ABSTRACT
Since the structure of real time software is very complex, it must be designed clearly. In this growing complexity of real-time system, timeliness feature is a very important in such system. For instance, in hard real time system if the responses of events or messages are delayed, then the system might be dragged to a major problem by which all of system can be led to destroy. Because of that we should focus more on timeliness and schedulability analysis of embedded real time systems. Moreover, all real-time system elements should be firmly introduced in a model. We must choose an efficient model of which each element is equated to each element of real-time systems. Also model should depict timeliness and how to adapt UML to Schedulability Analysis of Real-time systems. In this paper, we compared two UML design models to choose which UML design model is suitable for adapting real-time nonfunctional requirements. In the comparison, we set two important criteria which are semantics and timeliness. In order to validate the criteria mentioned, a Robot Sensor Case Study is designed by using UML- SPT and UML- RT.

Keywords
UML-SPT, UML-RT, real-time, timeliness

1. INTRODUCTION
Real-time software systems consist of several tasks and concurrent processes, the tasks need to communicate and synchronize with each other and their environments. In a real-time system, events are unpredictable since there are so many concurrent tasks inputs from many sources because of that issue the real time systems structure is expected to provide the response fairly and fast. real-time software is critical systems that require high dependability and timeliness during the system operations. Moreover, a real-time system is anticipated to allocate the tasks equally to the available system sources. [1]

While the complexities of real time systems structure rise increasingly, fortunately the importance of design of such systems is also raise. For this purpose there are models used to ensure real time properties with reference to object-oriented paradigms such SDL, OCL and UML and so on.

The UML is the best standard Object-Oriented Modeling language for all software and it is widely used due to its extensibility mechanism, stereotypes, tag values and profiles. Besides, another important feature of UML is object paradigm, which support to cope with software complexity effectively.UML provides a rich visual notation to support all analysis and design activities.[2] There are some UML profiles tried to solve the real-time systems complexity. Most of them involved in some specific features which are proposed to fit UML into the real-time systems.

Comparison of the UML design models have been done by researchers. One of those comparisons is Gherbi and Khandek study. [3] They have compared and classified seven design models based on some Real-time criteria according to formal foundation, expressiveness and tool support.

Another research was studied by Bernardi and Petriu. They compared two UML profiles which are UML-SPT and UML-QoS based on the non functional requirements of software systems. They have focused on a list of key-points to compare between the profiles such as parameterizations capabilities, discrimination of the type of specification, specification of stochastic timings and related issues, annotation approach, annotation in the system model so on. At the end of comparative analysis shows that new concepts are needed in both profiles to express time intervals between two arbitrary events. Another problem is the parameterization of models, as in many cases fixed values for model parameters are not enough. UML-SPT profile goes a step further by supporting symbolic variables and expressions, but the UML-QoS profile does not have such a capability yet. [4].

In this paper, we have redesigned the existing case study in UML-SPT modeling design to UML-RT modeling design using Rational Rose Real Time tool.

The paper is organized as follows. Section 2 describes the UML-RT (Real-time UML). Section 3 presents UML-SPT design model. Section 4 compares briefly the advantages and disadvantages of the UML-SPT and UML-RT Profiles. Section 5 presents about our case study and comparisons of the design models. Finally, the conclusions are presented in Section 6.
2. Real-time UML

UML–RT (UML for Real Time) was produced by Rational Software which uses the UML built-in extensibility mechanism to integrate ROOM concepts within UML. [5] It also has been developed to address the complex, event-driven and potentially distributed real time systems. [1] A set of new concepts have been added to the UML which derived from ROOM model. UML-RT and Automatic Code Generation Tool from Object Time has already integrated into Rational Rose in which has created a product namely Rational Rose Real Time. It allows designers to automatically generate an executable program which consists of application code and run time system service libraries. [6] In addition, it supports the real time modeling constructs, which involve passive classes, capsules (also known as active classes) which involve ports and connectors and protocol classes.

