# Self-inspection mechanisms for the support of autonomic decisions in Internet-based systems

Mauro Andreolini, Sara Casolari, Michele Colajanni

Department of Information Engineering
University of Modena and Reggio Emilia
{mauro.andreolini, sara.casolari, colajanni}@unimo.it

## Abstract

*Any autonomic system must implement mechanisms to automatically capture the most significant information about the internal state and also adapt the monitoring system to internal and external conditions. We refer to these activities as* self-inspection *and we consider them in the context of Internet-based services that are subject to workloads characterized by burst arrivals and heavy-tailed distributions. The large majority of the mechanisms driving these systems must take fast decisions on the basis of past and/or present load conditions of the system resources. In this context, self-inspection requires an adequate representation of the load behavior of the system resources that makes it possible to perform good actions under soft real-time constraints. In this paper, we show through a large set of experiments the need of basing load analyses and decisions on linear and non-linear models, such as the Exponential Moving Average and the 90-percentile models. All the considered models are applied to a multi-tier Web-based system that is instrumented with suitable self-inspection mechanisms at operating system level. However, the results can be extended to other Internet-based contexts where the systems are characterized by similar workload and resource behaviors.*

## I. Introduction

The advent of self-adaptive systems and autonomic computing [1]–[3] exploits the necessity for management algorithms that will take important decisions on the basis of a run-time evaluation of the load conditions of hardware and software system resources, especially oriented to load balancing and load sharing [4], overload and admission control [5], [6], job dispatching and redirection even at a geographical scale [7]. Self-adaptive systems seem an inevitable mean to manage the increasing complexity of networked information systems that have to satisfy scalability and availability requirements, and have to avoid performance degradation and system overload.

The ability of taking autonomous decisions according to some objective rules, for example, for event detection and for triggering actions concerning data/service placement, consistency, but also to detect overloaded or faulty components requires the ability of automatically capturing significant information about the internal state of the resources and also adapting the monitoring system to internal and external conditions.

In this paper, we focus on the following supports that are necessary to any runtime management system for self-adaptive applications: resource utilization monitoring mechanism, measurement and sampling, comparison of different models for extracting useful information from rough data. We refer to the previous activities as *self-inspection* that is, the ability of automatically capturing the most significant information about the internal state and also adapting the monitoring system to internal and external conditions.

The large majority of available algorithms and mechanisms for self-inspection evaluate the load conditions of a system through the periodic *sampling* of monitored raw data and use these values (or simple combinations of them) as a basis for determining the present system condition and any significant system change. While a measure offers an instantaneous view of the load conditions of a resource, it is of little help for distinguishing overload conditions from transient peaks, for understanding load trends and for anticipating future conditions, that are of utmost importance for taking correct decisions. These considerations are especially true when we consider self-adaptive supports for Internet-based services that receive a workload typically characterized by heavy-tailed distributions [8] and by flash crowds [9] that contribute to augment the skew of raw data.

Operating an Internet-based distributed system without accurate statistics is inappropriate. However, it is not easy to find the right combination between data sources providing low volume, coarse-grained, non-application specific data, and data sources providing high volume, fine-grained, and application specific information. These issues are even more complex in autonomic systems that are governed by some imposed service level agreement. Their policies should use system-wide and component status information to take the appropriate actions and to react to events, but this valuable information is not directly available. Distributed monitors usually yield raw, OS-level data (e.g., CPU and disk utilizations, network throughput) or application-level data (e.g., request throughput) that have to be aggregated to infer conclusions about the specific subsystem. Moreover, the resource measures obtained from load monitors of self-inspection mechanisms are extremely variable even at different time scales, and tend to become obsolete quickly [10]. Thus, long time measurement intervals reduce the effectiveness of the resource measures as suitable indicators of the real system conditions.

This paper demonstrate that in Internet-based contexts, self-inspection activities need an adequate "representation" of the load behavior of system resources that makes it possible to perform good (not necessarily optimal) actions under soft real-time constraints. Adequate means that in a heavy-tailed burst scenario we cannot use direct samples, or smoothing burst sample measures through simple solutions, such as arithmetic average or long measurement intervals.

