Information Models for Managing Monitoring Adaptation Enforcement

Audrey Moui, Thierry Desprats, Emmanuel Lavinal, Michelle Sibilla
IRIT, Université Paul Sabatier
118 route de Narbonne, 31062 Toulouse cedex 9, France
Email: {moui,desprats,lavinal,sibilla}@irit.fr

***Abstract*—Integrated management should cope with numerous, heterogeneous and complex systems in a multi-dimensional environment. In this context, the monitoring activity should be efficient and sensitive to variations of management applications requirements. In this paper, we define a framework which includes three capabilities that support monitoring adaptation. Particularly, we have defined information models for the first two capabilities, namely configurability and adaptability. This framework is modular enough to integrate any existing solution that proposes monitoring adaptation decisions. A partial CIM/WBEM implementation has been tested to measure the overhead due to the management cost of the proposed approach.**

***Keywords*—Monitoring; Adaptation; Information model; CIM/WBEM; Integrated management.**

I. INTRODUCTION

The multi-level aspect of integrated management (nodes, networks, systems, services), the dependence between heterogeneous components and the affluence of management information make the management activity more and more difficult to perform; indeed, new management paradigms are based on more and more autonomous and decentralized decision-making [14].

The efficiency of the monitoring activity has become a major preoccupation in various contexts such as integrated management of complex (networked) systems or autonomic and self-managed entities. In many cases, management paradigms are organized with the fundamental help of the classical MAPE loop (Monitor – Analyze – Plan – Execute) [7]. The managed system is observed (M) and an analysis (A) is performed to either detect or prevent failures or every relevant situations to be interpreted. In such a case, a reactive or proactive technical decision is taken and planned (P). Consequently, adjustment actions are then executed (E) on the system to hopefully improve its efficiency.

The quality of the control is provided by the quality of the implementation of each of these four functions, but also by the fullness of the interactions occurring between them. Particularly, at the first step of the cycle, the analysis function relies on the quality of the information (QoI) delivered by the monitoring mechanisms. One of the major issues is to improve this relationship by allowing a feedback from the analysis function to the monitoring one. This feedback constitutes a mean by which the analysis function, according to its management objectives, can continuously express requirements about the information and its quality, which are provided by the monitoring function. Dynamically taking into consideration the variations of these management functional requirements implies that the monitoring function has to be adaptive at runtime.

From an operational viewpoint, performing monitoring activities consumes several resources such as CPU, memory, energy, bandwidth, etc. This execution environment can also constrain the monitoring activities either directly, when resources are – partially or fully – temporally unavailable, or indirectly when resource consumption restrictions policies are applied (e.g., do not allow more than 10 % of global bandwidth for monitoring traffic). Consequently, the monitoring also needs to adapt itself to fit with any particular issues, which dynamically occur within the constrained operational environment.

Finally, a self-optimization concern may be supported by the monitoring function to increase the level of its efficiency. In this case, resulting from an introspective analysis of its performance, some self-adjusting decisions can be taken but enforced at the only condition that the monitoring function is adaptable.

This paper presents work performed to model the adaptation enforcement management, and a partial CIM/WBEM (Common Information Model [2] /Web-Based Enterprise Management [15]) implementation, which has been tested to measure the overhead due to the management cost of the proposed approach. It firstly presents our motivation and the global context of adaptation enforcement. The third and fourth sections present the modelling of the enforcement of monitoring strategy and the information models for adaptation enforcement management, respectively. The fifth section then presents the implemented prototype and the first measures, before concluding the paper.

II. MOTIVATION

For us, *adaptive monitoring* refers to the ability an online monitoring function has to decide and to enforce, without disruption, the adjustment of its behavior for maintaining its effectiveness, in respect of the variations of both functional requirements and operational constraints, and possibly for improving its efficiency according to self-optimization objectives.

 ISBN: 978-1-61208-219-6

A. Managing the Adaptation Enforcement

Automating monitoring adaptation requires to manage and to control the monitoring activity itself.

Independently of the deployment level, the monitoring activity is a process which relies on the gathering of raw, symbolic, aggregated, transformed or filtered data. Typically, gathering is based on the possibly combined use of "polling" (i.e., pulling data, by periodically requesting some targeted source or aggregator for data) and "event reporting" (i.e., receiving, in an ad hoc, sporadic or periodic way, pushed data from a provider or an aggregator) mechanisms.

