An Intelligent Framework for Predicting Shifts in the Workloads of Autonomic Database Management Systems

Said S. ELNAFFAR and Pat T. MARTIN, Queen’s University

Abstract—Autonomic computing systems are intelligent systems that manage their own performance. An important characteristic of these systems is an awareness of their environment, particularly their workloads. For a complex system such as the database management system (DBMS) to be self-managing, it should be adaptive to the type of the workload put upon it. Identifying the workload type is key to tuning a DBMS and adjusting its resource allocation. We previously proposed a workload classification methodology that automatically recognizes the workload type and assesses each type’s concentration. Since a DBMS may experience changes in the type of workload it handles during its normal processing cycle, it is not enough for autonomic DBMSs to identify the current type of the workload, but also to predict when a change in the workload type will occur. We could simply keep the workload classifier activated and monitor the system constantly to detect significant shifts in the type of workloads. However, this approach imposes undesirable overhead. In this paper, we propose the Psychic-Skeptic Prediction framework (PSP) that allows the DBMS to forecast major shifts in the workload by combining off-line and on-line prediction methods. Experimental results show that the PSP outperforms other possible operation modes. Furthermore, our approach is generic and can be applied to other similar prediction problems.

Index Terms—workload prediction, database management system, DBMS performance, adaptive systems, autonomic DBMS, artificial intelligence, regression

I. INTRODUCTION

Autonomic computing systems are intelligent systems that manage their own performance. An important characteristic of these systems is an awareness of their environment, particularly their workloads. For a complex system such as the database management system (DBMS) to be self-managing, it should be adaptive to the type of the workload put upon it [8]. Specifically, the distinction whether the workload is OLTP (On-line Transaction Processing) or DSS (Decision Support System) is key to tuning a DBMS and adjusting its resource allocation (buffer pools, sort heap, locking, etc.). We previously proposed a workload identification methodology that automatically identifies the workload type and assesses each type’s concentration using data mining classification techniques [1][2]. We build a workload classifier by analyzing a number of resource-oriented performance measures collected from the DBMS. This workload classifier can be used to identify any workload sample collected over a small time interval. The primary output of the classifier is the DSSness index, which is the percentage of the DSS type vs. the OLTP type in the workload. A DSSness of 80% means that 80% of the workload is classified as DSS and 20% as OLTP.

Identifying the type of the workload, however, is just the beginning. A DBMS may experience changes in the type of workload it handles during its normal processing cycle. For example, a bank may experience an OLTP-like of short debit/credit transactions for most of the month, while in the last few days of the month, the workload becomes more DSS-like with the need to produce financial reports and run long executive queries to produce summaries. In the money market, it also has been observed that traders may exhibit some daily pattern as they access the information systems of their brokers [4].

We believe that such changes can be predicted by analyzing historical data of a DSSness time series. Therefore, it is not enough for autonomic DBMSs to identify the current type of...
the workload, but also to predict when a change in the workload type will occur. We could simply keep the workload classifier activated and monitor the system constantly to sense significant shifts of workloads. However, this approach imposes undesirable overhead and perturbation on the system. We found in our experiments, for example, that running the workload classifier reduces the throughput of the DBMS by 10% on average.

The main contribution of this paper is a more efficient solution by which the DBMS can learn about a workload's dynamic behavior over time and forecast when a change in the workload type might occur in order to proactively reset the DBMS parameters to suit the new workload. It is important to realize that the workload prediction problem builds on top of the work on workload identification, and complements it as illustrated in Fig. 1. The workload classifier's job is to assess the DSSness of the workload at a given time. The workload prediction framework, after it analyzes a time series of DSSness, forecasts major shifts in the DSSness and alerts the DBMS of these shifts. Major shifts are formed when the DSSness reaches predefined thresholds that warrant reconfiguring the DBMS. These thresholds divide the DSSness range into three zones that lead to the identification of three main workload types: OLTP, MIX (of OLTP and DSS), and DSS.

