01 Sep 2006

A Methodology for Structured Object-Oriented Elicitation and Analysis of Temporal Constraints in Hardware/Software Co-Analysis and Co-Design of Real-Time Systems

Sun Yan

Xiaoqing Frank Liu
Missouri University of Science and Technology, fliu@mst.edu

Bruce M. McMillin
Missouri University of Science and Technology, ff@mst.edu

Recommended Citation

S. Yan et al., "A Methodology for Structured Object-Oriented Elicitation and Analysis of Temporal Constraints in Hardware/Software Co-Analysis and Co-Design of Real-Time Systems," *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06) (2006, Chicago, IL)*, vol. 1, Institute of Electrical and Electronics Engineers (IEEE), Sep 2006.
The definitive version is available at https://doi.org/10.1109/COMPSAC.2006.11

A Methodology for Structured Object-Oriented Elicitation and Analysis of Temporal Constraints in Hardware/Software Co-analysis and Co-design of Real-Time Systems

Yan Sun[*], Xiaoqing (Frank) Liu[*†], Bruce McMillin[*†]
Department of Computer Science
University of Missouri-Rolla
{ysgvd, fliu, ff}@umr.edu

Abstract

The hardware/software co-design of a high-quality real-time system relies heavily on the modeling of both the hardware and software components from three aspects: structures, functionalities, and constraints, especially the temporal constraints. However, there is not a systematic process for the elicitation and analysis of temporal constraints in hardware/software co-design. Furthermore, existing object-oriented methods provide no means for the explicit specification of system/component constraints in object models. This paper presents a systematic methodology for structured object-oriented analysis and specification of temporal constraints in hardware/software co-analysis and co-design using an extended High-Order Object-Oriented Modeling Technique (HOOMT). This methodology hierarchically elicits and analyzes the temporal constraints in hardware/software co-design based on the integration of the High-Order Object Model (HOOM) and Hierarchical Timed Automata (HTA). It helps to identify temporal constraints of hardware and software components as well as their interactions level by level. In addition, it helps trace the relationships among these constraints at multiple levels during the co-design of real-time systems. A case study from the hardware/software co-design of the simulated FACTS Power Transmission System is used to illustrate the feasibility and merits of this methodology.

Keywords

Temporal Constraints, Hardware/Software Co-analysis, Hardware/Software Co-design, Structured Object-Oriented Elicitation and Analysis, Hierarchical Decomposition.

1. Introduction

The design of a highly dependable real-time system relies heavily on the modeling of its 1) structure, 2) functionalities, and 3) constraints, especially its temporal constraints. It is desirable for designers to obtain the critical information that is relevant to either hardware or software components from all the three aspects in the object model. However, problems emerge during the hierarchical co-analysis and co-design of an advanced electric power transmission system [3] [15].

First, current object-oriented analysis and design techniques lack the notation to explicitly specify the temporal constraints of a real-time system in the object models, especially in the case of hardware/software co-analysis and co-design where the temporal constraints for hardware and software components as well as the interactions between them need to be clearly specified.

Second, very few object-oriented methods provide a built-in capability for the analysis of the relationships (such as traceability analysis) among elicited hardware and software temporal constraints at multiple abstract levels.

In order to integrate the relationship analysis of temporal constraints with the existing hierarchical hardware/software co-analysis and co-design of the advanced electric power transmission system, a process for the hierarchical elicitation and analysis of temporal constraints is needed.

Because traditional object-oriented modeling techniques do not provide a structured approach for the integrated analysis of object structure, hierarchical object-oriented techniques have been proposed [4] [7] [11] [13] [18]. However, these hierarchical OO techniques do not specify temporal constraints in the object models. In addition, built-in structural analysis of temporal constraints is not available in these techniques.

Timed automata is considered an appropriate technique for the modeling and verification of temporal behaviors. Khatib et al. proposed to transform existing temporal constraints into timed automata for model checking [12]. However, the issue of where to obtain these temporal constraints is not addressed.

Efforts have also been made to integrate object-

[*] Supported in part by NSF MRI grant CNS-0420869.

[†] Supported in part by the UMR Intelligent Systems Center.

orientation with timed automaton to capture and verify the temporal behavior of a real-time system. A common approach is to transform the OO dynamic models into timed automata for verification [1] [16] [17]. Similar to the object-oriented modeling techniques, it is difficult to model a complex system using a flattened network of timed automata. Thus, hierarchical modeling techniques, such as Hierarchical Timed Automaton (HTA), were proposed by researchers [2] [6] [14]. HTA is capable of capturing and verifying the temporal behaviors at multiple levels. However, there is not a systematic process for the hierarchical elicitation and analysis of temporal constraints based on the integration of HTA and hierarchical OO techniques.

In view of the above problems, a more systematic hierarchical process for the elicitation, analysis, and representation of temporal constraints for the hardware/software co-design of real-time systems is developed during the hardware/software co-analysis and co-design of an advanced electric power grid. The High-Order Object-Oriented Modeling Technique (HOOMT) [3] [13] [15] was extended to incorporate the temporal constraints in the co-analysis/co-design object models, and a hierarchical temporal constraints elicitation and analysis methodology was developed.

