4. Schedulability Analysis

Once the PIM model was checked for its functional properties, a key issue, in real-time software, is to evaluate whether its tasks (threads) are scalable. To perform this verification key, this paper proposes to evaluate the PSM, once it has exactly the structure and behavior of software on the target platform and such information are key factors in this type of analysis.

However, this paper has not explored techniques or tools to determine the WCET, which is essential for the schedulability analysis. Thus, an expert in software should complement the PSM with additional information, e.g., the definition of mandatory activation time (WCET) of a class annotated automatically with the stereotype *SwSchedulableResource*. The present work suggests the use of historical information or evaluations of small pieces of code on the target platform to obtain this information.

Since there is a PSM model, and this was complemented with information about the activation time of each element which contains the stereotype *SwSchedulableResource*. It is possible to perform schedulability analysis. This paper has chosen to transform the PSM into an AADL model to allow the assessment of schedulability in any tool able to interpret AADL models. To generate an AADL model a MOF M2T transformation is applied on PSM. This transformation is simple, since the PSM contains all information necessary for generating the AADL, standing out period, time and activation time for each task as well as shared objects between them.

The next section presents the case study used to evaluate the proposed approach.

IV. Case Study

A case study was developed to evaluate the proposed approach, and selected tools. Specifications, models, and transformations discussed above were applied to part of the attitude control of the Multi-Mission Platform (MMP) from INPE (National Institute for Space Research) in nominal mode presented by Moreira (2006). The modeled and verified part was the sensor reading. Sensor reading subsystem encompasses: collect measures, computation of a simple transfer function, and sending results to other system modules. This module was selected by combining algorithmic simplicity (only a simple transfer function), and coverage (it would be possible to exercise all elements addressed in the current article). The target platform selected for case study was Java language, and part of characteristics of MMP. It should be noted that case study is not intended to generate code applicable to MMP but it assesses whether the proposed approach is feasible.

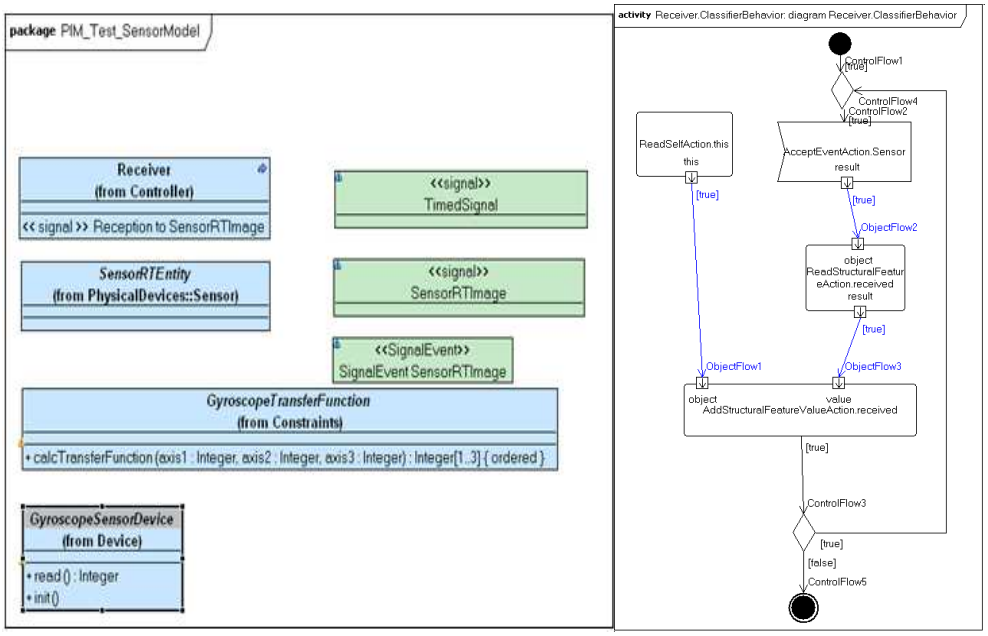
In order to allow comparison of the proposed approach, the same system using the same level of detail was modeled using an approach where all information is defined in a single model containing characteristics of the target platform. This model is called detailed design model.