The literature helps only partially, because conversion from multi-objective to single objective is often done by computing a weighted sum of the different metrics, as shown in [11], [12]. The logical relationships among features and the distinction of mandatory, desirable, and optional selection criteria are not incorporated in these early models. Even more sophisticated hierarchical models [13] do not capture and combine the dynamics of transient phenomena accurately. The large majority of statistical models provide off-line data analyses [14], [15].

Instead, we propose to base self-inspection systems on models that offer a better representation of the system conditions. In this paper we consider models that offer a better representation of the system conditions to implement self-inspection systems. Besides the traditional Simple Moving Average (SMA) model, we consider also the Exponential Moving Average (EMA) as an alternative linear model, and also the 90-percentile as an alternative non-linear model. All these models share the common trait that their computational complexity is compatible to the temporal constraints of run-time monitoring, evaluation and possible autonomic decisions. Our results confirm that the precision of the models largely influences the quality of the self-inspection in terms of an improved ability to detect non-transient load changes. We can also anticipate that a non-linear model, such as the 90-percentile, is the best scheme to extract from many raw data "the information" that is really valuable for taking appropriate and quick actions in the context of Internet-based systems. This model gives even better results than the EMA model that, in its turn, is preferable to the SMA model. The proposed models are applied to a multi-tier Web-based system, but the results can be extended to other Internet-based contexts, where the systems are characterized by similar workloads and resource behaviors.

The paper is organized as follows. Section II contains a motivation for this paper by showing the extreme variability of resource samples in a prototype Internet-based system that is subject to realistic Web workload. Section III proposes and analyzes different models that can be used to get some useful information from raw load samples. In Section IV illustrates the results of a self-inspection mechanism based on different models and evaluates the two main parameters of interest, that is, the model precision and reactivity. Section V concludes the paper with some final remarks.

## II. Motivations

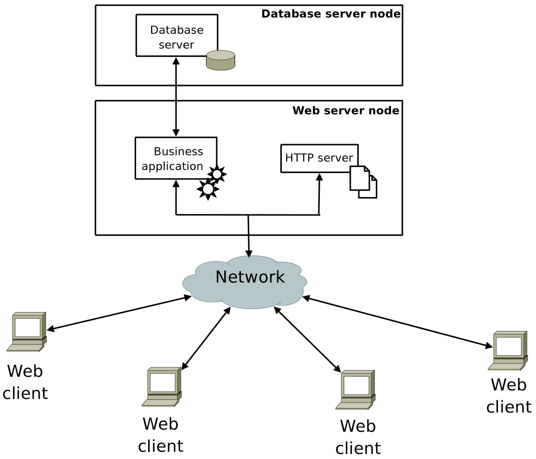
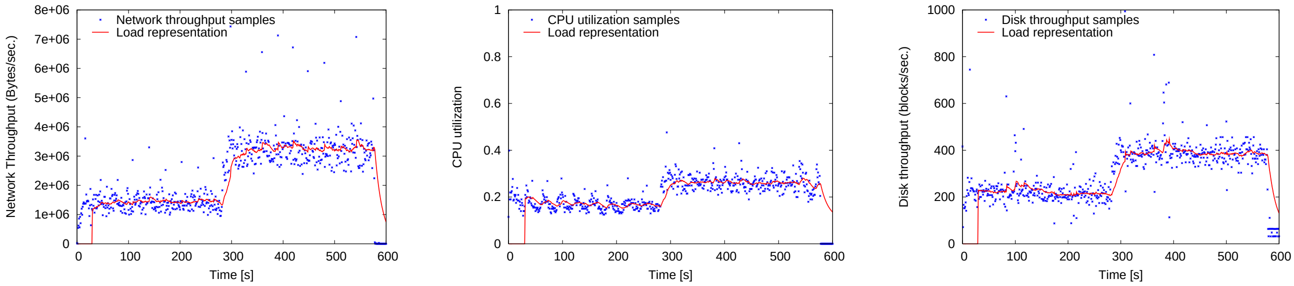


**Fig. 1. Architecture of the prototype**

There are many critical resources in any component of an Internet-based system. The resource load or status can be measured through common system monitors that typically yield instantaneous measures at regular time intervals. We have carried out a very large set of experiments for analyzing the typical behavior of commonly measured resources, such as CPU utilization, disk and network throughput, number of open sockets, number of