The remainder of this paper introduces the HOOMT extension and the hierarchical temporal constraints elicitation and analysis methodology as follows. Section 2 introduces the hardware/software co-analysis and co-design process from the perspective of temporal constraints. Section 3 presents a methodology for the hierarchical elicitation of hardware and/or software temporal constraints in hardware/software co-analysis and co-design. Section 4 illustrates the hierarchical elicitation methodology with an application example from the development of an advanced electric power transmission system. Section 5 contains the conclusion of this paper.

2. Co-analysis and Co-design Process from the Perspective of Temporal Constraints

The elicitation and influence of temporal constraints in a hardware/software co-analysis and co-design process is illustrated in Figure 1. The system objectives, which are derived from the voices of customers and the voices of developers, guide the development of the co-analysis and co-design models, including both the object models and the dynamic models. The co-analysis and co-design models can be decomposed in a hierarchical manner. As the co-analysis and co-design models become more detailed, their temporal behaviors can be refined accordingly. As a result, more specific temporal behaviors can be derived.

Temporal constraints are derived from temporal behaviors and the system functionalities, which are specified in the design models. Three types of temporal constraints can possibly be elicited: hardware temporal constraints, software temporal constraints, and constraints

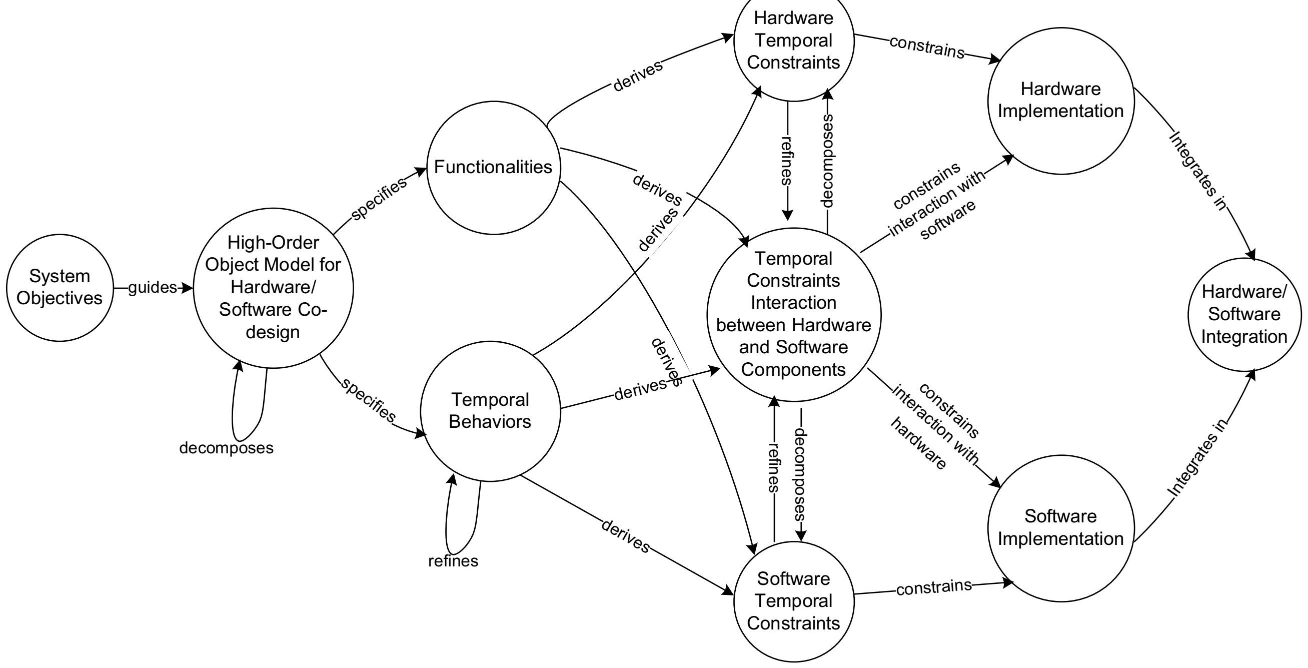

System Objectives
guides
High-Order Object Model for Hardware/ Software Co-design
decomposes
specifies
specifies
Functionalities
Temporal Behaviors
refines
derives
derives
derives
derives
derives
derives
Hardware Temporal Constraints
Temporal Constraints Interaction between Hardware and Software Components
Software Temporal Constraints
refines
decomposes
refines
decomposes
constrains
constrains interaction with software
constrains interaction with hardware
constrains
Hardware Implementation
Software Implementation
Integrates in
Integrates in
Hardware/ Software Integration

Figure 1. Hardware/Software Co-analysis and Co-design Process from the Perspective of Temporal Constraints

governing the interactions between hardware and software components. The constraints for the interactions between hardware and software components may be further decomposed into pure hardware or software constraints. Hardware and software constraints, on the other hand, can be used to refine the interactions between them.

The implementation of both hardware and software components needs to satisfy the temporal constraints elicited. It should be noted that the constraints governing hardware-software interactions must be considered during the implementation of both hardware and software components. Once the hardware and software components are implemented, they can be integrated into the whole system using any available traditional method.