This paper is structured as follows. Section II describes the general prediction approaches that can be used to identify workload shifts. Section III presents a high-level description of our approach and our prediction framework. Sections IV, V, and VI describe the core components of the framework. In Section VII, we describe our experimental setup and evaluate the performance of our prediction approach in comparison with other operation modes. We conclude with a brief description of current and future work.

II. PREDICTION APPROACHES

Initially, we can think of solving this prediction problem by either an on-line or an off-line approach:

**On-line Prediction.** Some prediction problems that attempt to detect the idle periods in computer systems use on-line prediction techniques [5] that require continuous monitoring of the system as long as it is on-line and operational. This approach aims to forecast (using, for example, moving averages or exponential smoothing techniques) very near-future events such as when a disk becomes idle so the system can spin it down in order to save energy.

However, our problem differs from the traditional idleness prediction problem as the latter counts on monitoring one basic, primitive performance index (e.g., number of I/Os) collected at run-time in such a way that does not impose significant overhead on the system. In contrast, our DSSness index, produced by the workload classifier, is a metric resulting from a non-trivial analysis of several performance variables collected on-line from an intricate system such as the DBMS. This inevitably causes extra overhead on the system and impairs its performance. Therefore, counting solely on on-line prediction is not a feasible solution.

**Off-line Prediction.** Another way of predicting workload type changes is by performing a one-time, off-line analysis. This approach is useful for data that are relatively easy to forecast as they likely exhibit a certain cyclical patterns over a time window (e.g., daily or weekly). Based on our experience, the
workload type prediction problem is a good candidate for this approach due to the low volatility of change of workload type in real systems. A change typically occurs over several hours as a result of users’ tendencies to run particular types of applications at certain times.

This one-time, off-line analysis, however, is less trustworthy than the on-line prediction because exceptional behavior may occur during the course of the day in a way that contradicts the suggestion made by the off-line predictor. Consequently, if the DBMS puts absolute trust in the off-line prediction and resets its parameters accordingly then performance could dramatically degrade and the penalty of such a wrong prediction becomes very costly.

Next, we describe our suggested solution embodied in the Psychic-Skeptic Prediction framework.

III. THE PSYCHIC-SKEPTIC PREDICTION FRAMEWORK (PSP)

There are three possible operation modes under which a DBMS can operate with respect to workload type. The first operation mode is the Default mode in which the DBMS uses the default, out-of-the-box settings that suit mixed workloads in general. The second mode is Dominant Workload mode in which the DBMS is tuned to suit the dominant workload throughout the day. The third mode is the On-line mode in which the DBMS counts on constant monitoring in order to forecast near future shifts in the workload type using moving averages.

We propose a fourth mode that uses the PSP framework. This framework takes advantage of the combination of the on-line and off-line predictive approaches in order to make effective, low cost predictions. Note that our focus is on repeatable, daily patterns. We are not concerned with handling bursts that may suddenly occur during the day for some unexpected reason (e.g., producing unscheduled summary report).

As depicted in Fig. 2, the PSP consists of three main components: the TrainingDataModel, the Psychic, and the Skeptic. The premise of the PSP is as follows. The Psychic analyzes a daily time series of DSSness stored in the TrainingDataModel and produces an off-line prediction model, polynomial \( f(t) \), that can estimate major shifts in the DSSness with respect to some DSSness thresholds (see Fig. 3). These shifts are passed to the Skeptic who does not give absolute trust to the Psychic’s predictions. Rather, the Skeptic selectively monitors the system in order to intercept the nearest upcoming forecasted shift. When the shift is due, the Skeptic validates the shift by performing an on-line, short-term prediction using linear regression. So, as shown in Fig. 3, if the shift is due at time \( t \), the Skeptic monitors the system for the \( [t - \Delta, t + \Delta] \) interval. The Skeptic does not instruct the DBMS to reset its parameters unless it confirms the trend of the shift using the linear model.