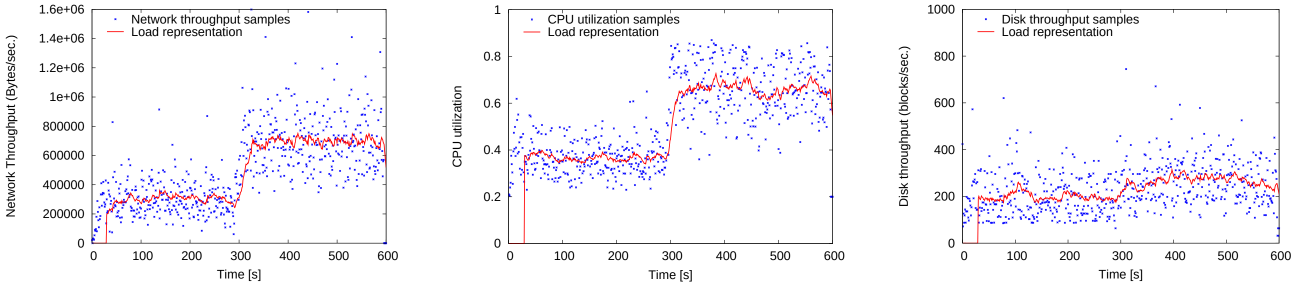


**(a) - network throughput**

**(b) - CPU utilization**

**(c) - disk throughput**

**Fig. 2. Resource samples - Application Server**



**(a) - network throughput**

**(b) - CPU utilization**

**(c) - disk throughput**

**Fig. 3. Resource samples -DB Server**

open files, process load, amount of used main memory. As a test-bed example, we consider a dynamic Web-based system referring to a typical multi-tier logical architecture (Figure 1) that is based on the implementation presented in [16].

The workload refers to the TPC-W model that is becoming a de facto standard for the performance evaluation of Web-based systems (e.g., [16]). Requests are generated through a set of *emulated browsers*, where each browser is implemented as a Java thread that emulates an entire user session with the Web site. We focus on a specific workload scenario that emulates a sudden change in the workload intensity, from a relatively unloaded to a more loaded system. The population is kept at 300 emulated browsers for 5 minutes, then it is suddenly increased to 700 emulated browsers for other 5 minutes.

To illustrate the difficulties in extracting useful information from raw performance samples, in Figures 2 and 3 we report the behavior of some popular resource usage metrics (CPU utilization, disk utilization, network throughput) for the application and the database server, when the system is subject to the TPC-W workload scenario. Besides the single samples, we also report the *load representation* curve that is based on a smoothed exponential average and that shows a cleaner trend of the resource behavior. All these figures show that the view of a resource obtained from system monitors is extremely variable to the extent that any run-time decision based on these values may be risky when not completely wrong. For example, let us consider a system that must take different decisions depending on the load of a CPU. When the CPU utilization measurements are similar to those in Figure 3(b), any load change detector would alternate frequent on-off alarms, thus making it impossible to a run-time decision system to judge whether a node is really off-loaded or not. On the other hand, a simple average of the samples would mitigate the on-off effects, but at the expenses of the efficacy of the load change detection algorithm.

To prove the degree of the variability of the samples and their temporal dependence, we compute some statistics on the resource measures behavior when the system is subject to the TPC-W workload.

The autocorrelation functions of the CPU utilization, disk throughput and bandwidth on the database server of the multi-tier architecture suggest *long-range dependent* processes because the values of the Hurst parameter ($H_C = 0.65$, $H_D = 0.63$ and $H_N = 0.61$, respectively) are all $> 0.5$ [17]. This means that the present behavior of resource measures depends on past history.