In Figure 1, the left half shows the temporal constraints elicitation process and the right half shows how these temporal constraints, hardware and/or software, are used in the implementation phase. Because the focus of this paper is on the elicitation process, the next section will introduce, in correspondence to the hierarchical decomposition of the hardware/software co-analysis and co-design model, a hierarchical methodology for the elicitation of temporal constraints.

3. Methodology for Hierarchical Elicitation of Hardware/Software Temporal Constraints

The methodology introduced in this section follows the hierarchical hardware/software co-analysis and co-design process using HOOMT [13] [15]. To better integrate the temporal constraints into the existing HOOMT, this section presents the hierarchical elicitation and analysis methodology from three aspects. First, the methodology needs to represent the temporal constraints in HOOMT. Therefore, the High-Order Object Model (HOOM) in HOOMT is extended to include temporal constraints. Second, the hierarchical process of eliciting temporal constraints with the help of HOOM and HTA is introduced. Third, the methodology also helps to analyze the relationship among temporal constraints.

3.1. Incorporating Constraints in the Object-Oriented Model

In current object-oriented models such as Unified Modeling Language (UML) and HOOMT, the object models capture the object components as well as their attributes and methods. However, to a real-time system involving hardware and software components, the timing constraints are critical. Therefore, during the hardware/ software co-analysis and co-design of a real-time system, it is desirable to specify the hardware/software temporal constraints explicitly in the design model. Unfortunately, current modeling methods, such as UML, do not explicitly specify hardware/software temporal constraints in object models. Thus, as shown in Figure 2, HOOM is extended by introducing a constraints box, in addition to the existing attributes box and methods box, to the each of the objects in the model.

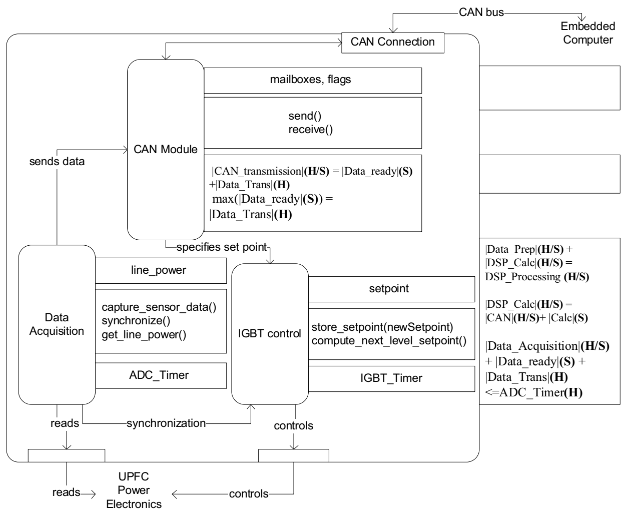

CAN bus
Embedded Computer
CAN Connection
mailboxes, flags
send()
receive()
CAN Module
sends data
|CAN_transmission|(H/S) = |Data_ready|(S) +|Data_Trans|(H)
max(|Data_ready|(S)) = |Data_Trans|(H)
specifies set point
|Data_Prep|(H/S) + |DSP_Calc|(H/S) = DSP_Processing (H/S)
|DSP_Calc|(H/S) = |CAN|(H/S)+ |Calc|(S)
|Data_Acquisition|(H/S) + |Data_ready|(S) + |Data_Trans|(H) <=ADC_Timer(H)
line_power
Data Acquisition
capture_sensor_data()
synchronize()
get_line_power()
ADC_Timer
IGBT control
setpoint
store_setpoint(newSetpoint)
compute_next_level_setpoint()
IGBT_Timer
reads
synchronization
controls
reads
UPFC Power Electronics
controls

Figure 2. High-Order Object Model with Constraints Box

Efforts have been made on the real-time expansion of object-oriented models such as the introduction of the timed sequence diagram and the timed state diagram [5] [8] [9] [10]. However, they have not been integrated with hierarchical object models. For verification purposes, they often are converted to timed automata [10].

Depending on the preference of hardware and software components designers, the temporal constraints can be specified in many formats. For instance, they can be specified either formally for verification purposes, such as using temporal logic, or informally for easier understanding, such as using natural language statements. In this methodology, an expression format is proposed for the purpose of elicitation and analysis of temporal constraints. For instance, a temporal constraint in the expression format is shown as follows:

$|CAN|(H/S) = Data\_ready|(S)+|CAN\_trans|(H)$

where $|CAN|$ is the duration of CAN state,

$|Data\_ready|$ is the duration of $Data\_ready$ state,

$|Data\_trans|$ is the duration of CAN_trans state,

(H), (S), and (H/S) following each of the variables specifies whether it is a hardware constraint, a software constraint, or a constraint on the interaction between hardware and software components.

The benefit of using these specification symbols in hardware/software co-analysis and co-design is that they make it easier for the designers of hardware and software components to divide their responsibilities.

The temporal constraints in the expression format as mentioned above can be transformed into temporal logics or natural language statements as needed. During the implementation of the hardware and software components, temporal assertions can also be derived from these constraints to monitor the dynamic behaviors of the hardware and software components in real time.

