Experiment for Layer Interaction Diagram Based on Context-Oriented Programming

CHINATSU YAMAMOTO†1 IKUTA TANIGAWA†2
TAKESHI OHKAWA†1 MIKIKO SATO†1 HARUMI WATANABE†1

Keywords: Context-Oriented Programming, Model, Model-Driven Development

1. Introduction

This article describes the first step experiment towards one of Model-Driven Development (MDD) tools with a layer interaction diagram based on Context-oriented Programming (COP). Recently, robots are expected to support multi-functional services, as same as smartphones. However, services on many of current robots are limited a single. For providing multiple services, the system needs to change services at runtime. We expect COP helps to satisfy this requirement because it can change behavior to depend on the contexts [1]. Needless to say, MDD is important for development as well as programming. Towards MDD based on COP, our research team proposed a layer interaction diagram [2]. A layer interaction diagram aims to solve the problem of layer interactions and exclusive controls. In the previous work, the layer interaction diagram did not generate source code. To achieve this final goal of MDD, we make a development plan.

The remainder is organized as follows. Sections 2 explains the layer interaction diagram. Section 3 illustrates the development plan for MDD and the first experiment. Finally, Section 4 concludes this article.

2. Layer interaction diagram

This section illustrates the layer interaction diagram. This diagram aims to draw relations among layers. The relations of the layers consist of communication and (de-) activation. Figure 1 shows an example of the diagram, which shows robots' behavior depends on the surrounding environments. The robot moves to the surrounding environments: Wooden Floor, Carpet Floor, and Trash Station. On the Wooden Floor, the robot wipes it; On Carpet Floor, the robot vacuums trash. When the robot is aware of the full of its trash box, it moves to Trash Station for throwing trash away. During going to the trash station, the robot does not behave the wipe and vacuum. It means the robot keeps the behavior on Trash Station layer until throwing out trash.

3. Experiment

This section explains the development plan and the first experiment. This plan is as follows.

(1) Prototyping Analysis: Assume a simple MDD process that is limited to the layer interaction diagram and state machine diagrams. Generate codes manually on this process. Check the behavior, whether it follows the intention of the model.

(2) Design: On the result of the above activity, we reconsider elements of the layer interaction diagram. Define the elements of the diagram as stereotypes of UML.

(3) Implementation: Implement the layer interaction diagram and construct code generation.

(4) Evaluation: Build an application on MDD with the layer interaction diagram.

Currently, we experiment the first step, as shown in Figure 2. This experiment checks whether the manual generation code follows the intention. It means the robot behaves whether follows the model. Figure 2 shows a part of experiment: the activation with a critical region. During the critical region, the first layer keeps its behavior and is inhibited the activation of the next layer.

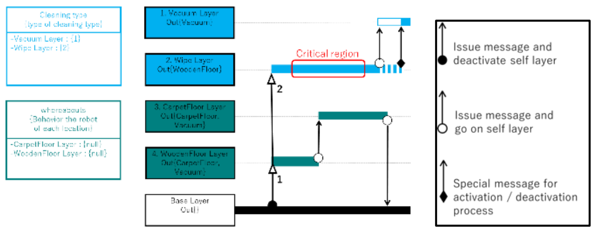
Cleaning type
(type of cleaning type)
-Vacuum Layer : {1}
-Wipe Layer : {2}
whereabouts
(Behavior the robot of each location)
-CarpetFloor Layer : {null}
-WoodenFloor Layer : {null}
1. Vacuum Layer Out(Vacuum)
2. Wipe Layer Out(WoodenFloor)
3. CarpetFloor Layer Out(CarpetFloor, Vacuum)
4. WoodenFloor Layer Out(CarpetFloor, Vacuum)
Base Layer Out()
Critical region
2
1
Issue message and deactivate self layer
Issue message and go on self layer
Special message for activation / deactivation process

Figure 1 the layer interaction diagram

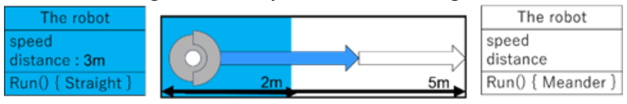

Figure 2 Experiment for activation with critical region

4. Conclusion

This article introduced the plan and an experiment towards an MDD tool for COP. In future work, we will develop the MDD tool according to the plan. In this process, we will apply major tools. Finally, we will build to generate the source code method and to implement tools.

Reference

[1] R. Hirschfeld, P. Costanza and O. Nierstrasz: Context-oriented Programming, Journal of Object Technology, Vol.7, No.3, pp.125-151, 2008.

[2] Harumi Watanabe, Ikuta Tanigawa, Midori Sugaya, Nobuhiko Ogura, Kenji Hisazumi: A Layer Structure Diagram and a Layer Interaction Diagram towards a Context-Oriented Development Methodology for Embedded System, Workshop on Live Adaptation of Software Systems co-located with 15th International Conference on Modularity, 2016.

†1 Tokai University
†2 Kyushu University