We analyze the cross-correlation between the emulated load and the resource measures through the Pearson product-moment correlation coefficient $P$. We find that $P_C = 0.80$, $P_D = 0.32$ and $P_N = 0.72$ for the CPU utilization, the disk throughput and the net throughput,

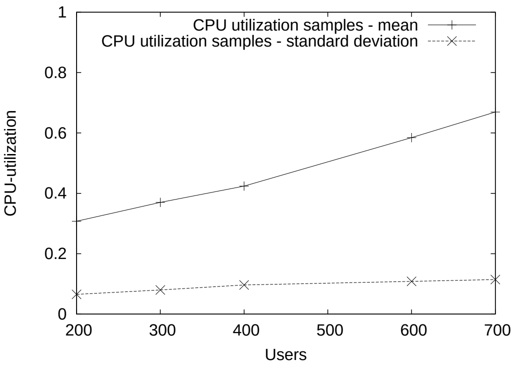


**Fig. 4. Statistical properties of CPU utilization samples with varying workload intensities**

respectively. Values of the Pearson coefficient close to 1 imply a temporal relationship between the imposed load and the utilization of system resources, that can be used to infer a more representative view of the system behavior [18].

We show how simple load aggregations can yield a consistent trend with varying workload intensities. We focus on the CPU utilization, because it exhibits the highest temporal dependence. Figure 4 shows the sample mean and standard deviation of the CPU utilization as a function of the number of emulated browsers. The sample mean evidences a clear linear dependence with the number of emulated users. As a consequence, even a very simple aggregation strategy (in this case, an average over the last 10 minutes) can point out differences in the load trend. On the other hand, the dispersion of the CPU utilization samples (measured by the standard deviation) remains almost constant. In other words, a load aggregation strategy can capture the behavioral trend with the same precision at different workload intensities. All these considerations seem to suggest that resource measure aggregations are a valid mean for load representation and that even simple aggregations of resource measures can be robust with respect to the offered workload, which is a crucial feature of a self-inspection system.

## III. Load representation

The variability of single resource samples must be reduced through load aggregation techniques. We first consider the class of *moving averages* as linear representative loads, because they smooth out resource measures, reduce the effect of out-of-scale values, are fairly easy to compute at run-time, and are commonly used as trend indicators [19]. We focus on two classes of moving average: the *Simple Moving Average* (SMA) and the *Exponential Moving Average* (EMA) that use uniform and non-uniform weighted distributions of the past samples, respectively. We also consider the *90-percentile* (P90) of past resource samples as a non-linear model. As we are interested to run-time models in a context of highly variable systems, we cannot consider other popular linear auto-regressive models, such as ARMA and ARIMA [14], [20], because in the considered scenarios they would require frequent updates of their parameters. These operations are computationally too expensive and inadequate to support run-time decision systems. For these reasons, the auto-regressive models find better applications when they are created off-line after examination of all available data, or are applied to workloads that are characterized by low variability and high auto-correlation of load measures.

We describe the models by supposing that the last resource sample $s_i$ and a set of previously collected $n-1$ samples $s_{i-1}, \ldots, s_{i-n+1}$ are available at time $t_i$.

**Simple Moving Average** (SMA). It is the unweighted mean of the past $n$ resource measures, evaluated at time $t_i$ $(i > n)$:

$$SMA_{i,n} = \frac{\sum\limits_{i-(n-1)\leq j\leq i} s_j}{n} \tag{1}$$

An SMA-based load representation evaluates a new value for each sample $s_i$ during the period of observation. The number of considered resource measures is a parameter of the SMA model, hence hereafter we use SMA$_n$ to denote an SMA-based representative load based on $n$ samples. As SMA models assign an equal weight to every resource measure, they introduce a significant delay in the trend representation, especially when the number of considered samples $n$ increases. The EMA models are often considered with the purpose of limiting this delay.

**Exponential Moving Average** (EMA). It is the weighted mean of the past $n$ resource measures, where the weights decrease exponentially. An EMA-based load representation, at time $t_i$, is equal to:

$$EMA_{i,n} = \alpha * s_i + (1-\alpha) * EMA_{i-1,n} \tag{2}$$

where the parameter $\alpha = 2/(n+1)$ is the *smoothing factor*. The initial value is the average of the first $n$ measures:

$$EMA_{n+1,n} = \frac{\sum\limits_{0\leq j\leq n} s_j}{n} \tag{3}$$

Similarly to the SMA model, the number of considered resource measures is a parameter of the EMA model, hence by EMA$_n$ we denote an EMA-based representative load based on $n$ samples.