A. Modeling according proposed approach

As proposed by the approach outlined above, case study began with two independent activities: definition and verification of platform-independent model, and target platform definition.

1. Platform Independent Model

First, the subsystem structure of the sensor reading was defined. Figure 6.a shows part of this structure. Afterwards, behaviors and the test scenarios were modeled using fUML for this subsystem. Figure 6.b presents a behavior defined for verification purpose.



(a) (b)

Figure 6. PIM – (a) Structure; (b) Behavior.

Moreover, test scenarios were executed to verify the expected functional properties. Figure 7 shows execution results for a test scenario, *testGyroscopeSensorDevice*.

```
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [receiveOffer] node = testSucc
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [_beginIsolation] Begin isolat
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [receiveOffer] Firing.
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [_endIsolation] End isolation.
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [fire] output activity paramet
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [addToken] node = testSuccess
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [removeToken] node = result
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [fire] Checking if TestIdentit
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [_beginIsolation] Begin isolat
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [_endIsolation] End isolation.
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [fire] Checking if ValueSpecif
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [_beginIsolation] Begin isolat
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [_endIsolation] End isolation.
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [execute] Activity testGyrosco
13 Aug 2010 00:04:17,122 DEBUG [main] fUML.Debug [destroy] object = org.modeldr
[execute] output parameter 'testsuccess'has 1 value(s)
[execute] value [0] = true
```

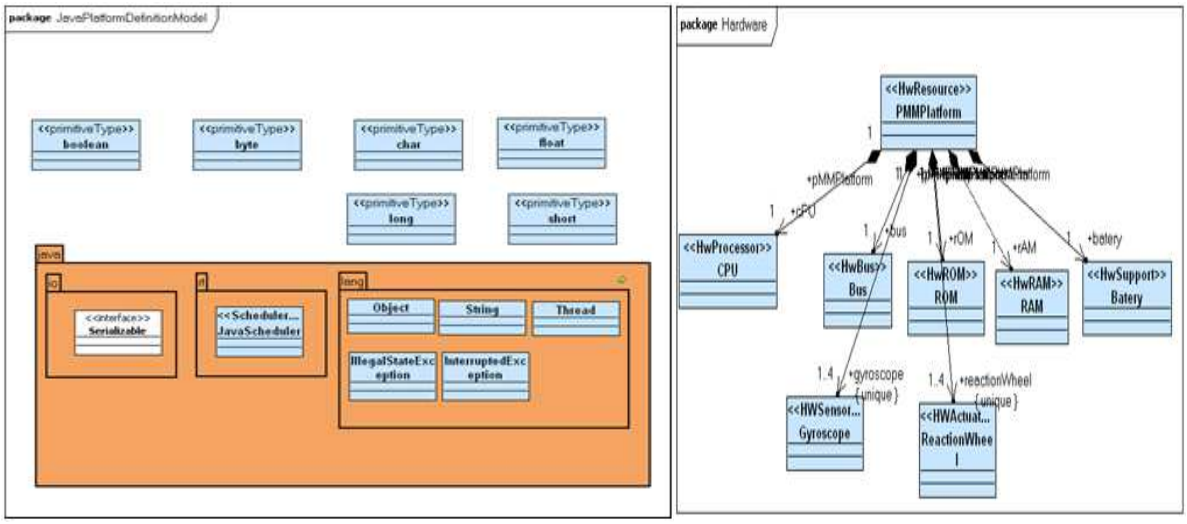
Figure 7. Platform independent test cases execution

At this point, PIM for reading sensor subsystem was modeled. This model was functionally verified through test scenarios execution. All test scenarios were executed using fUML- Reference Implementation 0.4 (ModelDriven, 2009). It is important to notice that non-functional properties were defined in this model, but they have not been verified since they do not belong to the functional domain.