3.2. Hierarchical Elicitation of Temporal Constraints Based on HOOM and HTA

With the constraints box introduced to the objects in the object model, the next question is how to derive the temporal constraints, which can be entered in that box. Considering that HOOMT supports the object-oriented analysis and design of the system in a hierarchical manner, temporal constraints can be elicited with the help of HTA, which is also hierarchical. Below are the detailed steps in the elicitation process.

Step 1: The top-level object model is developed with the system objectives being its constraints. As shown in the first column in Figure 3, the object model contains both the object decomposition and the high-level temporal constraints. They serve as the basis for the generation of the states and state transitions in the HTA.

Step 2: For each level of the object model, one or more timed-automata can be developed. In case the state transitions are complex, the timed-automata can be further refined for more details. This is illustrated in the second column in Figure 3. The optional "refines" arrow denotes the possibility of HTA refinement.

Step 3: The timed-automata are analyzed to produce temporal constraints. As mentioned in the previous subsection, these temporal constraints are specified in the expression format, which is composed of variables and values as shown in the last two columns in Figure 3. These expressions can easily be inserted in the program code as safety assertions. They can also be transformed to temporal logics or natural language statements as needed.

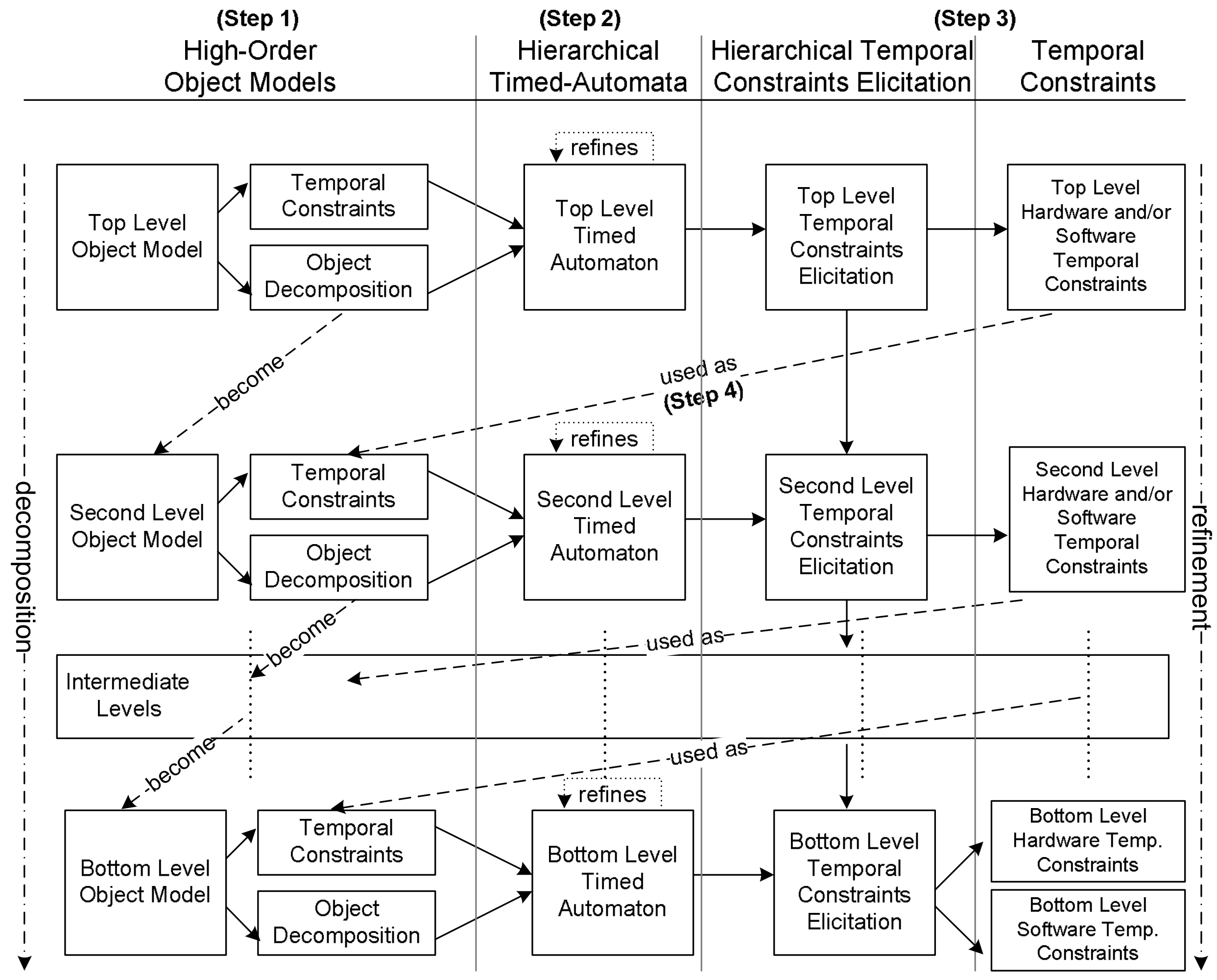

(Step 1)
High-Order Object Models
(Step 2)
Hierarchical Timed-Automata
(Step 3)
Hierarchical Temporal Constraints Elicitation
Temporal Constraints
refines
Top Level Object Model
Temporal Constraints
Object Decomposition
Top Level Timed Automaton
Top Level Temporal Constraints Elicitation
Top Level Hardware and/or Software Temporal Constraints
become
used as (Step 4)
Second Level Object Model
Second Level Timed Automaton
Second Level Temporal Constraints Elicitation
Second Level Hardware and/or Software Temporal Constraints
Intermediate Levels
used as
Bottom Level Object Model
Bottom Level Timed Automaton
Bottom Level Temporal Constraints Elicitation
Bottom Level Hardware Temp. Constraints
Bottom Level Software Temp. Constraints
decomposition
refinement

Figure 3. Hierarchical Elicitation of Temporal Constraints Based on HOOM and HTA

In each level of the hierarchical analysis, the temporal constraints may come from two sources: a) the decomposition of a constraint at a higher level, and b) the observation of the timed automata.

Because of the hierarchical characteristics of the object model and HTA, each of the constraint expressions needs to be examined for the possibility of decomposition. If a variable in an expression is not primitive, then it needs to be decomposed into new expressions consisting of variables relevant to the current-level object models and HTA.

