Monitoring Data Types in Distributed Real-Time Systems*

Idriz Smaili and Peter Puschner
Real-Time Systems Group,
Vienna University of Technology, Vienna, Austria
{smaili,peter}@vmars.tuwien.ac.at

Abstract - In this paper monitoring data (MD) types are defined that enable the predicting of the influence of monitoring systems on the target systems. The amount of MD within an observation interval (even in the worst-case) and the gathering methods can be determined in advance before the monitoring process is started. The practical applicability of these concepts in monitoring of TTA systems is demonstrated.

I. INTRODUCTION

MD represent the run-time behavior of the target system at the intended abstraction level. The goal of monitoring systems is to collect all MD, from which the run-time behavior of the target system at the intended abstraction level can be reproduced. The correctness of real-time systems (RTS) depends not only on the results they deliver, but also on the point in time on which these results are delivered. Therefore, the key issue during monitoring of RTS is to keep the interference that is caused by a monitoring system on a RTS deterministic, if it cannot be completely removed. This interference depends on the way how MD are collected from the target system, and on the amount of collected MD. The interference of the monitoring system depends on the amount and on the approach used for gathering of monitoring data. Thus, the user of the monitoring system has to answer the following key questions before starting to monitor the target system:

- Which types of monitoring data must be collected at the intended abstraction level?
- What is the amount of these monitoring data?
- How can these monitoring data be gathered?

The classification of collected MD is influenced by abstraction levels at which target systems are monitored. In [2] the MD are classified into: i) hardware-, ii) process-, and iii) application-level events. A similar classification has been also presented by Tsai et.al. [9], Schuetz [5] presented an interesting classification in context of testing. He notes that the tester may wish to observe the input(s), intermediate variables (auxiliary output), the output(s), or all of them. Another detailed classification was presented by Thane in [3], in which the MD are categorized into three main groups: i) data flow - information on the data flow, ii) control flow - information on the control flow, and iii) resources - information on the resources of the target system. To our best knowledge, no approach exists in literature that can help the monitoring system to predict its influence on the target system in advance. To be able to predict this influence and to keep this influence deterministic we define MD types and present different gathering methods.

This paper is organized as follows. Terms and notions that are used throughout this paper are explained in Section II. The definition of MD types is presented in Section III. The methods for gathering of MD are dealt with in Section IV. A case study in which the practical applicability of this classification is presented is the focus of Section V.

II. TERMS AND NOTATIONS

In [1] Kopetz determines that a controlled object, e.g., a car, changes its state as a function of time, and the dynamics of a real-time application is modeled by a set of relevant state variables [2]. A significant state variable is called a real-time (RT) entity, and an observation of a RT entity is represented by a real-time (RT) image [1]. An observation is defined in [1] as information about the state of an RT entity at a particular instant of time.

A RT image is a mirror of a RT entity within the real-time controlling system (Figure 1) during the accuracy interval [1], after which the RT image becomes invalid. We conclude that the state of the physical controlled system at a particular instant can be described by the set of values of its RT entities at instant . The state of the controlled system at a particular instant is seen by the controlling system (i.e., the real-time computer system (RTCS)) can be described by the set of temporally accurate RT images. These states can be formally presented as:

\[ P_S = \{RTentity_{i,t} \mid i \in [1,N]\} \]

\[ C_P = \{RTimage_{i,t} \mid i \in [1,N]\} \]

where, \( P_S \) denotes the state of the physical (controlled) system and \( C_P \) denotes the state of the controlled system at the particular instant as seen by the RTCS. \( RTentity_{i,t} \) and \( RTimage_{i,t} \) denote the \( p \)th RT entity and its image at the instant , and \( N \) denotes the number of RT entities belonging to the system.

Analog to the notions presented above we found that the state of the target RTCS (i.e., the controlling system) that is monitored by the monitoring system can be modeled (from the monitoring system’s point of view) by the set of values of significant state variables, i.e., variables that are relevant for the monitoring system. Similar to the RT entities these significant variables are changed with the progression of physical time.
Analog to the RT entity, we introduce a new notion called real-time system (RTS) entity, which in fact is a significant “state variable” of the target real-time computer system and which is relevant only for the monitoring system. We say that the dynamics of the target RTCS from the monitoring system’s point of view can be modeled by a set of RTS entities. Examples of RTS entities are: number of tasks that are waiting for a semaphore, function ID that is generated by an instrumentation code during monitoring process, when a correlated function is called, etc. The difference between RT and RTS entities is:

- RT entities are used for modeling of the dynamics of the controlling and controlled system and they are used by the RTCS (i.e., target system) during its operation.
- RTS entities are used only for monitoring purposes. RTS entities together with RT entities are used for modeling of the dynamics of the target RTCS from the monitoring system’s point of view.

