

1. Abstract

Eclipse 4diac[1] is an open source environment for programming distributed industrial automation solutions and control systems based on the IEC 61499[2] standard. One of the components of Eclipse 4diac is the 4diac IDE which is an integrated development environment for modeling distributed control applications compliant to the IEC 61499 standard. The current version of 4diac IDE does not support the validation of the models, which makes it difficult for users to get design-time feedback on inconsistencies regarding the models developed in 4diac. The Object Constraint Language (OCL)[3] could be a solution to find issues in 4diac models, since it provides capabilities for specifying generic constraints a model has to fulfill. The aim of this project is to develop OCL constraints and well-formedness rules to the metamodels of 4diac in order to improve the usability of the IDE.

2. Detailed Information

In this section, I introduce the technologies and existing Eclipse projects that may provide a background to my work. I discuss how the presented technologies are related to this project.

The 4diac IDE is written in Java[4]. Obviously, I have to use Java to implement the new features in this project. At the same time 4diac is strongly based on the Eclipse platform using several of its technologies (e.g., EMF[5], GEF[6], Xtext[7]). The Eclipse Modeling Framework (EMF) will play an important role in this project, since the abstract syntax of the 4diac modeling language is stored in Ecore models. I have to make many changes in the 4 diac Ecore models in order to adopt them to the needs of my GSoC project.

---

[1] "Eclipse 4diac." https://www.eclipse.org/4diac/. Accessed 28 Mar. 2020.
[2] "IEC 61499 - Wikipedia." https://en.wikipedia.org/wiki/IEC_61499. Accessed 28 Mar. 2020.
[3] "Object Constraint Language - Object Management Group." https://www.omg.org/spec/OCL/2.4/PDF. Accessed 28 Mar. 2020.
[4] "Java (programming language) - Wikipedia." https://en.wikipedia.org/wiki/Java_(programming_language). Accessed 28 Mar. 2020.
[5] "Eclipse Modeling Framework (EMF)." https://www.eclipse.org/modeling/emf/. Accessed 28 Mar. 2020.
[6] "GEF - Eclipse." https://www.eclipse.org/gef/. Accessed 28 Mar. 2020.
[7] "Xtext - Language Engineering Made Easy! - Eclipse." https://www.eclipse.org/Xtext/. Accessed 28 Mar. 2020.

Regarding OCL, the obvious best choice is the OCL infrastructure[8] provided by Eclipse, which provides many capabilities to support OCL integration. One of the most useful features of Eclipse OCL is the OCLinEcore Editor[9] which is convenient for developing and maintaining OCL constraints in Ecore models.

3. Experience

As for my previous experience, I have done several projects in the field of software modeling. During my master's studies I worked on smaller projects using Ecore, Xtext and Sirius[10]. As a PhD student, my research interests include domain-specific modeling and multi-level modeling. Currently, I work in our department's research group where I am developing our multi-layer modeling framework, the Dynamic Multi-Layer Algebra[11].

Publication list:

Sándor Bácsi, Gergely Mezei: Towards a Visualization of Multi-level Metamodeling Techniques. ICSOFT 2018: 389-396

Sándor Bácsi, Gergely Mezei. (2019). Towards a Classification to Facilitate the Design of Domain-Specific Visual Languages. Acta Cybernetica. 24. 5-16. 10.14232/actacyb.24.1.2019.2.

Zoltán Theisz, **Sándor Bácsi**, Gergely Mezei, Ferenc A. Somogyi, Dániel Palatinszky: By Multi-layer to Multi-level Modeling. MODELS Companion 2019: 134-141

Ferenc A. Somogyi, Gergely Mezei, Dániel Urbán, Zoltán Theisz, **Sándor Bácsi**, Dániel Palatinszky: Multi-level Modeling with DMLA - A Contribution to the MULTI Process Challenge. MODELS Companion 2019: 119-127

Gergely Mezei, Zoltán Theisz, **Sándor Bácsi**, Ferenc A. Somogyi, Dániel Palatinszky: Towards Flexible, Rigorous Refinement in Metamodeling. MODELS Companion 2019: 455-459

Gergely Mezei, Zoltán Theisz, Dániel Urbán, **Sándor Bácsi**: The bicycle challenge in DMLA, where validation means correct modeling. MODELS Workshops 2018: 643-652