The main element of the UML-RT is a capsule that is also known as an active object. Each capsule has one or more ports through which it communicates with other capsules. A capsule also may contain one or more sub capsules joined together by connectors. In addition, a capsule may have a state machine in order to send and receive message via the end ports of the capsule. As mentioned before, port of capsule provides communication with its environment and other capsules. There are two types of port, which either viewed from outside of the capsule or viewed within the capsule. For the later issue, the ports can be Relay Ports or End Ports. They provide two different internal connections that relay ports are connected to sub capsules and end ports are connected directly to the state machine of capsules. [7] On the other hand, a connector refers to the communication channel that provides transmission facilities to supports a particular signal-based protocol. Figure 1 shows a relationship between it and its environment as below:

![Figure 1. A capsule structure and communicate with its environment](image1)

Rational Rose Real Time produces capsule paradigm in order to provide the real time feature, which includes communication that uses asynchronous type. Capsule processes one message at a time until queue is empty. When the queue is empty, the object waits for another message. Each capsule also has its own structure, behavior and thread of control.

2.1. Structure Modeling

In UML-RT, the elements are shown together with the active objects or capsules. Moreover, capsules communicate to each other via ports and this communication involves of sending and receiving messages. Thus, capsule may have an internal structure and it can be specified by using the UML collaboration diagram. The structure diagram in the UML-RT specification depicts the relationship between capsules.

2.2. Behavior Modeling

Behavior is referring to the system’s response to external stimuli or any spontaneous actions. It specifies the sequence of states that an object goes through during its life time in response to events. [8] A capsule behavior is modeled with a Finite State Machine (FSM). Moreover, this state machine could be hierarchical because a state can be deployed into other Finite State Machines. As a result, a state diagram is used to depict the sequence of states in UML. An action may be associated with the transition either entry or exit of a state.

Rational Rose Real-time modeling consists of one or more threads, which an independent flow of control that permits a set of operations to execute in sequence. Usually each thread is assigned to one procedure for execution. When the messages are generated, they will be placed in the event queue of the destination object thread. A thread simply executes an event handling loop, processing one event at a time. [5]

3. UML-SPT

This profile was produced by OMG to capture timeliness, performance and schedulability properties of real time systems. The important feature of UML-SPT is it uses timeliness properties among different tools such as UML modeling tools and schedulability analysis tools. In addition, this profile does not invent any new technique, just gathers existing technique to capture about timeliness and other properties. [1] Moreover, this profile does not break the standard semantics of UML. This profile also aims to define some concepts for real time applications. These concepts compose three things, which are the stereotypes, tags, and constraints. Figure 2 illustrates us the structure of UML-SPT Profile.

The main advantage of UML-SPT is it provides us to standardize idioms design patterns common in a particular domain. However, the UML-SPT profile only provides the annotation timing and performance characteristics of the GRM sub profiles and performance analysis sub profiles.

![Figure 2: The structure of UML-SPT Profile](image2)
UML-SPT is a metamodel that uses the coherent set of UML stereotypes with associated tagged values which are used via constraints, possibly accompanied by a model library specific to the domain of interest. Since this profile is quite large, it can be divided into sub profiles. [7] Therefore, user can use the part that they need from the profile. As mentioned above, this profile consists of three main packages.

GRM (General Resource Model) is one of the packages which contain the basic concepts of real time systems. Moreover, GRM model is built on a classic client server model. The client requests services to be performed by the servers. [7] GRM can be divided into three parts of sub profiles as below:

i. **RTresource** Modeling consists of general concepts of the real time systems such as, events, actions, response, scenario, and execution engine and so on.

ii. **RTconcurrency** Modeling includes about concurrency modeling of actions, responses, and shareable resources.

iii. **RTtime** Modeling contains time-related mechanism for schedulability analysis. Indeed this modeling is a framework which usually uses some of elements which are clocks and timers to send triggers at a specified interval. [8]

Second package is analysis model which provides sub packages for different kinds of analytical method. Furthermore this type of modeling includes two sub profiles as follow:

i. **SAprofile** for schedulability analysis defines the stereotypes, tags, and constraints for various Schedulability Analyses. In addition, this sub profile ensures a framework to analyze the schedulability of real time situations. A numerous of tags and profiles can be used by **SAprofile** for Schedulability Analysis as shown in Table1. Figure 3 shows schedulability analysis metamodel.

ii. **PAprofile** defines common performance analysis methods. This sub profile is also concerned on annotating a model for computation of system performance. Moreover, this sub profile uses general concepts of concurrency and time sub profiles to create stereotypes and related tags that are very useful to add quantitative measures of system performance.

