UML based software development under safety constraints

Christian Diedrich, Jan Krause, Andreas Franke
Informationsmanagement für Automation und Umwelt
ifak e.V. Magdeburg
Steinfeldstr. 3
39179 Barleben

cristian.diedrich@ifak-md.de
jan.krause@ifak-md.de
andreas.franke@ifak-md.de

Abstract: Automation devices which are based on digital information processing fulfilling functional safety is an increasing market area. This increases the requirements on the software of the communication controller of the devices. UML as a semi-formal method can provide new implementation and testing methods in this area. This paper gives an overview about the specification of a UML PROFIsafe specification and its approach to come to a generated implementation.

1 Introduction

There is a gap between formal languages and methods for the specification, verification and test case generation of software systems and the product development process in the industry. Many formal methods are very abstract so that engineers are not in a position to use them during their daily work. UML as a collection of graphical languages and the possibility to integrate source code seems to be a means to fill this gap. Safety software systems are needing formalised development processes. UML language elements /OMG: 2002/ and the V model are suggested for this reason. The V model steps influenced by requirements of functional safety are described in the IEC 61508 standard “Functional safety of electrical/electronic/ programmable electronic safety related systems”, mainly in /IEC 61508-1 and IEC 61508-3 : 1998/.

The PROFIsafe profile is a safety related profile for PROFIBUS and PROFINET IO. It is intended to use UML and requirement specification methods for the entire profile development similar to the use of PROFINET IO train profile development /Winzen : 2004/.
While the V model describes the development process in general, the safety requirements are constraints to each step of the V model. The safety requirements influence the activities and decisions of the V model steps (Figure 1). Both V model and IEC 61508 do not define the languages which have to be used. Therefore the practical profile development work takes use of UML for those steps where UML provides language elements. This gives the opportunity to use the consistency rules of the UML meta model during the check of the specification.

Covering the constraints of safety, real time and security the means of UML have to be restricted. The UML community provides so called UML profiles with subsets of the UML meta model for the language such as state machine, sequence diagram or class diagram. Based on these UML profiles and additional specific stereotypes and tagged values, the UML languages are specialised for the use for PROFIsafe. The profile developers will use these UML profiles during their specification work.

Verification and test is part of the V model. Verification and test case generation need formal specifications. Therefore one focus of the UML profile is the transformation of parts of the UML diagrams to specifications which are based on formal models for example state machines to Extended Place Transition Petri nets /Krause:2005/. This gives the opportunity to integrate verification and test case generation in the test specification.

Figure 1- Aspects of software development processes
2 PROFIsafe Specification

Starting point for the use of digital communication systems in the area of safety is the analysis of possible errors and faults. The analysis results are shown in Table 1 where both the possible errors and the means for their recognition mechanism is shown.

<table>
<thead>
<tr>
<th>Failure:</th>
<th>Measure:</th>
<th>Consecutive Number</th>
<th>Time expectation with acknowledge</th>
<th>Codename for sender and recipient</th>
<th>Data Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repetition</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Loss</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Insertion</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Incorrect Sequence</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrupted Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interconnection of F- and Standard Messages (Masquerade), incl. wrong - and double addressing</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Table 1 Errors and their recognition mechanisms

PROFIBUS provides supervision mechanisms within layer 1 and 2. The safety-integrated level 3 (SIL3) needs a probability of dangerous errors per hour between 10-7 and 10-8 between sensor and actuator. The contribution of the digital communication to the probability of dangerous errors per hour should not be higher than 1%, i.e. it has to be 10-9. Therefore additional means have to be integrated.

The PROFIsafe /PROFIsafe:2004/ solutions add a so-called safety layer on top of PROFIBUS layer 7. This gives the opportunity to use the existing industrial communication stacks as well as to operate a mixture of non-safety and safety application on the same segment at the same time.

PROFIsafe uses the polling mechanism of PROFIBUS between masters and their associated slaves. The safety means are encapsulated in the safety layer and the communication stack as well as the medium are treated as a so-called black channel. The possible errors such as damaged addresses and loss or delay of data are recognised by using the following mechanisms:

- A continuously increasing numbering of the data.
- A time supervision both in the master as well in the slave.
- Authorisation by passwords.
- An optimised CRC check sum calculation.