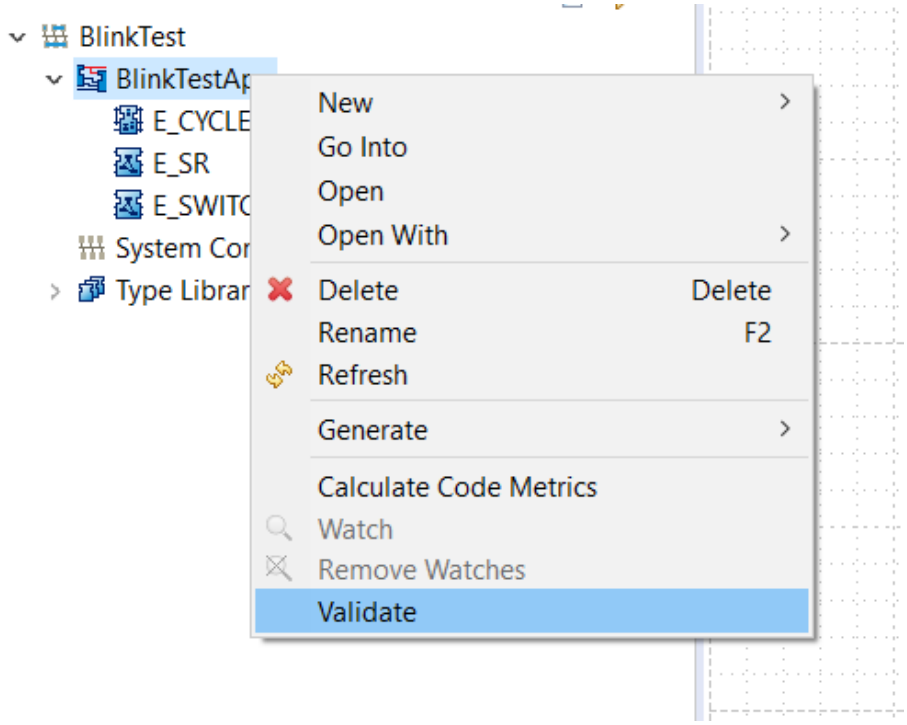
---

[8] "Eclipse OCL (Object Constraint Language) | projects.eclipse.org." https://projects.eclipse.org/projects/modeling.mdt.ocl. Accessed 28 Mar. 2020.
[9] "OCL/OCLinEcore - Eclipsepedia - Eclipse Wiki." 31 May. 2013, https://wiki.eclipse.org/OCL/OCLinEcore. Accessed 28 Mar. 2020.
[10] "Sirius - The easiest way to get your own ...." https://www.eclipse.org/sirius/. Accessed 28 Mar. 2020.
[11] "The bicycle challenge in DMLA, where validation means ...." http://ceur-ws.org/Vol-2245/multi_paper_2.pdf. Accessed 29 Mar. 2020.

4. Background

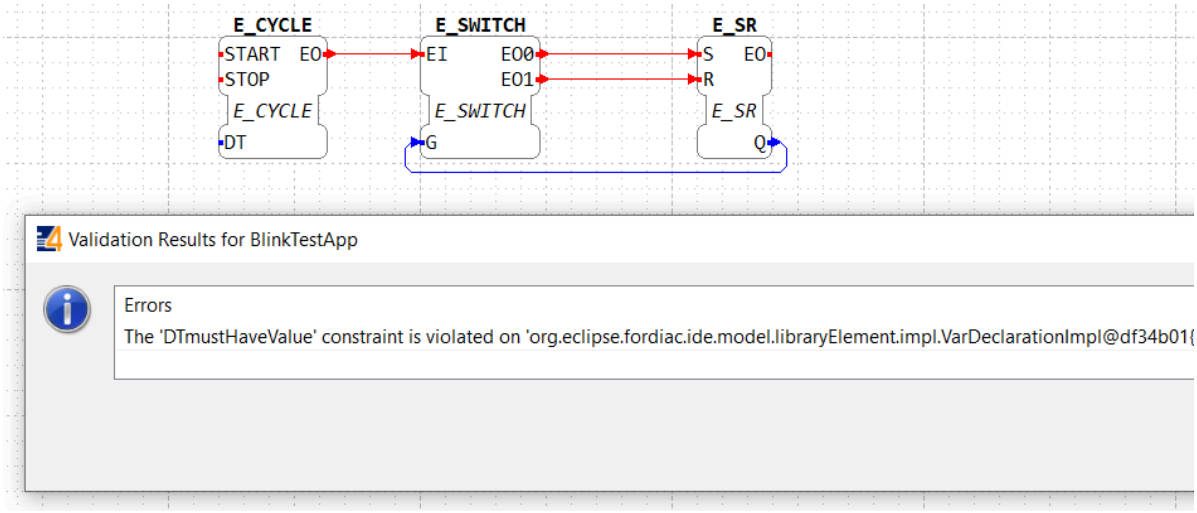
There is a long period of time until the coding period starts (proposal selection period and the community bonding period), which I plan to use to get familiar with particular Eclipse 4diac IDE components relevant to this project.

I have already taken a few steps to get to know the implementation of 4diac IDE. I looked through the ecore metamodels (data.ecore, lib.ecore, palette.ecore, virtualDNS.ecore) of the *org.eclipse.fordiac.ide.model* project. I converted the lib.ecore model to an OCLinEcore document, which makes it more convenient to specify OCL constraints in the OCLinEcore Editor. I implemented a rudimentary validity view, which can display the validation results for a 4diac Application. The validation results can be displayed choosing *Validate* from the context menu of an Application.