It is also possible that the states and/or state transitions in the HTA reflect the interactions among the sub-objects in the object model. If the interactions are invisible to the upper-level HTA and are not reflected by any of the variables in the upper-level constraints, then there is no way to obtain them from decomposition. In this case, new temporal constraints are derived from the observation of the states and state transitions in the HTA, which reflect the interactions between the sub-objects. These new temporal constraints are only applicable to this particular level of object model and HTA.

As mentioned earlier, in hardware/software co-analysis and co-design, the temporal constraints need to be specified explicitly as either hardware constraints or software constraints or both. In this way, the tasks for hardware and software designers are clearly assigned. This is specified in the last column in Figure 3.

Step 4: Each of the sub-objects in the object model at this particular level is decomposed into a low-level object model. The temporal constraints derived at this level can be used as the temporal constraints of the corresponding sub-objects in the low-level object model (illustrated by the "used as" arrow in Figure 3). They are entered in the constraints box of the corresponding sub-objects.

Step 5: Steps 2 to 4 are repeated for the following levels of objects. This hierarchical process is repeated when the primitive objects are modeled and the primitive temporal constraints are derived.

3.3. Relationship Analysis of Temporal Constraints at Multiple Levels

Because this methodology for the elicitation of temporal constraints is hierarchical, the analysis of the relationships among the elicited temporal constraints is straightforward. A tree-like graph as shown in Figure 4 can be used to illustrate all the temporal constraints elicited using this methodology for better traceability.

The tree-like graph grows as the elicitation moves down the hierarchy, with each temporal constraint represented as a node. One-direction arrows (solid edges) are used to show the decomposition linkages. If a temporal constraint is not derived from the decomposition, then it has no parent node. However, it must have relationships with the other temporal constraints at the same level. This relationship is represented as double direction arrows (dotted edges). For instance, in Figure 4, the temporal constraints of TC 0.0.1 and 0.0.0.1 have no parents because they are not derived from the decomposition of any high-level temporal constraints. However, TC 0.0.1 is related to TC 1.1.2. Thus, they are linked with a dotted edge.

Using this graph, a primitive temporal constraint can be traced backward to its ancestors, which justifies its existence. For instance, in Figure 4, the temporal constraint of TC 1.1.2.1 can be justified by showing that it contributes to the satisfaction of TC 1.

On the other hand, a high-level temporal constraint can be traced down the tree-like structure to its lowest descendant. This shows the designers what needs to be achieved in order to satisfy this high-level constraint. Figure 4 shows that in order to satisfy TC 1.1, primitive TC 1.1.1, TC 1.1.2.1, and TC 1.1.2.2 must be satisfied.