Modifying the current behavior of a monitoring activity will finally result in achieving actions which will enforce adaptation decisions that aim at varying the scope and/or the modalities of the observations.

Reaching a CPU resource consumption reduction goal will generally lead to the decrease of the number of basic monitoring mechanisms which are running; at the opposite, trying to extend the scope of the monitoring will certainly cause new basic monitoring mechanisms to be launched (for example, giving the monitoring activity the possibility to question another managed element on the monitored underlying system gives an enlarged vision). Requiring a higher level of freshness of the collected data will cause the intensification of a polling frequency while limiting the monitoring traffic on the network will result in increasing the duration between two successive pull requests. For other purpose, like self-optimization, decision can be made to temporally suspend some particular mechanisms while considering some functional management needs (for example in a process of diagnostics refinement) will cause a temporal prolongation of a running mechanism. Illustrated by these previous typical situations, it can be concluded that, in the large majority of cases, the achievement of a monitoring adaptation will be materialized by managing the operational cycle of basic monitoring mechanisms and some of the parameters that can govern their behavior.

The main requirements to operate such a management of the enforcement of an adaptation decision include:

1) The need to capture any operational query about the adjustment of the behavior of the running monitoring activity. It implies to offer some well-defined interface allowing the decisional level to express actions to be done to the underlying enforcement infrastructure.
2) The necessity to verify at runtime the coherence and the feasibility of the requested actions. In particular, they can concern an updating of some settings governing the behavior of one or more running monitoring mechanisms. They can also affect the creation, the termination, or more generally, the operational state change of such a mechanism. Ensuring the consistency of the current state of the monitoring activity is another preoccupation. This must rely on the online availability of some information giving a management view of the state and the behavior of the monitoring mechanisms.
3) Finally, the ability to obtain some statistical data on each monitoring operations may serve for limitation resource consumption or self-optimization driven decisions making.

In this paper, we do not focus on the process of a monitoring adaptation decision making; we assume that it is based on any of well-known solutions including CSP, constraints resolution, inference engine, management policies, rules, bio-inspired approach, etc. In a complementary way, our concern is to focus more on the effective realization of the decided adaptation by providing a generic framework to support the management of this post-decision process.

B. Related Works

Numerous works, which are mainly dedicated to network monitoring, already contributed to enhance the monitoring adaptiveness level.

Duarte et al. [5] proposes a language for programming configurable monitoring applications, but which is dedicated to an SNMP/DISMAN environment; this facility is not generic enough to be used in every management context.

Massie et al. [9], Dilman et al. [3] and Moghé et al. [10] focus on finding protocol solutions and are mainly concerned with performance criteria (RAP [10] is only concerned with polling adaptability and Ganglia [9] provides the ability to reconfigure the monitoring mechanisms according to the newtork context, but only for small networks).

Some of the works have designed adaptive protocol-based solutions allowing trade-offs between performance and quality of the information [13], which exclusively relies on push data models. Other works target different area than newtork.

Xue et al. [16] deals with adaptive and scalable monitoring of cluster and propose facility for switching between a light and a heavy monitoring mode.

III. Modelling the Enforcement of Monitoring Strategy Adaptation

A. Preliminary definitions

- *Monitored Elements*
  Both polling and event reporting mechanisms are applied on system components.
  1) for polling mechanism, these components are called "targets" and can be seen as managed elements or sources of pulled data: these targets can be of different kinds:
     a) a whole object (e.g., a switch on a network) or a collection of objects (e.g., all the switches of a particular network);
     b) a property (e.g., the operational state of a(several) switch(es)) or a set of properties (e.g., any relevant information of a(several) switch(es));
  2) for event reporting, the components are considered as "sources" of pushed information or indications (e.g., warnings, alerts, particular data, pre-filtered or aggregated pushed information).