<table>
<thead>
<tr>
<th>Table 1. UML-SPT Common Stereotypes [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stereotype</strong></td>
</tr>
<tr>
<td>&lt;&lt;SASituation&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;SATrigger&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;SAResponse&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;SAAction&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;SASchedulable&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;SAResource&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;SAEngine&gt;&gt;</td>
</tr>
</tbody>
</table>

Figure 3. UML-SPT Schedulability Analysis Metamodel. [9]

4. Comparison of UML Support Tools

In this section, we present the comparison of UML-RT (Rose RT) and UML-SPT (Rhapsody) design models tools. Table 2 is structured based on the comparison of both design model tools. As shown in Table 2, UML-SPT introduces the quality of real time services requirements, resources and concept of timing by using stereotype to design the system. But UML-RT does not introduce clearly the timing constraints.

Table 2 indicates the type of system in which UML model can be modeled, UML-RT is useful for complex and event driven real-time system, while UML-SPT is much suitable for time driven (clock driven) systems, event driven (priority driven) systems and model driven systems. [10] As mentioned, since UML-RT is compatible for event driven real-time system, it has been used mostly in the telecommunication area where deadlines are more often soft than hard.

However, even UML-RT claims to be a Real-time modeling itself, it actually does not support schedulability analysis. [11]

UML-SPT is much suitable profile for users due to its easy modeling and annotating style. However, the complexity and inflexibility of the structure do not allow new user to define QoS feature and various analysis techniques. Following table (Table 2) illustrates some different criteria between UML-RT and UML-SPT. This table was created by having consideration the previous studies mentioned above.
Table 2: Shows different features between UML tools.

<table>
<thead>
<tr>
<th></th>
<th>UML-SPT</th>
<th>UML-RT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Issues</strong></td>
<td>Organization (OMG)</td>
<td>Industry (Rational Rose)</td>
</tr>
<tr>
<td><strong>Tool Support</strong></td>
<td>Rhapsody (iLogix)</td>
<td>RationalRT</td>
</tr>
<tr>
<td><strong>System</strong></td>
<td>Generic and real-time systems</td>
<td>Event-driven</td>
</tr>
<tr>
<td><strong>Schedulability</strong></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Expressiveness</strong></td>
<td>Active Class</td>
<td>Capsules</td>
</tr>
<tr>
<td></td>
<td>Port</td>
<td>Port</td>
</tr>
<tr>
<td></td>
<td>Interface</td>
<td>Protocol</td>
</tr>
<tr>
<td></td>
<td>Class</td>
<td>Class</td>
</tr>
<tr>
<td></td>
<td>Signal</td>
<td>Signal</td>
</tr>
<tr>
<td><strong>Full code generation</strong></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Simulation (GUI)</strong></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>RT Requirements</strong></td>
<td>Support</td>
<td>Not support</td>
</tr>
<tr>
<td><strong>Structural Diagram</strong></td>
<td>Object Model, Structure Diag.</td>
<td>Class, Capsule Structure Diag.</td>
</tr>
<tr>
<td><strong>UML Type</strong></td>
<td>UML 2.0 and UML 2.1</td>
<td>UML 1.4 x</td>
</tr>
</tbody>
</table>

5. Robot Sensor Case Study

The Robot Sensor case study is originally described by Klein, Lehoczky and Rajkumar. [12] It involves an application in which a robot system uses a distance sensor to measure the shape of pipes by moving around.

