Model and Verify CPS using the Vienna Development Method

Shehroz Bashir Malik[1]

Contents

[1] shehroz-bashir.malik@stud.hshl.de

Abstract: In this paper, I will formally describe the Vienna Method. I will explain its theory and background. I will find and describe case studies. I will find tools that support it.

1 Motivation

[?]

2 Foundation

3 What is a Cyberphysical System?

In this section, I am going to define CPS, give examples of it, why do we need it, why is it the future https://www.youtube.com/watch?v=8HLJFlncMls good reference

4 Why do we have to verify it?

self explanatory

5 Why do we have to validate it?

self explanatory

6 Vienna Development Method?

6.1 Introduction

The Vienna Development Method was developed in the IBM Laboratory Vienna in the 1970s to model and develop sequential software systems. [AP98] This specification language was standardized by the International Standards Organization (ISO) in 1996. [ISO96]
In order to specify, develop and verify computer systems - Formal Methods need to be employed. These involve the use of mathematical logic to perform rigorous analysis to ensure the reliability and robustness of a system.[La22].
VDM allows you to examine regions of incompleteness and vagueness in system specifications thus ensuring your final output is a valid implementation of your initial design. It is able to maintain flexibility is a landscape of ever-changing requirements. Multiple tools have been developed to employ VDM. The ones I will be exploring in this paper are Overture [Ov22] and VDM Tools [VD16].
An extension of the VDM Specification Language was developed to embrace object-orientation and concurrency. It is known as VDM++. It allows one to describe real-time distributed and embedded systems.[VLH06]

6.2 VDM-Specification Language

1. Types - There are two types namely Primitive Types and Quote Types. Primitive Types are similar to declaring a variable. All sorts of arithmetic and logical operations can be carried out on these types. It includes high-level support for Mathematical Set Notation. These can include

$$\begin{aligned} \mathbb{N} - Integers \\ \mathbb{Z} - NaturalNumbers \\ \mathbb{R} - RealNumbers \\ \mathbb{Q} - RationalNumbers \\ \mathbb{B} - Boolean \end{aligned} \tag{1}$$

2. Composite Types

6.3 VDM++ Notation

VDM++ employs a lot of concepts from Object Orientation. It can contain a group of class specifications. These can be categorized into two types, namely Active and Passive Classes. Active Classes can run on their own without an external stimulant whereas Passive Classes can be activated from an Active Class.
Each individual class has its own set of components:

1. Class Header - Class Name Declaration with support for single or multiple inheritance
2. Instance Variables - Variables along with their types are declared in this section. An invariant and the initial state can also be defined.
3. Functions - Can modify the state using implicit and explicit expressions but they can not refer to instance variables
4. Operations - Can modify the state using implicit pre/post condition expressions or statements. Considered to be more powerful than functions.
5. Threads - Threads enable concurrent behaviour. It is a stream of actions that need to be performed while the class is active. It can also be a unlimited stream of actions or in other words: an infinite loop.
6. Synchronization - Certain Boolean conditions need to be set in order to maintain mutual exclusion and enable concurrency.

VMD++ also includes basic concepts from C++ such as Constructors and Access Modifiers.

7 Case Studies of VDM

INCLUDE AN EXAMPLE OF CODE GENERATION TO C++/JAVA

7.1 Self-Balancing Scooter

8 Tools

9 Future Works

maybe I can add a section that discusses more about what has been done

10 Conclusion

Bibliography

[AP98] Alagar, V. S.; Periyasamy, K.: Vienna Development Method. In: Specification of Software Systems, pp. 219–279. Springer New York, New York, NY, 1998.

[ISO96] : Information technology — Programming languages, their environments and system software interfaces — Vienna Development Method — Specification Language — Part 1: Base language. Standard, International Organization for Standardization, Geneva, CH, December 1996.

[La22] Langley Formal Methods Program • What is Formal Methods, 06-Oct-22.

[Ov22] Overture Tool, open-source integrated development environment (IDE) for developing and analysing VDM models., 28-Aug-22.

[VD16] VDMTools v9.0.6 by Kyushu Univeristy, 25-Oct-16.

[VLH06] Verhoef, Marcel; Larsen, Peter Gorm; Hooman, Jozef: Modeling and Validating Distributed Embedded Real-Time Systems with VDM++. In (Misra, Jayadev; Nipkow, Tobias; Sekerinski, Emil, eds): FM 2006: Formal Methods. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 147–162, 2006.