 ISBN: 978-1-61208-219-6

With concerns of genericity, we consider targets and sources as any relevant information related to managed elements. Let T be the set of these components within the scope of a monitoring activity, such as:

$$\forall n \in \mathbb{N}, \quad T = \langle t_1, t_2, ..., t_n \rangle$$

- *Basic Monitoring Mechanism*
  We can unify the polling and event reporting mechanisms (respectively named as P and E) as a set of basic monitoring mechanisms. This set is represented as M:
  $$M_T = P_T \cup E_T \neq \emptyset$$
- *Mechanism Configuration*
  Moreover, a basic monitoring mechanism must have to consider, initially and when changes are demanded on line, the values of a set of parameters which influence the way this mechanism operates in relation to its associated monitored elements. The set of these parameters will be represented as a configuration C, grouping two kinds of parameters: some are conceptually not modifiable, and the others can be modified.
- *Operational State*
  A mechanism owns an operational state Op which indicates if an active mechanism is currently $running$ or $pending$.

B. Monitoring Strategy

We define a *monitoring strategy* S as an association between the mechanisms M (polling or event reporting) applied on their respective targets, their configuration C and their operational state Op:

$$S = (M_T, C, Op) = \{\langle M_{t_1}, C_1, Op_1 \rangle, ..., \langle M_{t_n}, C_n, Op_n \rangle\}$$

C. Monitoring Strategy Adaptation

An adaptation will result in a monitoring strategy change: e.g., the modification of a temporal parameter value (for instance the polling periodicity value), the adjunction or the deletion of a target, the suspension or the deletion of the monitoring operation, etc.

A monitoring strategy change will consequently modify the monitoring activity behavior. The evolution from a strategy S to a strategy S' is defined as an *adaptation* δ_S [1].

$$\delta_S = S \rightarrow S'$$

An evolution of this strategy is then one of the following change (possibly combined) of the previously defined set (strategy formalization):

- The set of the mechanisms can be modified ($M \rightarrow M'$) by adding or deleting a monitoring mechanism M (polling or event reporting as well);
- The set of the targets can be modified ($T \rightarrow T'$) by adding or deleting a target T associated to a monitoring mechanism;
- The set of the configuration parameters can be modified ($C \rightarrow C'$) by updating the value of one or more configuration parameters C for a monitoring mechanism;
- The value of an operational parameter can be modified ($Op \rightarrow Op'$) by suspension or resumption of a monitoring mechanism.

These modifications have been explicited on Figure 3.

IV. Informational Models for Management of Adaptation Enforcement

A. Management Framework Overview

A particular environment has been defined to support the concepts of strategy and strategy adaptation. To do so, three capabilities have been determined as the requirements that a framework enforcing monitoring adaptation must ensure. These interconnected capabilities can be seen on Figure 1 and have been introduced in [12].

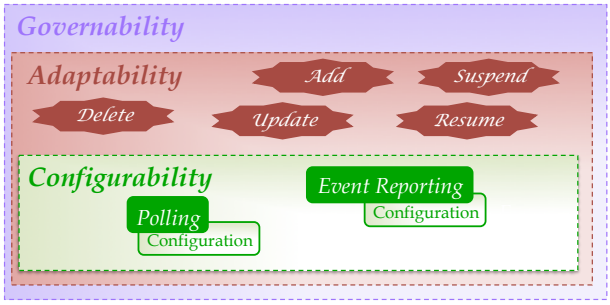

Fig. 1. Required Framework Capabilities

1) Configurability: The configurability is defined as the capacity of the monitoring mechanisms to be dynamically adapted: the configuration parameters governing the behavior of the monitoring mechanisms can be defined initially and then modified dynamically at runtime and without disruption when needed to obtain the best vision of the underlying system.

Defining a period parameter `PollingPeriod` for polling that we can modify at runtime is an example of configurability capability.

2) Adaptability: The adaptability is the ability to execute the monitoring adaptation. This capability is enforced by performing atomic adaptation operations over the underlying mechanisms configuration, with the objective to modify the current monitoring strategy. Consequently, this capability makes possible to dynamically modify at runtime the behavior of a monitoring activity.

An example of adaptability capability is to be able to dynamically modify the value of `PollingPeriod`.

3) Governability: Some intelligence is required to support the adaptation decision: the point is to decide if and how the monitoring activity has to be adjusted.

Piloting the adaptation operations to adjust and/or optimize the monitoring activity is the role of the governability capability. An adaptation decision can be taken in response to business objectives changes or for monitoring self-optimization, or in reaction to some constraints variations on technological resources availability.