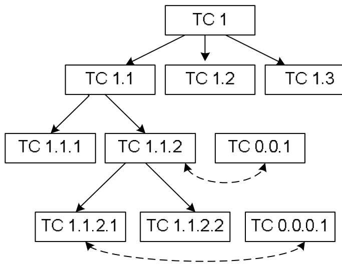


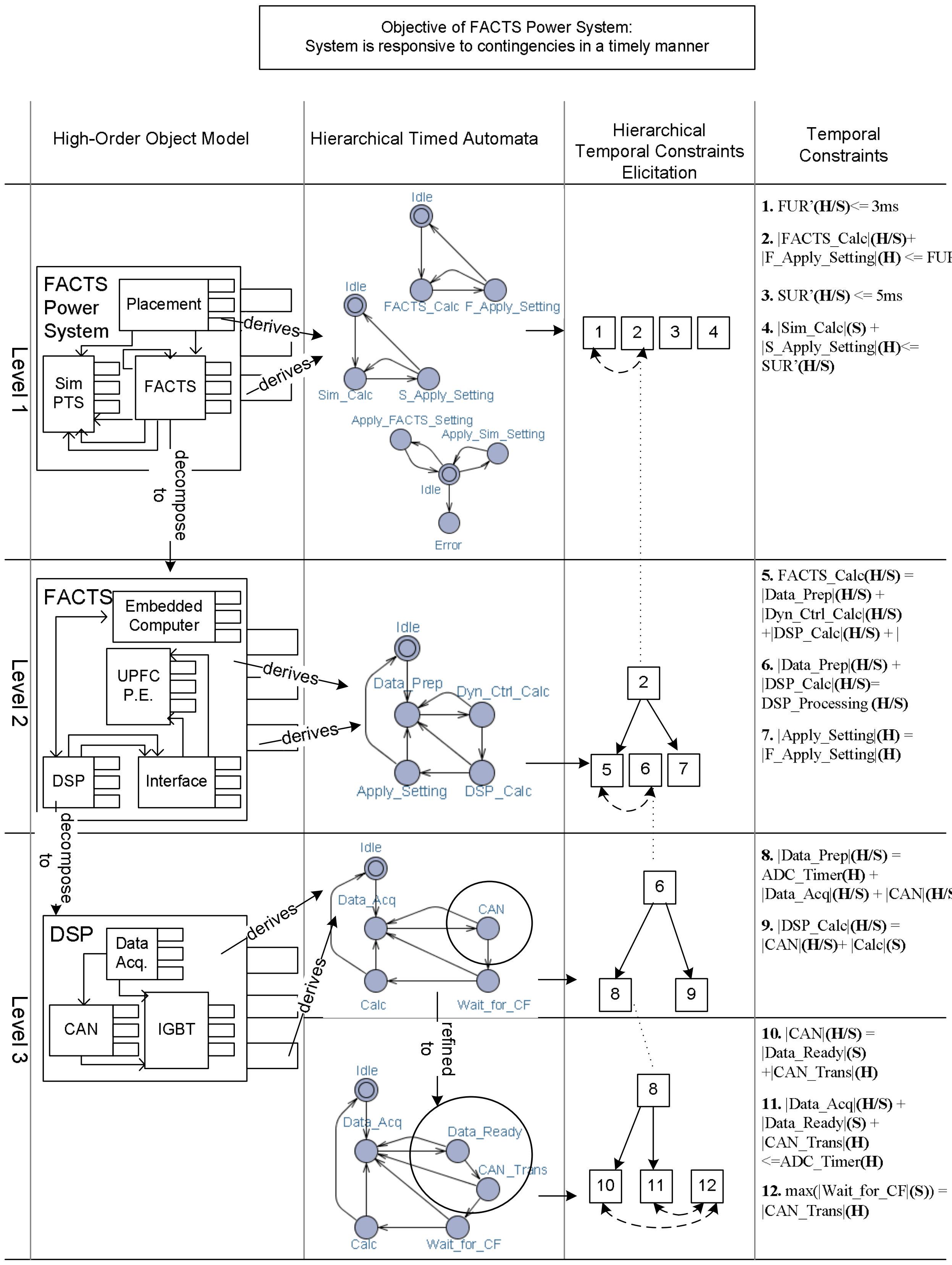
Figure 4. Temporal Constraints Relationship Analysis Graph

4. Application Example

Using the hierarchical temporal constraints elicitation and analysis methodology, a case study was conducted for the design of an advanced electric power grid, in which a number of FACTS (Flexible AC Transmission System) devices are used to monitor and adjust the power flows in a simulated power transmission system in real-time. The HOOMT is used for the hardware/software co-analysis and co-design of such a real-time system.

As introduced in Section 3, the HOOM is expanded to incorporate the temporal constraints. The temporal constraints are elicited hierarchically with the help of the object model and HTA. Due to the page limitation, only a portion of the results is shown in Figure 5. In order to show the process of the elicitation and analysis, the results are organized in the same format as shown in Figure 3.

The whole elicitation and analysis starts with the system objectives. As illustrated at the top of Figure 5, the objective of this advanced power grid is to be able to respond to contingencies in the system in a timely manner.



1. FUR'**(H/S)**<= 3ms

2. |FACTS_Calc|**(H/S)**+ |F_Apply_Setting|**(H)** <= FUR'

3. SUR'**(H/S)** <= 5ms

4. |Sim_Calc|**(S)** + |S_Apply_Setting|**(H)**<= SUR'**(H/S)**

5. FACTS_Calc**(H/S)** = |Data_Prep|**(H/S)** + |Dyn_Ctrl_Calc|**(H/S)** +|DSP_Calc|**(H/S)** + |

6. |Data_Prep|**(H/S)** + |DSP_Calc|**(H/S)**= DSP_Processing **(H/S)**

7. |Apply_Setting|**(H)** = |F_Apply_Setting|**(H)**

8. |Data_Prep|**(H/S)** = ADC_Timer**(H)** + |Data_Acq|**(H/S)** + |CAN|**(H/S)**

9. |DSP_Calc|**(H/S)** = |CAN|**(H/S)**+ |Calc|**(S)**

10. |CAN|**(H/S)** = |Data_Ready|**(S)** +|CAN_Trans|**(H)**

11. |Data_Acq|**(H/S)** + |Data_Ready|**(S)** + |CAN_Trans|**(H)** <=ADC_Timer**(H)**