For deterministic reproducibility of the run-time behavior of the target system, (i.e., the real-time computer controlling and physical controlled system) the monitoring system has to collect both RT and RTS entities. Therefore, for monitoring purposes we introduce the notion monitoring entity (Figure 1), which contains either an RTS entity or an RT object (i.e., RT entity or RT image). Thus, every entity that must be observed by the monitoring system for representing the run-time behavior of the target system (i.e., the RTCS and its controlled physical environment) at the intended abstraction level, is called monitoring entity. In the rest of this paper the notions entity and monitoring entity will be used alternatively. From the monitoring system’s point of view the state of the target RTS at the intended abstraction level at a particular instant \( t \) can be modeled by the set of values of monitoring entities:

\[
T_5 = \{ \text{entity}_{i,j} \mid i \in [1,N] \}
\] (3)

where, \( T_5 \) denotes the state of the target system (as seen from the monitoring system’s point of view) at the instant \( t \).

The sequence of timely ordered observations that are made over a particular entity is called observation history:

\[
H(\text{OI}) = \{ \text{Obs}_i \mid i \in [1,K] \land K = f(\text{OI}) \}
\] (4)

where, \( K \) denotes the number of observations in the history, while \( f(\text{OI}) \) denotes the number of observations made within an observation interval (OI). The observation interval is the finite time interval, during which the target system is observed by the monitoring system. The set of observation histories that contain observations that are observed within an observation interval and which represent the run-time behavior of the monitored target system at the intended abstraction level are called monitoring data. The amount of MD within an observation interval is the sum of the timely ordered sets of observations, i.e., observation histories, over all entities:

\[
A_M(\text{OI}) = \sum_{i=1}^{N} H_i(\text{OI})
\] (5)

In Figure 2 the amount of monitoring data is presented, which contains the observation histories \( (H_{e_1}, e_2, \ldots, e_n) \) over different entities \( (e_1, e_2, \ldots, e_n) \).

### III. MONITORING DATA TYPES

The interference caused by the monitoring system on the target system during monitoring at the intended abstraction level depends:

- on the amount of MD that must be collected within an observation interval, and
- on the way how these MD are gathered.

#### A. Regular vs. Non-regular Monitoring Data

In order to make the interference of the monitoring system deterministic, the monitoring system must provide support to either exactly calculate or estimate the expected amount
of MD per time interval before the monitoring process of the target system starts. To achieve this, we define the regular and non-regular MD, depending on the rate at which the respective entities must be observed.

**Regular Monitoring Data.** We call MD regular, if there is a regular pattern of observations made over the respective entities. Examples of regular MD are: observations over an entity that represents the temperature of the engine and which is observed regularly, or observations over a message that is sent regularly over the shared transmission medium to other interconnected nodes. Another example of regular MD are observations over entities that are introduced within a particular periodic task in form of instrumentation code for monitoring of this task.

Regular MD consist of a set of regular state observations, which are defined in [2] as observations that record the state of state variables at particular instants, the point of observations. State observations produce a sequence of equidistant snapshots of the environment [2]. Thus, a basic characteristic of regular MD is their constant rate, at which they are generated, i.e., the respective entities are sampled. Therefore, for regular MD the respective amount within an observation interval is constant and it can be exactly calculated in advance.

**Non-regular Monitoring Data.** We call MD non-regular, if there is no regular pattern of observations made over the respective entities. Typical examples of non-regular MD are observations over entities (e.g., RTS entities) that are introduced within interrupt service routines in form of instrumentation code for their monitoring.

Non-regular MD consists of a set of event observations, which are defined in [2] as observations that contain the difference between the state before and after the event. A basic characteristic of non-regular MD is their non-constant rate, at which they are generated. Thus, the amount of these MD within an observation interval cannot be exactly calculated in advance, but it can only be estimated. In order to make the monitoring process deterministic, we must find out the amount of non-regular MD within an observation interval in the worst-case, i.e., the worst-case amount (WCA). In contrast to the regular MD the amount of non-regular MD is not constant within an observation interval, but it is bounded with the WCA \( A_{WCA}(OI) \) in Equation 6.

\[
A_M(OI) \leq A_{WCA}(OI)
\]

Depending on the fact, whether the \( A_{WCA} \) can be calculated in advance or only estimated (approximated), non-regular MD can be classified into:

**Sporadic Monitoring Data:** We call non-regular MD sporadic, if they contain observations over entities, which are not observed regularly, but a minimum time interval between successive observations exists and is known. \( A_{WCA} \) of these MD can be calculated in advance, if we suppose that the sporadic entities would be observed regularly with the period equal to the minimum time interval between two successive changes.