This capability is an interface aiming at providing the inputs needed for the monitoring adaptation enforcement (the adaptation decisions). According to the previous example,

 ISBN: 978-1-61208-219-6

specifying the rule which will trigger the modification of the value of `PollingPeriod` is a concern of the governability level.

B. An Informational Model for Adaptability

We determined that a monitoring strategy is then a view at a particular moment of a set of monitoring mechanisms applied on their respective targets, their configuration parameters, their operational state, and optionally a set of statistical information data. Figure 2 sums up the concept of strategy.

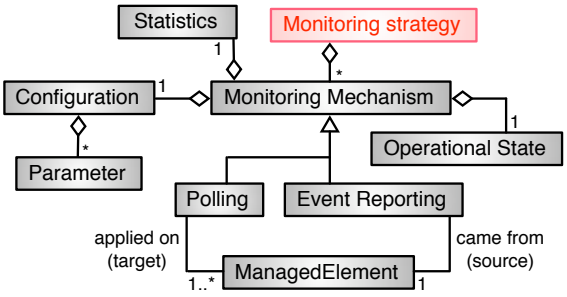

Fig. 2. Informational Model for Monitoring Strategy

Adaptability is needed to enforce the adaptation. Therefore, we have defined five basic operators, which may be identified as service interfaces which support at runtime the reconfiguration enforcement of the monitoring activities. These operators are atomic operations, which can also be combined as agregations.

1) *Adjunction*
   A mechanism, polling or event reporting, can be added by using the $\mathcal{A}$ operator: the configuration of the mechanism is required; the operational state will be automatically set to "active" (and the mechanism will be executed once created); and the managed element(s) on which the mechanism applies (target(s)) is(are) required.
2) *Deletion*
   A mechanism, polling or event reporting, can be deleted by using the $\mathcal{D}$ operator.
3) *Update*
   A configuration of a mechanism, polling or event reporting, can be updated at runtime, thanks to the $\mathcal{U}$ operator. This operator makes possible the modification of whole or part of its current configuration (all the modifiable parameters).
   Moreover, the list of the monitored elements can be modified: a target can be added to or removed from this list of targets at runtime and without disruption.
4) *Suspension*
   A polling mechanism can be used more than one time. Rather than deleting the mechanism, it can be relevant to suspend it, thanks to the $\mathcal{S}$ operator. Then the polling mechanism can be used again later if needed.
5) *Resumption*
   The resumption, performed with the $\mathcal{R}$ operator, corresponds to the reactivation of a previously suspended polling mechanism.

   Figure 3 defines the inputs and outputs of these operators, in order to clarify their operational use.

Algorithms have been writen to take into account every possible constraints at runtime.

- Two levels of parameters have been determined. Selector parameters define the global behavior of the mechanism (e.g., the stop mode describes the way a polling ends); according to the value of this parameter, additional parameters need to be filled to complete the behavior description (e.g., when the stop mode is set to "Iterative", the number of needed iterations has to be indicated). The algorithms check the coherence of the new value with these particularities;
- The new values for the parameters have to be validated: e.g., coherence of the new value and feasibility at runtime (for instance, the "MaxIteration" cannot be adjusted from 20 iterations to 10 iterations if the running polling has already performed 13 iterations), etc.

C. Informational Models for Configurability

The *polling configurability* and its modifiable and non-modifiable parameters have already been introduced in more details in [11] and the main ones are recalled in Figure 4.