2. Target Platform Definition

Concurrently, a target platform was defined following previous described characteristics.

This activity developed two models: (a) *PDM - Software* containing the essential part of the Java language that it is required for the transformation from PIM to PSM; (b) PDM - Hardware containing a description of some MMP real characteristics. Figure 8.a and 8.b shows part of these two models.



(a) (b)

Figure 8. (a) PDM – Software; (b) PDM – Hardware.

Afterwards, transformations were defined:

- m2m – model to model, it transforms the structure of a PIM (following the UML meta-model) to PSM (considering PDM for Java language as a target software platform);
- *blackbox library* – it is triggered by m2m transformation aim at transforming fUML activity diagrams into behavior in Java. Results are stored in PSM;
- m2text – responsible for transforming the PSM model into Java code.

3. Using target platform

At this point, we had target platform defined so it is possible to transform PIM (following the meta-model expected by the target platform) into PSM; eventually, code and AADL model can be generated.

Transformations from PIM to PSM and PSM to code were executed successfully. Figure 9 shows size of models through evolution of proposed approach. It is possible to see how big the PSM is compared to PIM. Indeed, this is due to transformation from PIM to PSM.

The transformation from PSM to AADL was performed, which allowed schedulability analysis by using the automatic evaluation of the model AADL with Cheddar (LISyC, 2010). This analysis indicated that the PSM model was scalable. As a result, generated code (LOC = 1255) could be used with indications of functional and temporal correction.

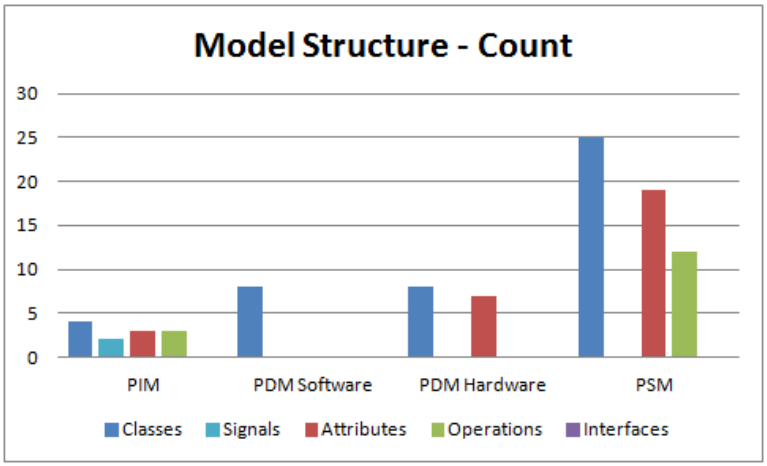


Figure 9. Model structure size evolution through proposed approach

B. Modeling using a detailed design model

Usually, real-time software requires high level of reliability. One used technique is to design it using some graphical notation, for example, UML. This is done in order to express some level of technical details. In line with this point and to allow comparison between approaches, the same system using the same level of detail was modeled using an approach where all information (detailed structure and behavior) are defined in a single model containing characteristics of target platform. To allow a comparative view of approaches, behavior was modeled using activity diagrams, considering only actions provided by fUML.

fUML was not created for this type of detailed modeling but it was used to allow the comparison. It is important to notice that the manual translation of activity diagrams to code can cause errors and requires an unnecessary effort from the point of view of proposed approach. Another alternative would be to use a tool that is able to transform activity diagrams into target language.