So far, I have created simple OCL constraints, like an OCL invariant for checking the value for certain *VarDeclarations*. For example, a DTMustHaveValue invariant tested on the Blinking tutorial[12]:

[12] "Step 1 - Use 4diac Locally (Blinking Tutorial) - Eclipse." https://www.eclipse.org/4diac/en_help.php?helppage=html/4diacIDE/use4diacLocally.html. Accessed 29 Mar. 2020.



I intend to *continue* the implementation of this prototype in the time leading up to the start of the GSoC programme. As I have some experience in Xtext, I would also like to get familiar with the 4diac Xtext models and check whether it would be advantageous to use OCL constraints there. I expect after this "prototyping period" to have a very clear understanding of how the OCL constraints should be developed.

Regarding the background information I will need to be successful: While I was designing the above-mentioned constraints, I realized that I need to gain more experience in the IEC 61499 standard so that I can find out more advanced constraints which are capable of identifying more complex issues in the models. I need to gather a lot of information regarding the IEC 61499 standard. Fortunately, there are good materials on the official 4diac page[13].

5. Deliverables

In this section, I present the list of deliverables that I intend to deliver in my GSoC project:

D1: Creating a list of requirements for rules that need to checked on the Ecore models

- Precise and traceable requirement list for rules (e.g. in a table format) which should be validated on the models
- Assigning the requirements to OCL constraints to facilitate traceability

[13] "Introduction to IEC 61499 in the Eclipse 4diac documentation." https://www.eclipse.org/4diac/en_help.php?helppage=html/before4DIAC/iec61499.html. Accessed 29 Mar. 2020.

D2: Improving the 4diac Ecore models to fit the needs of this GSoC project

- Making suggestions for improvements on the Ecore models
- Adding new elements (e.g. classes and operations) to the Ecore models to facilitate the introduction of constraints

D3: Custom validity view for displaying validation results

- Displaying the violated OCL constraints and the problematic model elements in a user-friendly way
- Highlighting the problematic model elements in the model editor

D4: Implementing OCL constraints based upon the requirement list

- Developing the OCL constraints and well-formedness rules including invariants, derived elements, query operations and operation contracts

D5: Testing the newly implemented features and constraints

D6: Documentation

- Creating both user and developer documentation
- Extra: Publishing results in a research paper (e.g. writing a position paper to the International Workshop on OCL and Textual Modeling[14])

[14] "OCL 2020 : 20th International Workshop on OCL ... - WikiCFP." http://www.wikicfp.com/cfp/servlet/event.showcfp?eventid=99802&copyownerid=161770. Accessed 30 Mar. 2020.

6. Schedule and Milestones

Milestone	Tasks	Weeks	Date
1 - Requirement specification			
1.1	Create a list of requirements for rules that need to checked on the Ecore models	2	01/06/20
2 - Development			
2.1	Improve 4diac Ecore models	2	15/06/20
2.2	Create custom validity view	3	29/06/20
2.3	Implement OCL constraints	3	20/07/20
3 - Testing			
3.1	Test the newly implemented features and constraints	1	10/08/20
4 - Documenting			
4.1	Create user and developer documentation	1	17/08/20

For the GSoC program I plan to allocate at least 40 hours per week, in a flexible manner. My timezone is GMT+2. I consider the time difference won't be an issue. I think I will always be able to effectively communicate with 4diac project developers prior and during GSoC application period.

7. Expectations from the Mentor

Alois Zoitl has a strong background[15] in modelling control systems using IEC 61499. He has several publications in this field. If I got stuck (e.g. understanding the mechanism of certain function blocks), it would be nice if I could ask him for help. I need an in-depth knowledge of the IEC 61499 standard in order to be able to design constraints that can detect complex inconsistencies in 4diac models. On the other hand, I may have specific questions about the code base of the 4diac IDE.

[15] "The IET Shop - Modelling Control Systems Using IEC 61499 ...."
https://shop.theiet.org/model-cont-syst-iec-61499-2nd. Accessed 29 Mar. 2020.

8. Introduction and Contact Information

How would I introduce myself in one paragraph? I am a lifelong learner, problem solver, software developer and perfectionist. Experienced with all stages of the development cycle for various projects. Well-versed in numerous programming languages including Java, C#, JavaScript, and Swift. Strong background in domain-specific modeling.

Education

BSc: Computer Science Engineer - 2017 Budapest University of Technology and Economics, Hungary

MSc: Computer Science Engineer - 2019 Budapest University of Technology and Economics, Hungary Excellent with highest honours

PhD: Informatics Sciences (Present)

Awards

Fellowship granted by the Hungarian Republic

Archduke Joseph Award

Email: Bacsi.Sandor@aut.bme.hu

Postal Address: 24 Gesztenye Street, Gyál 2360 HUNGARY

Phone: +36304978633