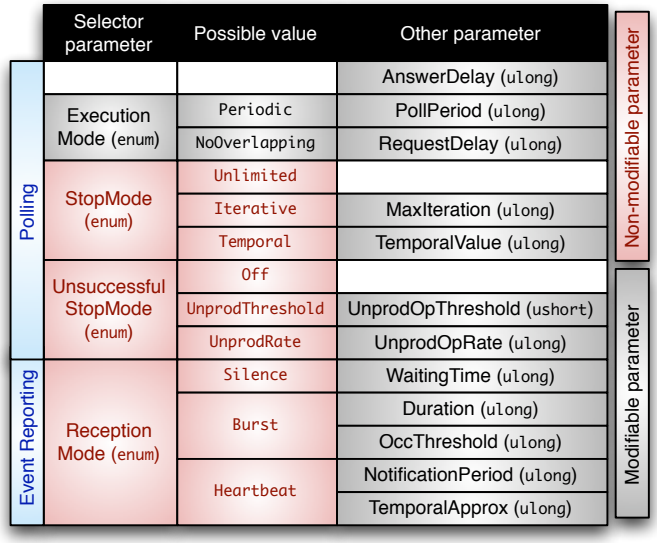

Fig. 4. Configuration Parameters for Monitoring Mechanisms

- The execution mode (`ExecutionMode`) selects the way the polling behaves: the polling operations can be launched either periodically, or in a non-overlapping mode: in the former case, the polling period (`PollPeriod`) parameter defines the periodicity; in the latter one, a time interval (`RequestDelay`) is needed to fix the time between the end of one operation and the beginning of the next one;
- The termination mode (`StopMode`) is the way the polling ends: the polling can indeed be unlimited or limited by a maximum number of occurrences (`MaxIteration`) or by a predefined duration (`TemporalValue`);

$$(M_T, C, Op) \xrightarrow{\mathcal{A}(m_{t_x}, c_x)} (M'_{T'}, C', Op') \text{ with } (M'_{T'}, C', Op') = (M_T, C, Op) \cup \{m_{t_x}, c_x, running\}$$

$$(M_T, C, Op) \xrightarrow{\mathcal{D}(m_{t_x}, c_x)} (M'_{T'}, C', Op') \text{ with } (M'_{T'}, C', Op') = (M_T, C, Op) - \{m_{t_x}, c_x, running\}$$

$$(M_T, C, Op) \xrightarrow{\mathcal{U}(m_{t_x}, c_x)} (M_T, C', Op) \text{ with } C \neq C'$$

$$(M_T, C, Op) \xrightarrow{\mathcal{S}(m_{t_x})} (M_T, C, Op') \text{ with } Op_x = pending$$

$$(M_T, C, Op) \xrightarrow{\mathcal{R}(m_{t_x})} (M_T, C, Op') \text{ with } Op_x = running$$

Fig. 3. Adaptability Operators: Formal Representation

- Let us consider that the polling is iteratively bounded: if a polling operation succeeds before the maximum waiting time (`AnswerDelay`) elapsed, the request is said productive; otherwise, when the answer is lost or delayed, then the request is said unproductive. An autonomous termination mode (`UnsuccessfulStopMode`) can be defined on an unproductive operation maximum rate (`UnprodOpRate`) or on a successive unproductive operation threshold (`UnprodOpThreshold`).

The *event reporting configurability* is more specific. According to some particular issues, it can be interesting to detect how the notifications or events are received (their *behavior*) independently of their *content*. Indeed, the notifications reception frequency can be relevant of any particular issue occurring on the managed system.

Therefore, the reception mode (`ReceptionMode`) parameter has been defined in order to select the management mode of the reception frequency of the notification operations. Three interesting situation profiles have been identified:

- No notification are received during a fixed time interval. This situation is called "silence", and can have two possible interpretations: the system works correctly and no warning or alert have to be raised; or, one (or more) notification(s) was(were) supposed to be received but did not make it, so an issue may have occurred on the system (e.g., the notifier may be on failure). An additional parameter is required to fix the maximum waiting time (`WaitingTime`) of a notification reception;
- Too many notifications are received in a fixed time interval. This behavior is called "burst" and can also be significant of a system issue. Two extra parameters have to be defined to manage it: the duration (`Duration`) on which the calculation is performed and the occurrence threshold (`OccThreshold`) which will trigger the detection of a sporadic behavior;
- The reception of notification operations can be periodic. This situation is called `heartbeat` and is defined with two additional parameters: the notification period (`NotificationPeriod`) and a temporal approximation (`TemporalApprox`) which allows a permitted temporal variation (e.g., due to traffic congestion).

Figure 4 makes a list of the reception mode parameters which describe how the notifications are received by the consumer. Additionally, other typical parameters should be defined to describe how notifications should be produced, filtered and delivered to the consumer when it subscribes to indications or pushed data.

V. Prototype Overview

In order to prove the feasibility of our approach and to measure the overhead due to the adaptation management (e.g., an increased execution time), we have implemented a prototype in an CIM/WBEM environment.