V. Conclusion

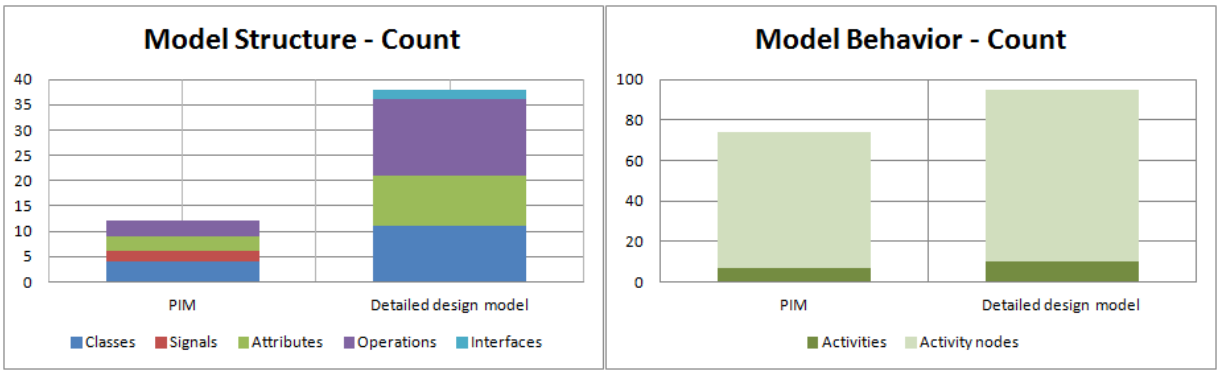
During case study, it has been possible to exercise the feasibility of the proposed approach. An important point of this work is its independence from commercial tools since the proposed approach is based on vendor neutral specifications. Another important point is that the case study used only open source tools, namely:

- TOCASED 3.3 (TOPCASED, 2011) – for UML;
- Eclipse Galileo Modeling (Eclipse Foundation, 2011) – to develop and to execute model to model and model to text transformations;
- fUML Reference Implementation 0.4 (ModelDriven, 2009) – to run PIM models defined using fUML;
- Cheddar 2.1 (LISyC, 2010) – to perform the schedulability analysis for an AADL model;

Approaches were compared considering the work of Monperrus *et al.* (2007) so a set of metrics for counting were selected, as follow:

- Structure: classes count, amount of signals (existing only in PIM), attributes count, operations count, and interfaces count (it only exists in the detailed model of the project due to the fact that fUML exclude *Interfaces* and target platform does not use them).
- Behavior: activities count, and activity nodes count (*ActivityNode*) required to model the behavior.

Figure 10 presents results for measures defined for the structure (10.a) and behavior (10.b). From data in Fig. 10, there is indication that proposed approach is complete, and it reduces size of model manipulated by users. It is complete because it defines the structure, behavior and non-functional properties of time on a single high level model. Considering PIM as reference, PIM structure is 175% lower with respect to count classes; moreover, it reduces operations count and attributes count. Regarding the behavior, reduction is less significant but it is still considerable. Activities count is 42% lower in PIM.



(a) (b)

Figure 10. Comparison between approaches; (a) Structure; (b) Behavior.

From our point of view, the proposed approach is advantageous whether we evaluate together with safety-critical software requirements that requires in detail structure and behavior modeling. Moreover, the approach still has the following benefits: (a) alignment indicated (without manual intervention) by the transformations between the PIM and PSM; (b) evaluation of non-functional properties based on models (schedulability); reuse of models and transformations; definition and use of hardware aspects.

In conclusion, this paper proposes a vendor independent approach that uses:

- PIM defined with fUML and MARTE (HLAM) – it allows modeling of structure, behavior and time requirements;
- PDM - Hardware and PDM - Software defined with MARTE (GRM, HRM e SRM) – they allow modeling of hardware elements and characterization of the target platform;
- MOF QVTO – it performs the transformation of a PIM into a PSM model considering the PDMs, evaluating fUML to generate behavior on the target platform;
- MOF M2T – it generates code for the target platform and an AADL model for schedulability analysis.