12. max(|Wait_for_CF|**(S)**) = |CAN_Trans|**(H)**

Figure 5. Hierarchical Temporal Constraints Elicitation and Analysis for FACTS Power Transmission System

This main objective guides the whole hardware/software co-analysis and co-design. The object decomposition is shown in the first column of Figure 5. Starting at the top of the hierarchy, the whole simulated FACTS power system should satisfy the main objective. Thus, a set of timed automata is generated with the aim of satisfying this objective. Based on the analysis of the timed automaton, four temporal constraints are derived from Level 1. They are listed in the right column. For instance, in order to respond to contingencies in a timely manner, the FACTS device must produce new settings to the power line within a certain amount of time (3ms, denoted as FUR'). Within this amount of time, the FACTS device must finish the dynamic control calculation and apply the new settings to the power line. The above information is derived from both the object model (what the FACTS device does to the HIL power line) and the timed automata (how does the FACTS do this) and this information helps produce the first two temporal constraints.

In order to relate the responsibilities of the hardware and software designers to the temporal constraints, each of the constraint variables is marked with either *H* (hardware constraint), *S* (software constraint), or *H/S* (constraint involving both hardware and software components). An example of the object model with elicited temporal constraints entered in the constraints box is shown in Figure 2.

Because the object model at this level is abstract, each of its sub-objects is decomposed. In Figure 5, the FACTS device object is chosen to illustrate the decomposition.

Relevant temporal constraints (constraints 1, and 2) are selected from the four Level-1 constraints and entered in the constraints box of the FACTS object. Based on the Level-2 object structure and the relevant temporal constraints, a Level-2 timed automaton is developed and analyzed. Each of the temporal constraints of the FACTS object is examined. If any of the constraint variables is not primitive, then it is decomposed/refined to more detailed constraints. For instance, the variable of *FACTS_Calc* in Temporal Constraint 2 is not primitive because the calculation involves more than one components/states and its design cannot be assigned to only hardware developers or software developers. Thus, Temporal Constraint 2 is further decomposed to constraints 5 and 7. Again, the variables in the new temporal constraints are specified as either *H*, *S*, or *H/S*. The new constraints are entered in the constraints boxes of the corresponding sub-objects.

The same process is repeated for each level of the hierarchy. It should be noted that, as shown in Level 3 of Figure 5, for the same level object model, the timed automata can be refined to multiple levels for more details. The circled states in Figure 5 showed the refinement. Thus, the temporal constraints can also be refined to multiple levels.

Because each of the variables in the temporal constraints is marked with *H*, *S*, or *H/S*, these temporal constraints can easily be used in the implementation and testing of the development of software and hardware components.

As mentioned in Section 3, new temporal constraints can also be elicited from the observation of HTA instead of from the decomposition. In Figure 5, Temporal Constraint 6 is:

|Data_Prep|(H/S) + |DSP_Calc|(H/S)= DSP_Processing (H/S).

This temporal constraint is not derived from the decomposition of any upper-level constraints. Instead, based on the observation and understanding of the object structure and timed automata, two temporal constraints are combined into a new constraints. This is done to show that these two constraints are related to the DSP component and the designers of DSP should take care of these two constraints. Thus, instead of having a solid edge from an upper-level constraint, it is connected with dotted edges to temporal constraints 5 and 6.

After the temporal constraints are elicited, they can be justified by using the technique introduced in Section 3.3. The analysis graph is illustrated in the third column of Figure 5. The high-level temporal constraints are linked to their descendents, and the primitive temporal constraints can be linked back to their ancestors if available. If needed, these temporal constraints can be further specified formally and then verified.

5. Conclusion

The hardware/software co-analysis and co-design of a real-time system such as the advanced FACTS power transmission system needs the analysis and explicit specification of temporal constraints of hardware components, software components, and their interactions in a structured manner. However, there has been a lack of such a temporal constraints elicitation, analysis, and specification method in current object-oriented modeling techniques. This paper introduces a methodology for the elicitation and analysis of temporal constraints, thus making three contributions. (1) It extends the structured object-oriented modeling technique, HOOMT, by adding a constraints box to each of the objects in the object model. This allows the explicit specification of the temporal constraints corresponding to specific objects, which is beneficial to the hardware/software co-analysis and co-design of a real-time system. (2) This methodology integrates the hierarchical modeling techniques of HOOMT and HTA to provide a systematic process for the hierarchical elicitation of temporal constraints, which should result in more complete identification and specification of constraints. (3) It also presents a method for the relationship analysis among temporal constraints to improve traceability. This methodology follows a top-down approach by decomposing and refining the temporal constraints to more detailed ones relating to lower-level objects in the real-time system. This shows what need to be satisfied level by level in order to achieve the overall temporal objective. If necessary, these temporal constraints can also be verified from the bottom up to prove that by satisfying the primitive temporal constraints, the overall objectives can be met. This verification step will be a future extension of the methodology.