**P-Percentile** (P$_p$). Given an ordered series of values $x_1, \ldots, x_n$ (with $x_1 \leq \cdots \leq x_n$), a value $x_j$ $(j \in [1, \ldots, n])$ is said to be the p-Percentile if:

$$\begin{cases} \text{at most } (p*n)/100 \text{ samples } x_i \leq x_j, \\ \text{at most } (100-p)*n/100 \text{ samples } x_i \geq x_j. \end{cases} \tag{4}$$

In descriptive statistics, the percentile is a common way of estimating proportions of the data that should fall above and below a given value. In this paper, we will be using the 90-Percentile (denoted as P90).

## IV. Performance evaluation

Two properties characterize the quality of a self-inspection system: the rapidity in signaling a significant change of the system state, and the ability to discern a steady change from a transient change. These two properties are conflicting, because a self-inspection system that is able to quickly signal state changes, has also higher chances of interpreting a transient spike as a steady change.

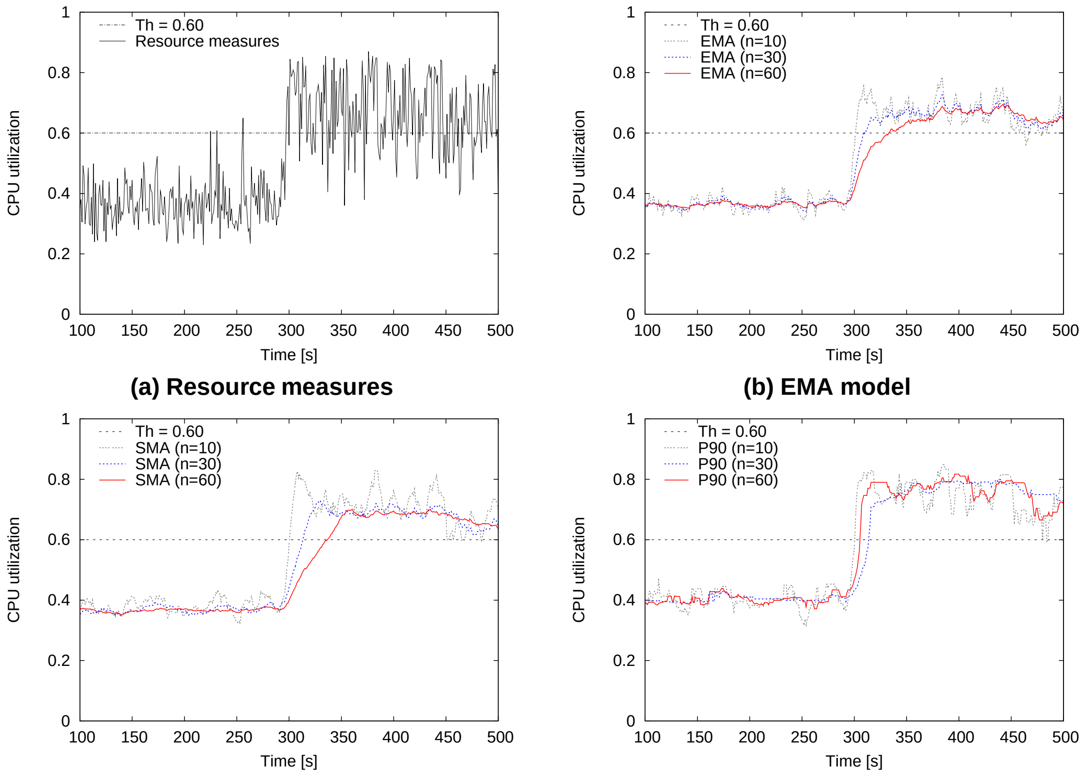
As a qualifying example, we consider the scenario described in Section II, where we have seen that the CPU utilization is the most representative measure of the system state. The self-inspection system signals a change of state to the run-time decision system whenever it detects that the CPU utilization passes over or under an *alarm threshold*, that for example is set to $Z = 0.6$. It is important to observe that the following results are representative of a large set of experiments that we do not report for limited space reasons. Figure 5 (a) shows the problems that affect a self-inspection system that is based on resource measures. Figures 5 (b-d) consider the same example when the EMA, SMA and P90 models are adopted. There is only one significant state change at 300 seconds, however a self-inspection system based on samples signal many other (false) alarms. In the other three cases, the number of false alarms tends to zero, however we can see that for some models the alarm comes with certain delays. The *accuracy* of the self-inspection mechanism is evaluated through two possible sources of false detections:

- **Reactivity error**. The excess of oscillations causes many false alarms (*false positive alarms*). This type of errors is extremely evident in the case of samples (Figure 5(a)), but also in the case of a too reactive self-inspection mechanism, such as $\text{EMA}_{10}$,$\text{SMA}_{10}$ and $\text{P90}_{10}$ (Figure 5(b-d)).
- **Delay error**. Excessive smoothing tends to cause delays in signaling even an alarm of significant variation of the state conditions (*false negative alarms*). This type of error is evident in the case of smoothed models, such as $\text{EMA}_{60}$ and $\text{SMA}_{60}$ (Figure 5(b-c)): both them signal the load change state occurring at $t = 300$ with a delay of about 40 seconds.

The Figures 5 represent the load behavior for different sample vector sizes. All the considered models have the advantage of monotonic and relatively stable results: their delay increases, and their reactivity decreases as a function of $n$. Hence, it is necessary to find a value of $n$ that represents a good trade-off between reduced delays and limited reactivity. The EMA and SMA linear models representations (in Figures 5(b) and (c), respectively) are more sensitive to the choice of $n$ than the representation based on the non linear model, reported in the Figure 5(d). The linear load representation based on short-term moving averages, that is, working on a small set of load measures $n$ (e.g., $\text{SMA}_{10}$, $\text{EMA}_{10}$), are more responsive to variations of the load conditions, but they are penalized by increased oscillations. On the other hand, long-term moving averages (e.g., $\text{SMA}_{60}$, $\text{EMA}_{60}$) are more accurate, because they smooth out all minor fluctuations, and tend to show only long-term trends [21], but at the price of low responsiveness. In particular, as the linear load representation based on SMA assigns an equal weight to every resource measure, the choice of the considered $n$ measures implies a clear trade-off. On the other hand, the EMA models limit the SMA delay effects in the representation of the load trend because they give a higher contribution to the last resource measures. From a cross comparison between Figure 5(b) and (c) we can see that an EMA-based load representation is able to signal resource load changes rather quickly and its oscillations are rather smoothed.

Although the EMA-based load representation improves the precision of the SMA-based load representation, both them tend to achieve small oscillations for a high number of samples, but this accuracy is combined with low responsiveness. The non linear load representations (P90), shown in Figure 5 (d), reduce these delay effects without incrementing the sample vector size $n$, as for the linear models. This P90 model is able to combine the two effects of high reactivity and low delay.

The Table I summarizes the results for several load representations by distinguishing the false indications due to delays (false negative alarms) and to reactivity (false positive alarms). When the self-inspection mechanism is based on resource measures, there is a significant number of oscillations around the threshold. In this case, the reactivity errors are the only contributions to false positive alarms, because there are no delays. The self-inspection systems that are based on EMA, SMA and P90 models exhibit a delay error that increases as a function of the number of samples $n$. This error represents the main contribution to false detections, because linear load representations seem smoothed enough to avoid reactivity errors, unless $n$ is too small (e.g, $n = 10$). An overall evaluation of the results in Table I shows that the best self-inspection systems are based on $\text{P90}_{10}$ and $\text{P90}_{30}$ models. Both these load detectors are characterized by low percentages of false detections, that are always lower than 5%. This is confirmed even by other results referring to scenarios that are not reported in this table. It is also important to analyze the two different errors that contribute to the good results of the 90-percentile models.



**(a) Resource measures**

**(b) EMA model**

**(c) SMA model**

**(d) 90-percentile model**

**Fig. 5. Results obtained by different models for self-inspection**

**TABLE I. Percentages of false alarms**

| | **Delay Error** | **Reactivity Error** | **Total Error** |
|---|---|---|---|
| **Resource Measures** | 0% | 17.75% | **17.75%** |
| **EMA$_{10}$** | 0.20% | 2.35% | **2.55%** |
| **EMA$_{30}$** | 2.20% | 0.20% | **2.40%** |
| **EMA$_{60}$** | 8.25% | 0% | **8.25%** |
| **SMA$_{10}$** | 0.25% | 2.40 | **2.65%** |
| **SMA$_{30}$** | 3.50% | 0% | **3.50%** |
| **SMA$_{60}$** | 9.25% | 0% | **9.25%** |
| **P90$_{10}$** | 0% | 1% | **1%** |
| **P90$_{30}$** | 1.6% | 0% | **1.6%** |
| **P90$_{60}$** | 4% | 0% | **4%** |