Data for these mechanisms is transferred additionally to the cyclic data between masters and slaves.
Devices have to be commissioned before they are switch in the safety mode. Figure 2 shows the safety commissioning state machine from /PROFIsafe_PA:2004/ in UML notation\(^1\), which specifies the behaviour of PROFIBUS PS devices, which can be operated in the Normal- as well as in the Safety-Mode. The state machine has one state (S4) in which the safety cyclic communication is active. The other states are necessary for parameterisation of the device and to switch into the safety mode.

\[\text{Figure 2 – Safety commissioning state machine}\]

### 3 PROFIsafe profile implementation and test

Each step of the V model has to be accompanied by verification and tests. The test and verification have the following aspects:

- The state machines (representing the behaviour of the components) specified in the profile has to be verified. The verification proves the quality of the state machine itself. Examples are correctness, freedom of dead locks and others. There are a broad range of model checking methods and tools available. Of course the main problem remains to bring the specification into the formal inputs. This topic is not the focus of this paper.

- Proof of all requirements, use cases and scenarios including the possible failures are fulfilled by the designed behaviour, i.e. validation. Using the V model coverage analysis, requirement tracking can be done. FMEA and Fault Tree Analysis have to be done additionally.

---

\(^1\) For reasons of clarity only the parameter (INSPECTION) is shown here.
• Proof of the correctness of the implementation against the specification, i.e. test of the devices and sub-systems. The implementation could be generated by today’s UML tools such as e.g. Rhapsody of i-logixs and XD of IBM. Even the tools provide C code according to a safety oriented C convention (Rhapsody provide C code according to the MISRA-Guideline) the C code has to be proven separately because the UML tools and their generators are not proven tools in terms of safety. Additional means have to be done for their use in safety devices.

4 Extended P/T Petrinets (EPTPN)

The PROFIsafe commissioning state machine for PA devices is used as example for the test of the correctness of implementations based on a UML specification. The method defines that the UML state machine is transformed into a Place/Transitions (P/T) Petri nets. The state machine includes guards at the transitions so that there made two extensions for the Place/Transitions (P/T) Petri nets based on /Roemer:2000/ in /Krause:2005/.

• EPTPN can have variables.
• The transitions of an EPTPN receive a UML syntax and semantic.

The developing nets are called Extended P/T Petri nets (EPTPN). Figure 3 shows the EPTPN of the UML state machine from Figure 2.

![Figure 3 – EPTPN of the UML state machine from figure 2](image)

Based on these extensions the EPTPN can be analyzed with the same methods (Deadlock and Reachability analysis, Model Checking) as the classical P/T Petri nets.
In addition to the well-known behaviour representation as reachability graph there is the possibility to compute the complete prefix of the unfolding of the Petrinet. Roemer:2000/ presents an efficient algorithm for this complete prefix computation of the unfolding by P/T Petrinets if some additions are made to meet the characteristics of a EPTPN /Krause:2005/. In particular the computed complete prefix of the unfolding net contains all possible states (markings) and processes of the Petrinet and is in the information content equivalent to the reachability graph. Figure 4 shows the complete prefix of the unfolding of the EPTPN from figure 3. One test case is a concatenation of all branches until a leaf of the tree.

![Figure 4 – prefix of the unfolding of the EPTPN from figure 3](image)

The following steps summarise the basic ideas of the test concept (Figure 5):

- Source code is a generated format using the integrated transformer of the UML tools (e.g. Rhapsody). Rhapsody provides C code according to the MISRA standard.
- The source code is compiled into the target format (binary code)
- The specification is exported into the standardised XMI format. A transformer uses the XMI project export and extracts the state machines
- The state machines is transformed into Extended Place Transition Petrinets (EPTPN) /Krause:2005/
- The EPTPN is the input for a verification and test case generation tool. This tool uses existing and approved methods. The verification result is a feed back to a possible redesign, the generated test cases are the basis for the profile test /Krause: 2005/
• The tests are carried out by a test tool which is able for remote tests using PROFINET IO communication services. The communication services have specific parameters which are also a result of the test case generation. The test cases stimulate communication services in the embedded target, i.e. the system under test (SuT) responds according to the implemented internal behaviour. The test tool is able to compare the response with the reaction which has to be responded in line with the specification.

• If all tests have passed positively, the implementation is in line with the specification. Both development paths, the test case generation and execution as well as the generation of the embedded code, are based on different tools with different development processes.

![Diagram showing the process of test case generation](image)

**Figure 5 – Test of embedded targets based on test case generation**

The following steps have been reached up to now:

• It is possible to generate embedded code out of a UML specification with a relatively low resource consumption and safety C guidelines

• The extraction of state charts and their transformation in a formal model (the authors use extended state transition Petri nets) is available
• The calculation of the complete prefix of the unfolding on the EPTPN is available

• Connection to a tool for model checking is not yet available

The next steps are the comparison of existing test cases for PROFIsafe with the generated one using the transformation to EPTPN method and the definition of the procedures to integrate the generated C code in the safety device.

5 Literatur


