A14-05 Detailed mechatronic SysML model development for a Festo didactic manufacturing station

Heng-you Lin

10/18/2014
A14-05 Detailed mechatronic SysML model development for a Festo didactic manufacturing station

- Instructor: Docent Seppo Sierla, seppo.sierla@aalto.fi, room 1563 (TUAS building)
- Credit estimation: 8cr
- Required knowledge: All members are required to have a solid grasp of SysML modeling in the MagicDraw environment, e.g. from the course "AS-116.3192 Managing the Product Life Cycle".
- Workers: Heng-You Lin

Introduction:
In this intermediary report, I will present the current status of the project by showing what has been done so far. The report will be structured as follow: in section 1) I will present the modelling works that have been constructed, which was conducted with an agile; in section 2) I describe my decision on the granularity of the model and the rationale behind it; and lastly in section 3) I will discuss a possible change to the project task in response to a risk that has arisen.

1. The status of the project: modelling works so far
   As described in the project plan, the project is carried out in an agile fashion where the distribution station is chosen as the priority target, which with the experience gained during the process form the basis of the discussion to the answer the research question of granularity. Apart from allocation of logical behaviours to physical model elements, as well as the logical communication model with the subsequent stations, the model for the distribution is practically completed. Before I present the model in SysML diagrams below, I want to emphasize that a set of model diagrams is different to the model itself and it is unnecessary and also difficult to display all aspects of the model on SysML diagrams. The complete model can be viewed with a model development tool such as NoMagic MagicDraw as used in this project. Nevertheless the following diagrams provide a sketch of what the distribution station model is like.
Physical architecture:
The first step in modelling the physical architecture is to create SysML blocks, which can be think of as classes in traditional UML, that represent every object in the system from a pneumatic valve, to an optic sensor to the work pieces that are to be work on. By populating them on a Block Definition Diagram, or bdd, it allows the modeller to quickly understand and add the necessary relationships between components (e.g. decomposition and dependency), the properties of each component (e.g. pressure of compressed air), boundary interface (ports), and the services/operations they provide(e.g. to activate the motor). The following bdd, named Distribution Hierarchy shows how the blocks representing the physical objects that make up the distribution station are split into two dimensions, the mechanical and the electrical dimension and more importantly, the instances of blocks that is present which will populate the ibds in the next section.

![Distribution Hierarchy](image)

*Figure 1 bdd [Distribution Hierarchy]*

After all the required components have been modelled as blocks and their instance created, the next step is to populate these instances (parts in SysML jargon) in an Internal Block Diagram or ibd, in order to display the physical connections between the parts. For example in figure 2 below, it shows which and how the mechanical components are connected to each other, as well as the flow of work
pieces from one component to another. In the figure 3, the electrical dimension ibd, it illustrates the electrical connections made between the sensors, actuators and the processing unit, especially where input and output signals are generated and connected to the PLC/actuators.
**Logical Behaviour:**
To understand and model the desired logical behaviours of the distribution station, I first created a use case diagram to illustrate the black box response of the system the interaction of the modelled system with its external environment (the system operator and any other utility provider). The use cases are created based on a textual description (shown in the text box in the figure below) and in comparison with the text, presents the modeller a better understanding of the desired system response, the inter use case relationship, as well as what actors or external system that is involved in each part of the system’s desired behaviour. The further refinement and decomposition of the use cases allows a complete understanding of what behaviour needs to be modelled. Figure 4 represents the use case involving the distribution station.

![Figure 4 uc [Distribution Use Case]](image)

The use cases specifies what the system should respond upon an invocation by the external system actors, but does not specifies how the system should do it. Therefore the next step of the modelling is to use create activities and populate activity diagrams with them to describe the control flows and the flows of physical objects through the system during those use case invocation. Figure 5 shows on
such activity diagram describing the activities involved in a demo sequence on the distribution station.

Apart from the activity diagram, the desire behaviours of each component can also be illustrated using state machine diagrams as shown in the figure 6 and figure 7 below. A benefit of using state machine diagram is it will be easier to produce either the IEC61131-3 SFC or the IEC61499 ECC if control software is to be developed later for the system.
2. Discussion on Granularity:

A major research question in this project is on the granularity of the SysML model, or in other word, how much detail the model element should be decomposed into finer elements. The following section provides a description and the rationale behind it for the chosen level of granularity in this project.
There is no universal standard in how granular the model should be in MBSE. Across literature, the granularity of a model varies differently based on the nature of the target system, and the subjectivity of the system to simulation and verification. For example, in [1] the authors described a case study where they used SysML to model a subsystem of a nuclear power plant. Due to the fact the SysML model created are not intended to be used for simulation, the model was only developed to a level of sufficient detail so it was possible for the authors to discuss the benefit of using the language. The authors also point out the practical difficulty of modelling too much detail which will results in unreadable diagrams and overly complex model. In [2] the authors argue that in fact SysML lacks the capability to model very fine domain specific - details for every domain, even when the language is extended with domain-specific stereotypes. Thus the authors suggest only model the high-level abstractions, which describe only the system at a functional and requirement level and may also include relevant information that is necessary to drive the later development activities. In [3] where the author model a similar FESTO MPS plant as in this project, the author has structured the model and select the level of detail that is suitable to his later need in verification and converting the SysML model to executable IEC61499 code.