A. Technical Environment

CIM/WBEM are Distributed Management Task Force [4] standards which include a device and service management architecture, an object-oriented information model for describing any type of managed resource and a set of interfaces for accessing them [6]. In our adaptive monitoring service prototype, we provided a CIM representation of our configurability information models and we implemented the adaptability operators using WBEM interfaces. The CIM server is provided by OpenPegasus, an open source implementation of DMTF standards. As for the CIM clients, we used the Java SBLIM API.

1) CIM Server: The CIM/WBEM server is a data supplier and stores into a repository the monitored CIM targets instrumented via the integration modules (gateways between the standard or proprietary monitoring protocols and CIM). It also enables the storage and the manipulation of the configuration parameters included in the monitoring class. Finally, it enables the notifications of CIM indications toward the subscribers.

Therefore, the configurability level has been instrumented on a CIM server, which is executed on a virtual machine running Linux Ubuntu TLS 8.04 "Hardy Heron". The virtualization system is VirtualBox r73507.

2) CIM client: The CIM/WBEM client takes the configuration from the server initially and when a readjustment is achieved. It acts too as a WBEM listener and subscribes on the server in order to receive every CIM indications relative to an update of a CIM instance of the monitoring class.

The modification of the monitoring mechanisms (adaptability level) implies the use of a CIM client. It allows the dynamic management of the proposed CIM models by the adjunction of operators to see, modify, add and delete CIM objects and associations. The adaptability level is then running on the physical machine hosting the virtual machine. This physical machine runs Mac OS X 10.5.8 "Leopard".

 ISBN: 978-1-61208-219-6

B. Temporal Overhead due to Adaptation Management

All the presented first results have been obtained by computing the average of a hundred measures. The host machine runs the CIM client and the virtual machine runs the CIM server. The needed CIM objects (targets, polling) are the only objects registered onto the CIM repository. The polling mechanism is the only mechanism that has been measured so far. The measures lead to a preliminary analysis, which will have to be enhanced.

Figure 5 shows the execution duration (in milliseconds) needed for the execution of each adaptability operator in the case of a polling mechanism (the only mechanism measured so far). These measures are the following:

ADD	699 ms
DELETE	1004 ms
RESUME	353 ms
SUSPEND	110 ms
UPDATE	
– selector parameter	141 ms
– sub-parameter	193 ms
– global update	150 ms
– target adjunction	374 ms
– target deletion	304 ms

Fig. 5. Execution Duration for Adaptability Operators Execution

- ADD: the duration between the time when the $\mathcal{A}$ operator is invoked (in order to add a two targets polling operation) and the time when the polling operationally begins;
- SUSPEND: the duration between the time when the $\mathcal{S}$ operator is invoked (suspension of the polling operation) and the time when the polling is operationally suspended;
- RESUME: the duration between the time when the $\mathcal{R}$ operator is invoked (resumption of the previously suspended polling operation) and the time when the polling is operationally resumed (according to the context registered while it was suspended);
- UPDATE: the duration between the time when the $\mathcal{U}$ operator is invoked and the time when the polling takes into account the modification value. Five kinds of update are measured:
  - Selector parameter: the modification of the value of the `ExecutionMode` parameter;
  - Sub-parameter: the modification of any other modifiable configuration parameters. It implies a test on the relative selector parameter: e.g., the polling period can be modified only if the execution mode selector parameter is set to “Periodic”;
  - Global update: a global modification of the polling configuration: more than one parameter will be updated;
  - Target adjunction: the adjunction of a third target to the polling;
  - Target deletion: the deletion of the third target of the polling;
- DELETE: the duration between the time when the $\mathcal{D}$ operator is invoked (deletion of the two targets polling operation of the CIM server) and the time when the polling is totally removed from the CIM server.

As shown in Figure 5, the execution duration of a polling deletion is the longest execution duration (1004 ms), followed by the polling adjunction (699 ms). From this, it appears that the suspension and resumption operators are really interesting when a polling has been added: a combined suspension and resumption lasts 463 ms, while a combined deletion and adjunction lasts 1703 ms.

An adaptation at runtime can be enforced fastly:

- A selector parameter (the execution time) can be modified in 141 ms;
- A sub-parameter can be modified in 193 ms: the extra time is indeed due to a test over the selector parameter to check if the parameter can be modified for consistency ensuring purpose;
- A global update (the modification of the whole polling configuration) lasts 150 ms;
- The adjunction and deletion of a polling target lasts 374 ms and 304 ms, respectively.