When large aggregations of measures (i.e., $n >= 10$) are adopted, the error is almost entirely due to delays. For small aggregations (i.e., $n < 15$), the main contribution is due to excessive reactivity. We can conclude that for the autonomic systems that require immediate signals, a self-inspection mechanism based on P90$_{10}$ is the best choice. Alternatively, a self-inspection mechanism based on P90$_{30}$ is preferable when the run-time management system has not to be disturbed by an excessive amounts of false alarms.

As a final result, we think it is important to evaluate the sensitivity of the best self-inspection systems to the alarm threshold values. In Figure 6 we report as example the percentage of false detections scenario as a function of different alarm thresholds in the interval $[30, 75]$. The goal is to show the stability of the results even in the most critical cases of a threshold value that is close to the average load reaching the system. From this figure we have that, for all alarm threshold values, the percentage of false alarms of the self-inspection system based on models are much lower than the corresponding alarms when the self-inspection system is based on resource measures. It is an important result that the two self-inspection systems based on EMA$_{30}$ and P90$_{30}$ never cause more than 10% of false detections even in the most critical cases. Moreover, we can confirm that the P90 model is preferable to the best linear model. Let us summarize the overall contributions of this study:

- The size of the sample vector plays an important role in the accuracy of load representation and, consequently, on the efficacy of the self-inspection system.
- Among linear and non-linear models, the 90-percentile models are by far the most accurate, with percentages of false alarms always below 5%. In particular, the autonomic systems that require im-

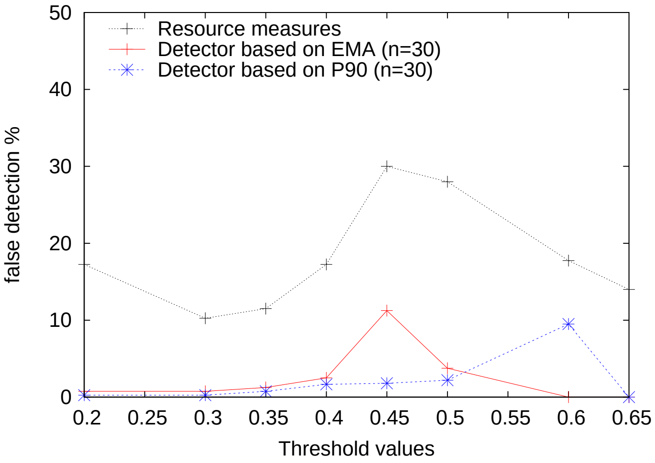


**Fig. 6. False detection errors as a function of the alarm threshold**

mediate signals could benefit from a self-inspection mechanism based on $P90_{10}$. Alternatively, the $P90_{30}$ load representation offers very good accuracy, and is recommended when the run-time management system does not have to be perturbed by false alarms.

- The most accurate self-inspection mechanisms based on $EMA_{30}$ and $P90_{30}$ models preserve their accuracy for a wide range of threshold values. Guaranteeing a low number of false alarms independently of the signaling threshold is one of the most desirable property that ensures stability of the self-* system independently of its main control parameters.

## V. Conclusions

Self-inspection mechanisms for evaluating the state of system resources and for detecting non-transient changes of load conditions are at the basis of the large majority of autonomic systems. Many run-time decision systems evaluate load conditions of system resources through the periodic sampling of monitored raw data. In the context of Internet-based applications subject to burst arrivals and heavy-tailed workloads, the direct use of monitored raw data or simple averages of sample values is not applicable to self-inspection problems, because the measures obtained from resource monitors are extremely variable and subject to staleness [10].

We demonstrate that simple linear models, such as SMA, do not work well, whereas more complex linear models, such as EMA that gives a different weight to the samples, support self-inspection mechanisms much better. However, both of them are surpassed by a non-linear model, such as the 90-percentile, that seems the most appropriate basis for autonomic systems subject to heavy-tailed workload and burst arrivals, that are typical of Internet-based systems.