This application has a fiber-distributed data interface network with four processing nodes as shown in Figure 4. Nodes 1, 2, and 3 are dedicated to robotics processing, while Node 4 has an operator console for sending commands to the other three nodes and displaying their system data measurements. This system is required to meet each task’s deadline on each node.

For illustration purposes, this example considers schedulability only for Node 4, which performs a data display function. The display system in Node 4 contains three tasks $\tau_1$, $\tau_2$, and $\tau_3$ and to display data sent from the other three nodes. Tasks $\tau_1$ and $\tau_3$ share a common display device, while $\tau_2$ has a dedicated display device.

We use the following criteria for analysis purposes:

i. **Semantics**: to examine the semantics of object-oriented models for annotating real-time requirements.

ii. **Task priority and response time requirements**: To examine the models ability to illustrate the notation of task priorities for satisfying functional correctness and non functional response time requirements.

According to the first criteria, real time systems should react to periodic and asynchronous events. This is because any real time modeling language must be able to ensure semantics for representing objects that support processing of periodic and asynchronous events. Functional requirements of a real time system are performed by executing a set of tasks. Objects participate in the execution of a task. Therefore a real time representative model should provide support to aggregate objects into a set of tasks. Because Real-time systems need to satisfy the timing constraints on their functionality, the model should support transfer of control between threads. We compared the case study of Robot Sensor in UML-SPT and UML-RT by modeling it into several diagrams which are deployment, sequence, and class diagram. As seen on Figure 4 and Figure 5 there is no any difference to be noticed between these two profiles in deployment diagrams.

![Figure 4. Deployment diagram of fiber-distributed data interface network in UML-SPT [12]](image)

![Figure 5. Deployment diagram in UML-RT](image)
A Sequence diagram is a graphical view of a scenario that shows an object interaction in a time-based sequence. It establishes the roles of objects and help to provide essential information to determine class responsibilities and interfaces.

According to second criteria of examining the models ability on illustrate the notations of task priorities in order to satisfy functional correctness and non functional response time requirements, UML-SPT sequences diagram was used to illustrate those notation. This sequence diagram is shown in Figure 8. In UML-RT sequence diagram (Figure 9), these notations could also be shown but not exactly as UML-SPT. These notations can be illustrated with other diagrams such as collaboration diagram easily once the real-time requirement defined at design time. In addition, even these notations can be depicted by using notation but UML-RT does not support schedulability analysis.

As shown in Figure 8, in UML-SPT sequence diagram, real-time requirements can be analyzed while system execution.

Figure 6. Object Model Diagram for node 4 display processor specification. [12]

Figure 7. Class diagram of case study in UML-RT.

Figure 8. Sequence diagram on UML-SPT for displaying Node 1. [12]
6. CONCLUSION

In this paper, we have showed and evaluated not only UML models but also some design tools such as Rhapsody for UML-SPT and Rational Rose for UML-RT. Based on our experience, we can conclude that UML-SPT model using Rhapsody design tools is the most suitable and relevant tools for real-time software design due to provide real time nonfunctional properties such as timeliness and schedulability. After a few comparisons, we can state some advantages and disadvantages of both tools that been used. UML-RT does not support timing analysis but in UML-SPT there is a notation to support the timing analysis and performance of the system. Also both of UML tools simplify the design of real-time systems for users. In addition, these design tools can generate automatic code prior at implementation phase. Due to that issue, designer can detect the system defect earlier. However, as discussed earlier in this paper, UML-SPT design model can be used to design the hard real-time system easily.

For future research we will design our Autonomous Mobile Robot (AMR) case study by using UML-SPT design model and we will try to perform our aim as mentioned before to fulfill predictable design for Real-time software. For this purpose we will try to integrate choosen design model with schedulability analysis technique.

7. ACKNOWLEDGMENT

Special thanks to Mss. Sina ALI who helped me to revise and rearrange the paper and Zulkifli Zaki for their support, Ministry of Science, Technology and Innovation (MOSTI) Science Fund for the financing and funding and Universiti Teknologi Malaysia(UTM) for the facilities and infrastructures.

8. REFERENCES