**Aperiodic Monitoring Data:** We call non-regular MD aperiodic, if they contain observations over entities, which are observed non-regularly, and no minimum time interval between successive observations exists. The calculation of \( A_{WCA} \) of these MD is impossible, because theoretically they can be observed arbitrarily often, i.e., at each instant of time. However, in target systems that use the sparse-time model for their global time, \( A_{WCA} \) of aperiodic MD can be calculated, if we suppose that the respective entities are observed only one time between two successive action lattices. In this case the aperiodic MD can be regarded as sporadic MD, which contain observations over entities that are observed with the minimum time interval between successive observations equal to the time interval between two action lattices.

Table 1 presents the classification of MD based on the amount, i.e., on the ability of the monitoring system to either calculate or estimate the expected amount of MD within an observation interval in advance.

### Table 1: Regular vs. Non-regular Monitoring Data

<table>
<thead>
<tr>
<th>Monitoring Data</th>
<th>Observations</th>
<th>Time Interval</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>State</td>
<td>Constant</td>
<td>Exact Calculation</td>
</tr>
<tr>
<td>Non-regular/Sporadic</td>
<td>Event</td>
<td>Minimum</td>
<td>Calculation of WCA$^*$</td>
</tr>
<tr>
<td>Non-regular/Aperiodic</td>
<td></td>
<td>No Minimum</td>
<td>No Calculation of WCA</td>
</tr>
</tbody>
</table>

$^*$These are typical examples of RTS entities, which are used only for monitoring purposes (see Section III).

$^1$In the sparse-time model the continuum of time is partitioned into an infinite sequence of alternating durations of activity and silence. From the point of view of temporal ordering, all events that occur within an interval of activity are considered to happen at the same time [1].
systems depends. To make this interference deterministic, we define the following MD types:

Pure Monitoring Data: We call MD pure monitoring data if they contain observations over entities that do not provide any contribution for achieving the computational goal of the target application. This type of entities, i.e., RTS entities (see Section II) are inserted into the target operating system or application in form of instrumentation code for gathering of run-time information for monitoring purposes only, and they are visible only within a particular node.

Monitored Application Data: We call MD monitored application data if they contain observations over entities that contribute to achieving the computational goal of the target application. Examples of monitored application data are: observations over an entity that represents the velocity of a car, or the operational mode of the real-time controlling computer system. The monitored application data can be classified into network and node MD depending on the scopes in which these observed entities are visible.

Network Monitoring Data. Network MD contain observations over entities, the observations of which are used by tasks running on different nodes, i.e., they are exchanged between interconnected nodes via the shared transmission medium. Examples of such MD are messages exchanged between nodes in a distributed RTS.

Node Monitoring Data. We call MD node MD if they contain observations over entities that are only visible within a particular node. Node MD are classified into local and global MD, depending on the scopes in which these entities are visible.

- Global Monitoring Data: We call the node MD global, if they contain observations over entities that are globally visible within a particular node. Examples of global data are: observations over inter-task (i.e., inter-process) messages, input values to the tasks (i.e., environment inputs that are processed, before they are correlated with other local data), output values of the tasks, which must be further processed before they are sent to another nodes, etc.

- Local Monitoring Data: Node monitoring data, which contain observations over entities that are not globally visible within a particular node are called local monitoring data. Examples of local data are: intermediate values or variables (auxiliary outputs), etc.

The difference between global and local MD is manifested during the gathering process, because different gathering methods must be used for gathering data of these two groups (see Section IV).

IV. GATHERING METHODS

We consider that a target system is a distributed embedded real-time computer system, and the monitoring approach used for gathering of monitoring information is either software or hybrid monitoring approach. Furthermore, the collection of MD, i.e., the gathering of observations, is done either by sensors or probes.

The classification of the gathering methods is influenced by the following three questions:

- Is the target system influenced by the monitoring system during the monitoring process?
- Is the gathering process of the MD transparent to the target application?
- Does the target application have to be instrumented for successfully gathering of MD?

In order to answer these questions we have classified the gathering methods into three groups:

- Monitoring-Node (MN),
- Operating-System (OS), and
- In-Line Gathering Method.