Future studies are related with the weaknesses of the current proposal. The most important are:

- It is difficult to model behavior using fUML through activity diagrams. They are very extensive and complicated. It is necessary to integrate the ALF (Foundational Action Language for UML), a textual language for concrete actions defined by the OMG (OMG, 2011);
- It is difficult to model continuous systems. Control algorithms are more complex when they are modeled using fUML. It is necessary to incorporate some type of value specification language (VSL), as defined in MARTE;
- Test scenarios in PIM are manually defined by domain experts focused on functional model; therefore, this creates the possibility of uncovered test scenarios. It is necessary the integration of the formal mechanisms to PIM model verification. Another concern is to use a UML Testing profile (OMG, 2012) to define them.
- It is difficult to ensure transformation correctness so it is important to evaluate alternatives to certification.
- Target platform definition requires high efforts. It is necessary to evaluate alternatives to parameterize transformations in order to increase your chance of reuse. An alternative is work of Chehade *et al.* (2011).

We believe that presented approach gives an important contribution to the definition of real-time software, since it has a well-defined path to: (a) model and test PIMs placing emphasis in model size reduction, interoperability, portability, and reuse; (b) indicated alignment and reuse of transformations; (c) evaluation of non-functional properties (schedulability). These points contribute to the certification process. Moreover, it allows extensions as listed as future work without invalidating the approach proposed here.

This work is part of Brazilian National Institute for Space Research (INPE) research about modeling and verifying real-time software, which the main focus is to obtain a better balance between dependability, schedule, and cost. As it was stated by OMG (2003), MDA is a key initiative to reduce costs while it can increase dependability. Results presented in this paper reinforce this potential.

References

Buckl, C.; Gaponova, I.; Geisenger, M.; Knoll, A.; Lee, E. A. (2010). Model-Based Specification of Timing Requirements. In Proceedings… EMSOFT 2010 Proceedings of the tenth ACM international conference on Embedded software.

Burmester, S.; Giese, H.; Schafer,W.; (2005). Model-driven architecture for hard real-time systems: From platform independent models to code. In: Hartman, A., Kreische, D. (eds.) ECMDA-FA 2005. LNCS, vol. 3748, pp. 25–40. Springer, Heidelberg (2005).

Chehade, W. E. H.; Radermacher, A.; Terrier, F.; Selicy, B.; Gerard, S. (2011). A Model-Driven Framework for the Development of Portable Real-time Embedded Systems. In proceedings… 2011 16th IEEE International Conference on Engineering of Complex Computer Systems.

Eclipse Foundation (2011). Eclipse Site. Available at: <http://www.eclipse.org>. Accessed on: 29 June 2011.

Feiler, P.; Niz, D.; Raistrick, C.; Lewis, B. (2007). From PIMs to PSMs. In Proceedings... IEEE International Conference on Engineering Complex Computer Systems, n. 23, 2007, Auckland, New Zealand.

Giese, H.; Karsai, G.; Lee, E.A.; Rumpe, B.; Schätz, B. (2010). Model-Based Engineering of Embedded Real-Time Systems. International Dagstuhl Workshop, Dagstuhl Castle, Germany, November 4-9, 2007. ISBN 978-3-642-16276-3

Lazar, C.L.; Lazar, I.; Parv, B.; Motogna, S.; Czibula, G. (2009). Using a fUML Action Language to construct UML models . In proceedings… 2009 11th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing.

Lettner, M.; Tschernuth, M. (2010). Applied MDA for Embedded Devices: Software design and code generation for alow-cost mobile phone. In proceedings… 2010 34th Annual IEEE Computer Software and Applications Conference Workshops.

Lin, Chao-Sheng; Lu, Chun-Hsien; Lin, Shang-Wei; Chen, Yean-Ru; Hsiung, Pao-Ann. (2011) VERTAF/Multi-Core: A SysML-based application framework for multi-core embedded software development. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 26(3): 448–462 May 2011. DOI 0.1007/s11390-011-1146-3.

LISyC (2010). Cheddar Site. LISyC, Université de Brest. 2010. Available at: <http://beru.univ-brest.fr/~singhoff/cheddar/index-fr.html>. Accessed on: 29 June 2011.

Maes, E. (2007). Validation de systèmes temps-réel et embarqué à partir d'un modèle MARTE. France: Thales Research and Technology, 2007. 24 p.