## References

[1] J. O. Kephart and D. M. Chess, “The vision of Autonomic Computing,” *IEEE Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

[2] A. G. Ganek and T. Corbi, “The dawning of the autonomic computing era,” *IBM Systems Journal*, vol. 42, no. 1, pp. 5–18, Jan. 2003.

[3] J. Wildstrom, P. Stone, E. Witchel, R. Mooney, and M. Dahlin, “Towards self-configuring hardware for distributed computer systems,” in *Proc. of the 2nd Int’l Conf. on Autonomic Computing*, Seattle, WA, June 2005.

[4] V. Cardellini, E. Casalicchio, M. Colajanni, and P. Yu, “The state of the art in locally distributed Web-server system,” *ACM Computing Surveys*, pp. 263–311, 2002.

[5] D. Menascé and J. Kephart, “Autonomic computing,” *IEEE Internet Computing*, vol. 11, no. 1, pp. 18–21, Jan. 2007.

[6] H. Chen and P. Mohapatra, “Session-based overload control in qos-aware web server,” in *Proc. of INFOCOM*, 2002.

[7] V. Cardellini, M. Colajanni, and P. Yu, “Request redirection algorithms for distributed Web systems,” *IEEE Trans. Parallel and Distributed Systems*, vol. 14, no. 5, pp. 355–368, May 2003.

[8] P. Barford and M. E. Crovella, “Generating representative Web workloads for network and server performance evaluation,” in *Proc. of ACM SIGMETRICS*, Madison, WI, July 1998.

[9] J. Jung, B. Krishnamurthy, and M. Rabinovich, “Flash crowds and denial of service attacks: characterization and implications for CDNs and Web sites,” in *Proc. of 11th Int’l World Wide Web Conference*, Honolulu, HW, May 2002.

[10] M. Dahlin, “Interpreting stale load information,” *IEEE Trans. Parallel and Distributed Systems*, vol. 11, no. 10, pp. 1033–1047, Oct. 2000.

[11] D. R. J. White, D. L. Scott, and R. N. Schulz, “POED – A method of Evaluating System performance.” *IEEE Trans. Eng. Manage.*, pp. 177–182, Dec. 1963.

[12] J. R. Miller, *Professional Decision-Making: a procedure for evaluating complex alternatives*. New York, NY: Praeger, 1970.

[13] S. Y. W. Su, J. Dujmovic, D. S. Batory, S. B. Navathe, and R. Elnicki, “A cost-benefit decision model: analysis, comparison and selection of data management,” *IEEE Trans. on Database Systems*, vol. 12, no. 3, pp. 472–520, Sept. 1987.

[14] P. Dinda and D. O’Hallaron., “Host load prediction using linear models,” *Cluster Computing*, vol. 3, no. 4, pp. 265–280, Dec. 2000.

[15] T. Kelly, “Detecting performance anomalies in global applications,” in *Proc. of USENIX Workshop on Real, Large Distributed Systems*, San Francisco, CA, USA, 2005.

[16] H. W. Cain, R. Rajwar, M. Marden, and M. H. Lipasti, “An architectural evaluation of Java TPC-W,” in *Proc. of the 7th Int.l Symposium on High-Performance Computer Architecture*, Nuovo Leone, ME, Jan. 2001.

[17] M. Taqqu and V. Teverovsky, “Robustness of whittle type estimators for time series with long-range dependence,” *Stochastic Models*, vol. 13, pp. 723–757, 1997.

[18] D. W. Stockburger, *Introductory Statistics: Concepts, Models, and Applications*, 1996.

[19] D. J. Lilja, *Measuring computer performance. A practitioner’s guide*. Cambridge University Press, 2000.

[20] N. Tran and D. Reed, “Automatic ARIMA time series modeling for adaptive I/O prefetchingp,” *IEEE Trans. Parallel and Distributed Systems*, vol. 15, no. 4, pp. 362–377, Apr. 2004.

[21] K. Arun, M. S. Squillante, L. Zhang, and J. Poirier, “Analysis and characterization of large-scale Web server access patterns and performance,” *World Wide Web*, vol. 2, no. 1-2, pp. 85–100, Mar. 1999.