A. MN Gathering Method

The monitoring-node gathering method uses a dedicated monitoring node for gathering of MD. A monitoring node snoops on the transmission medium used for interconnection of nodes (Figure 3). With this method one can collect only the network MD, because they contain observations over entities whose values are exchanged over the shared transmission medium.

B. OS Gathering Method

The OS gathering method can be used for gathering of MD that contain observations over entities that are not transmitted over the shared communication medium but are visible within the global scope of the given node. These MD have been presented in the previous section. They are called global node or pure MD.

In this paper we will use the notions presented by Ogle et al. in [3]. Thus, sensors reside within the target application inserted during instrumentation process, while probes reside within the resident monitors that are parts of the monitoring system, and both of them are responsible for gathering of MD.

The target application is the software application that is running on the target system.
A TTA system consists of a set of nodes interconnected to each other by the replicated communication channels. The nodes access the replicated communication channels according to a Time-Division-Multiple-Access (TDMA) static communication scheme. Each node in the system has a unique transmission slot, which is specified in the static data structure of the TTP/C controller called Message Description List (MEDL) \[1\]. A node consists of a TTP/C communication controller, a dual-ported RAM called the Communication Network Interface (CNI) and a host controller. On the host of a particular node the static generated tasks schedule is executed by the real-time operating system (RTOS).

B. Monitoring System

The monitoring of distributed real-time systems (including TTA systems) can be done at the following abstraction levels:

Network Abstraction Level: At the network abstraction level the monitoring system can collect the network MD by using the monitoring-node gathering method. This way the target system is not influenced by the monitoring system neither in the temporal nor in the value domain. Such a monitor is implemented in the software product TTPview [6], which is successfully applied in the industry for monitoring of TTA systems.

OS Abstraction Level: At the OS abstraction level [7] observations over entities that are not visible outside the OS are collected. At the OS abstraction level the OS gathering method must be used for collecting of MD. The observations collected at the OS abstraction level can represent: i) activity of the OS's dispatcher, ii) state of the stack and its consumption by different tasks, iii) state of the internal data structures, etc.

Application Abstraction Level: At the application abstraction level those observations over entities must be collected from which the local part of the target application running on the host of the particular node can be reproduced. At this abstraction level the target system can be monitored at: i) task (TAL), or ii) function abstraction level (FAL) \[7\]. At the TAL the input/output entities of the tasks are observed. These observations are collected by means of the OS gathering method. In contrast to the TAL, at the FAL the entities that describe the inside of the tasks are observed. The MD gathering at the FAL is done by in-line gathering method, which means that the observed tasks must be instrumented.

Because of space limitations the description of this monitoring system is not presented in this paper in more detail. A detailed description can be found in [7]. In this case study the presented classification of MD is used for calculation of the amount of MD within an observation interval, which in case of TTA systems is the cluster cycle \[1\]. The instrumentation process is done manually.

C. Discussion

Figure 6 presents the classification of MD based on two different criteria: i) on the ability of the monitoring system to calculate the expected amount of MD within an observation interval in advance, and ii) on the gathering methods. The interference of the monitoring system on the target system depends on these two factors.

During monitoring of a target system at the user’s intended abstraction level the prediction of the expected amount of MD within an observation interval is of utmost importance. The amount of the MD that must be collected determines the resource needs of the monitoring system on the target system. These resources are used for collecting, processing, and

\[http://www.vmars.tuwien.ac.at/projects/tta/\]
transmitting the collected MD. They relate to: CPU, memory, communication bandwidth, etc. On the basis of this information one can "measure" the influence of the monitoring system on the target system.

On the other hand, the need for instrumentation of target systems depends on the used (and needed) gathering methods (Figure 6). The applied gathering method is another factor on which the influence caused by the monitoring system on target systems depends. For example, the MN method for gathering of MD does not cause any influence on target systems. However, target systems are influenced when OS or in-line methods are used.

Currently, the instrumentation process is done manually. The intention is to incorporate the classification of the MD presented in this paper into the design tools to calculate the needed resources (i.e., MD amount calculation) for monitoring process and to automate the instrumentation process. Examples of design tools are DECOMSYS::Designer⁷ or TTP-Plan⁸ used for designing of FlexRay or TTA systems. This approach could also be incorporated into design tools for other distributed RTS, e.g., CAN⁹, LIN¹⁰, etc.

VI. CONCLUSION

In this paper we have defined different types of MD and gathering methods. These make it possible to predict monitoring resources requirement in advance and to keep the interference of the monitoring system caused on the target system deterministic. Furthermore, we have presented a case study, in which we have successfully applied these concepts in monitoring of TTA systems.

VII. REFERENCES