ModelDriven. (2009). fUML Reference Implementation Site. ModelDriven. Available at: <http://www.modeldriven.org>. Accessed on: 29 June 2011.

Monperrus, M.; Champeau, J.; Hoeltzener, B. (2007). Counts count. In proceedings… 2nd workshop on Model Size Metrics Co-located with MODELS 2007 1 October 2007 Nashville, USA.

Moreira M. L. B. (2006). Design and simulation of a discrete controller for the Multi-Mission Platform and its migration to a real-time operational system. 2006. 181 p. (INPE-14202-TDI/1103). Master dissertation (Space Engineering and Technology-ETE, Option Space Mechanics and Control -CSE) - National Institute for Space Research (INPE), São José dos Campos, Brazil, 2006.

Obermaisser, R.; Kopetz, H. (2009). Genesys – A candidate for an ARTEMIS Cross-Domain Reference Architecture for Embedded Systems. 2009. Available at: <http://www.genesys-platform.eu/genesys_book.pdf> Accessed on: 17 May 2011.

OBJECT MANAGEMENT GROUP (OMG) (2003). Model-Driven Architecture. USA: OMG, 2003. 62 p. Available at: <http://www.omg.org/mda>. Accessed on: 17 May 2009.

OBJECT MANAGEMENT GROUP (OMG) (2008). Meta Object Facility (MOF 2.0) Query/View/Transformation Specification: Version 1.0. USA: OMG, 2008. 240 p. Available at: <http://www.omg.org/spec/QVT/1.0/>. Accessed on: 17 May 2011.

OBJECT MANAGEMENT GROUP (OMG) (2008b). MOF Model to Text Transformation Language: Version 1.0. USA: OMG, 2008. 48 p. Available at: <http://www.omg.org/spec/MOFM2T/1.0/>. Accessed on: 17 May 2011.

OBJECT MANAGEMENT GROUP (OMG) (2009). Unified Modeling Language, Superstructure: Version 2.2. USA: OMG, 2009. 740 p. Available at: <http://www.uml.org/>. Accessed on: 17 May 2011.

OBJECT MANAGEMENT GROUP (OMG) (2009a). Semantics of a Foundational Subset for Executable UML Models: Version FTF Beta 2. USA: OMG, 2009. 349 p. Available at: <http://www.omg.org/spec/FUML/>. Accessed on: 17 May 2011.

OBJECT MANAGEMENT GROUP (OMG) (2009c). UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded Systems: Version 1. USA: OMG, 2009. 738 p. Available at: <http://www.omgmarte.org/>. Accessed on: 17 May 2011.

OBJECT MANAGEMENT GROUP (OMG) (2011). Concrete Syntax for UML Action Language (Action Language for Foundational UML - ALF): Version: 2nd FTF. USA: OMG, 2011. Available at: <http://www.omg.org/spec/ALF/>. Accessed on: 24 April 2012.

OBJECT MANAGEMENT GROUP (OMG) (2012). UML Testing profile: Version: 1.1. USA: OMG, 2012. Available at: < http://www.omg.org/spec/UTP/1.1/PDF/>. Accessed on: 24 April 2012.

Romero, A. G. (2010). An approach to model-driven architecture applied to space embedded real-time software. Version: 2010-12-13. 203 p. Master dissertation (Space Engineering and Technology-ETE, Option Engineering and Management of Space Systems-CSE) - National Institute for Space Research (INPE), São José dos Campos, Brazil, 2010.

TOPCASED (2011). TOPCASED Site. Available at: <http://www.topcased.org>. Accessed on: 29 June 2011.

Wehrmeister, M. A. (2009). An aspect-oriented model-driven engineering approach for distributed embedded real-time systems. 2009. 206 p. Doctoral thesis – Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil. 2009.

Zuo, W.; Feng, J.; Zhang, J. (2010). Model Transformation from xUML PIMs to AADL PSMs. In proceedings… 2010 International Conference on Computing, Control and Industrial Engineering.