VI. Conclusion and Future Work

This paper presented a generic monitoring framework that aims at enforcing the monitoring adaptation at runtime and without any disruption. When the adaptation is performed, the monitoring is made the less intrusive possible, by efficiently adjusting itself to every situation variation (business objectives changes, monitoring self-optimization, constraints variations on resources).

Following a control-oriented viewpoint, three capabilities have been identified within the framework: configurability, adaptability and governability. To ensure configurability, we have defined, for each type of basic mechanisms, generic configuration parameters: some are modifiable, the others are not. These specifications are independent of any management protocol; the configuration has been integrated on a CIM server. To ensure adaptability, the adaptation operators have been formalized and implemented on a CIM client thanks to the Java SBLIM API.

The resulting framework is a basis for the decision-making level as it is generic enough to support any solutions providing adaptation decisions.

Future works will tend now to integrate the governability capability to the presented framework. This framework is a basis which support monitoring adaptation enforcement. Integrating governability means controling the adaptation decision. A choice has to be done to determinate what is the best solution to enforce the governability capability, according to the underlying framework and needs. Moreover, we intend to test the complete prototype in a real managed environment.

References

[1] L. Chung and N. Subramanian, ”Adaptable Architecture Generation for Embedded Systems,” Journal of Systems and Software, 71(3), (2004), pp. 271–295.

 ISBN: 978-1-61208-219-6

[2] DMTF, "Common Information Model (CIM) Infrastructure" specification v.2.6.0 [online], (2010) [retrieved: May, 2012], http://www.dmtf.org/standards/cim.
[3] M. Dilman and D. Raz., "Efficient Reactive Monitoring," Infocom 2001, 2, (April 2001), pp. 1012–1019.
[4] DMTF Standards body, "Distributed Management Task Force, Inc." [online], (2012) [retrieved: May, 2012], http://www.dmtf.org.
[5] E.P. Duarte Jr., M.A. Musicante, and H.H. Fernandes, "ANEMONA: a Programming Language for Network Monitoring Applications," Internat. Journ. of Net. Managem., 18(4), (August 2008).
[6] Chris Hobbs: "A Practical Approach to WBEM/CIM Management," Auerbach, 2004, 344p.
[7] J.O. Kephart and D.M. Chess, "The Vision of Autonomic Computing," Computer, 36(1), (January 2003), pp. 41–50.
[8] A. Lahmadi, "Performances des fonctions et architectures de supervision de réseaux et de services," Doctorat de l'Université Nancy II, (Dec. 2007).
[9] M.L. Massie, B.N. Chun, and D.E. Culler, "The Ganglia Distributed Monitoring System: Design, Implementation and Experience," Parallel Computing, 30(7), (2004), pp. 817–840.
[10] P. Moghé, and M.H. Evangelista, "RAP – Rate Adaptive Polling for Network Management Applications," in Proceedings of IEEE NOMS'98, (1998), pp. 395–399.
[11] A. Moui, T. Desprats, E. Lavinal, and M. Sibilla, "Managing Polling Adaptability in a CIM/WBEM Infrastructure," Internat. DMTF Academic Alliance Workshop on Systems and Virtualization Management: Standards and New Technologies (SVM10), (2010), pp. 1–6.
[12] A. Moui, and T. Desprats, "Towards Self-Adaptive Monitoring Framework for Integrated Management," 5th IFIP Internat. Conference on Autonomous Infrastructure, Management and Security (AIMS 2011), (2011), pp. 160–163.
[13] A.G. Prieto, and R. Stadler, "Controlling Performance Trade-offs in Adaptive Network Monitoring," Proceedings of IM 2009, (2009), pp. 359–366.
[14] N. Samaan and N. Karmouch, "Towards Autonomic Network Management: an Analysis of Current and Future Research Directions," Communications Surveys & Tutorials, IEEE, 11(3), (2009), pp. 22–36.
[15] DMTF: "Web-Based Enterprise Management" specifications [online], (2012) [retrieved: May, 2012], http://www.dmtf.org/standards/wbem.
[16] Z. Xue, X. Dong, and W. Wu, "AOCMS: An Adaptive and Scalable Monitoring System for Large-Scale Clusters," Proc. APSCC, (2006), pp. 466–472.

 ISBN: 978-1-61208-219-6