These examples suggests that for this project, the granularity of the model is not subject to any strict rule or methodology, as the model created from this project is not expected to be subject to simulation, verification or other further system development, it is therefore up to the modeller to decide on a suitable level of detail that conveys all relevant information. Although, I feel that it is helpful to come up with one non-binding guideline to decide how fine the detail the model should be, this help to keep the model element consistent throughout the system. The following paragraphs describe the guideline I will follow in modelling the MPS system.

**Architecture:**
For modelling the architecture, the system is divided into two dimensions: mechanical and electrical components. For mechanical components, granularity of
the model element is determined by following the two principles described below: 1) The system is decomposed until the decomposed structure is no longer involve in an direct interaction/use case with the flow of the physical object the system is acting on, that is, in our model the flow of work piece element. To illustrate this principle, use case analysis is carried out for the distribution station, and even though in practice of SysML modelling, chain of included use cases are discouraged, it is used here to show a functional decomposition of the base use case (one which interacts with the primary actor, the system operator), run Demo Sequence, which can be in figure 8 below.

![Figure 8 uc [Chain of included use cases]](image)

After the chain of use cases are created, each use case is then allocated to a responsible SysML block of mechanical element, which is arranged in a similar hierarchical fashion. In this way, the system is decomposed into different layer of abstraction, where each use case of a given abstraction level represent a black-box response to the encapsulated architecture and behaviour. For example by following this principle, we can see that the pneumatic cylinder or the stack magazine is not decomposed any further, as there is no inferior corresponding use case that encapsulated their responses, hence they represent the most detailed level for mechanical components in the model.
The second principle is applied after the decomposition based on the first principle has been completed. It is obvious that for some use case, the components which they have encapsulated the responses do not include all the necessary service/structure to perform the desired black box response, thus the second principle is applied when 2) any other mechanical component which directly provide a service to the blocks encapsulated by the use case, is also modelled as an independent block. For example in our model, the rotary drive block is responsible for the use case ‘pick up work piece’, where it requires the service of compressed air for actuation. This service is provided by the pneumatic valve system including a 3/2 solenoid valve and a vacuum generator, and hereby the second principle, these two components are also added to the model. This service require/provide relationship is illustrated in figure 9.

Now moving on to the electrical dimension, the principle on deciding the granularity is much simpler, since in this MPS system, all sensors are single bit binary signal generator whose values are all read by the Digital Input terminal s attached to the PLCs, thus any component that is capable of generating such a binary signal is considered to be at the lowest layer of abstraction in the model hierarchy. Examples of these electrical components include the various buttons on the control panel (with binary signal pressed or not pressed) and the optical sensors used to detect the positions of the work pieces (work piece present or not present). In this essence, the hierarchy of the electrical dimension can be
considered as building from the bottom up, in contrast with to the top down approach in modelling the mechanical dimension.

Lastly, I want to mention that the guideline described above is non-binding, that is, while I will stick mostly to this guideline, it is not necessary illegal to decompose a component to a finer detail when needed, but a note of rationale should be added to the model to explain the reason of including the said element, especially explaining why instead of model it as a port or interface of a higher level component.

3. Response to Risk 1

A possible change to the project is a response to the rise of risk 1, with regard to underestimating the workload for task 3.1, that is to investigate academic/industrial works on MDD with model mapping and code generation, which is originated from a requirement in the project description: to develop the model to such a level of detail that there is a well-defined mapping to software and to IEC 61131-3 PLC programming constructs in particular. After some literature research, I feel that workload required in fulfilling this task and requirement is beyond the scope of this project. The reason is as follow: there is rather limited research work on mapping SysML model to IEC61131-3 Programming language, and for the few works that is present in this research domain, there is very few detail discussion or specification on the level of detail required for the SysML model to undergo such a mapping process. For this reason, the only way to complete this task is to study closely the mapping techniques described in these works and deduce from it the level of detail the SysML model is to be built. I feel that this requires too much workload and may divert the focus of this project from practical model building. I will discuss this possible change of the project requirement with the supervisor in order to concentrate the focus of this project on model building.
Reference