6. References

[1] Aaltonen, T., Katara, M., Pitkänen, R., "Verifying Real-time Joint Action Specifications Using Timed Automata," In Feng, Y., Notkin, D., Gaudel, M.C., eds.: *16th World Computer Congress 2000, Proceedings of Conference on Software: Theory and Practice*, Beijing, China, IFIP, Publishing House of Electronics Industry and International Federation for Information Processing (2000) 516–525.

[2] Tobias Amnell, Alexandre David, Elena Fersman, Oliver Möller, Paul Pettersson, and Wang Yi. "Tools for Real-Time UML: Formal Verification and Code Synthesis," in *Proceedings of the Workshop on Specification, Implementation and Validation of Object-oriented Embedded Systems (SIVOES'2001)*, 18-22 June 2001.

[3] Austin Armbruster, Matt Ryan, Frank Liu, Ying Cheng, and Bruce McMillin. "Hardware/Software Co-design for Power System Test Development," *Proceedings of the 2004 ACM Workshop on Interdisciplinary Software Engineering Research*, November 2004.

[4] Andrew R. Carmichael. "Defining Software Architectures Using the Hierarchical Object-Oriented Design method (HOOD)," *Proceedings of the conference on TRI-Ada '92*, December 1992.

[5] Werner Damm, Bernhard Josko, Hardi Hungar, and Amir Pnueli. "A Compositional Real-Time Semantics of STATEMATE Designs." In Willem-Paul de Roever, Hans Langmaack, and Amir Pnueli, editors, *Proc. Int. Symp. Compositionality (Revised Lectures), volume 1536 of Lect. Notes Comp. Sci.*, pages 186–238. Springer, Berlin, 1998.

[6] Alexandre David and Wang Yi, "Hierarchical Timed Automata for UPPAAL," *The 10th Nordic Workshop on Programming Theory (NWPT'98)*, Turku Centre for Computer Science (TUCS), Finland, October 14-16th, 1998.

[7] Thorsten Daum and Robert G. Sargent. "Scaling, Hierarchical Modeling, and Reuse in an Object-Oriented Modeling and Simulation System," in *Proceedings of the 31th Conference on Winter Simulation*. 1999.

[8] Bruce P. Douglass. *Real-Time UML*. Addison-Wesley, Reading, Mass., &c., 1998.

[9] Thomas Firley, Michaela Huhn, Karsten Diethers, Thomas Gehrke, and Ursula Goltz. "Timed Sequence Diagrams and Tool-Based Analysis - A Case Study." *The Second International Conference on the Unified Modeling Language, Beyond the Standard (UML'99)*. 1999.

[10] Alexander Knapp, Stephan Merz, and Christopher Rauh. "Model Checking Timed UML State Machines and Collaborations," *FTRTFT 2002, 7th International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems.* Oldenburg, Germany. Springer-Verlag, LNCS 2469, pp. 395-414. 2002.

[11] De Piante Henriksen, A. "Hierarchical, Object-Oriented Modeling and Simulation as a Tool for Managing the Manufacturing Enterprise," *PICMET '99. Portland International Conference on Management of Engineering and Technology, 1999. Technology and Innovation Management.* Volume 1, 25-29 Page(s):311 - 312 vol.1. July 1999.

[12] Lina Khatib, Nicola Muscettola, Klaus Havelund. "Mapping Temporal Planning Constraints into Timed Automata," *time*, *Eighth International Symposium on Temporal Representation and Reasoning (TIME'01)*, 2001.

[13] Xiaoqing Frank Liu, Hungwen Lin, and Lijun Dong, "High Order Object-Oriented Modeling Technique for Structured Object-Oriented Analysis," *International Journal of Computer and Information Science (IJCIS)*, Vol. 2, No. 2, pp. 74-96, June 2001.

[14] Claire Pagetti, Franck Cassez, and Olivier Roux. "Hierarchical Modeling and Verification of Timed Systems in Timed AltaRica," In *FACS 2003 the First Workshop on Formal Aspects of Component Software*, Pisa, Italy, 2003.

[15] Ryan, M.; Markose, S.; Xiaoging Liu; McMillin, B.; Ying Cheng. "Structured Object-Oriented Co-analysis/Co-design of Hardware/Software for the FACTS Power System," *COMPSAC 2005. 29th Annual International Computer Software and Applications Conference*, Volume 1, 26-28 July 2005 Page(s):396 - 402 Vol. 2. 2005.

[16] Guoqiang Shu, Chao Li, Qing Wang, Mingshu Li. "Validating Objected-Oriented Prototype of Real-Time Systems with Timed Automata," *rsp*, p. 99, *13th IEEE International Workshop on Rapid System Prototyping (RSP'02)*, 2002.

[17] Satoshi Yamane. "Object-Oriented Method for Real-Time Systems Based on Timed Automaton," *WORDS*, *p. 210, 2nd Workshop on Object-Oriented Real-Time Dependable Systems (WORDS '96)*, 1996.

[18] Zhu Xiaoguang, Wang Dongmu, Hong Bingrong. "An Object-Oriented Data Framework for Virtual Environments with Hierarchical Modeling," *ACM SIGSOFT Software Engineering Notes*, Volume 24 Issue 1, January 1999.