Without lack of generality, in this paper we assume a day is the time window over which our prediction framework operates. Therefore, the time scale consists of 1440 minutes (24 hours). However, the same concepts are applicable to any other time window such as weeks or months.

A. The PSP Parameters

Predictions produced by the PSP framework rely on a number of parameters. These parameters add flexibility to the framework and make it generic and adaptive to the setup of the IT environment. Most of these parameters are automatically...
estimated by the framework or derived from the computing setup surrounding the DBMS.

Here we provide a brief description of the global parameters that are used across several modules in the PSP. These parameters need to be set once and remain constant afterwards. Other types of parameters are explained in the subsequent sections.

**monCost** is the percentage of performance (throughput) degradation caused by running the workload classifier on-line (default = 10%). This percentage needs to be empirically determined once and remains constant afterwards.

**oltp_threshold** and **dss_threshold**. Current DBMSs are usually tuned based on identifying three main types of workloads [6]: DSS, OLTP, or MIX. We use the DSSness percentage and the **oltp_threshold** and **dss_threshold** to identify workload shifts as follows:

\[
\text{type(DSSness)} = \begin{cases} 
DSS & \text{if } \text{DSSness} > \text{dss_threshold} \\
MIX & \text{if } \text{oltp_threshold} \leq \text{DSSness} \leq \text{dss_threshold} \\
OLTP & \text{if } \text{DSSness} < \text{oltp_threshold} 
\end{cases}
\]

The values of **oltp_threshold** and **dss_threshold** represent the empirical values of the DSSness at which it is worth resetting the DBMS configuration to suit the new workload type. We have empirically found that 30 and 70 are good estimates for the **oltp_threshold** and **dss_threshold**, respectively.

**performanceMatrix** is a 3x3 matrix. If \( x \) and \( y \) \( \in \{ \text{OLTP, MIX, DSS} \) \), then each entry (Workload, Settings,) in the **performanceMatrix** is a performance factor that denotes the relative performance of a DBMS, processing workload type \( x \) when its settings are suitable for workload type \( y \), to the optimal performance of this DBMS when it processes workload \( y \) under settings suitable for type \( y \). Therefore the optimal performance is deemed 1.0 and each entry is a ratio between 0 and 1.0. All cost-benefit analyses use this performance matrix. These performance factors are empirically determined just once by running different combinations of different workloads vs. different DBMS settings. Table I shows the empirically estimated performance factors that we use in our experiments.

**min_check_time** is the minimum number of minutes needed by the Skeptic to execute on-line in order to validate the Psychic’s forecasted shift. This value determines the number of the DSSness samples that will be used to build the linear model at run-time. Throughout our experiments, we found that 30 minutes, which constitutes 2% of the time scale of the day (max_time_scale=1440 minutes), is a reasonable initial size. The final size is eventually determined by the Psychic after analyzing the training data.

Next, we describe the PSP components in more detail. We explain their functions, their parameters, and how they collaborate with each other.

### IV. The Training Data Model

The TrainingDataModel is a queue-like data structure used to store and manage historical samples, or Scenarios, used to train the Psychic and the Skeptic. TrainingDataModel stores numScenarios chronologically ordered scenarios (full day samples). Day_{numScenarios-1} is the most recent while Day_0 is the oldest. Each day is assigned a weight that is double that of the previous day. Assigning weights to the sample days is vital to the quality of the prediction models built in this framework as such weights put more emphasis on the most recent observed days than the older ones. Each scenario consists of a time series, \((t_i, \text{DSSness})\), where \(\text{DSSness}\) is the DSSness reading reported by the workload classifier at time \(t_i\).

Next, we describe the main functions that the TrainingDataModel provides to the other PSP components.

#### A. Predictability Assessment

The PSP solution is based on the premise of the existence of a predictable pattern. We use the autocorrelation coefficient [7], \(r_k\), to test the predictability of the daily DSSness pattern. The range of \(r_k\) is \([-1, 1]\). A near-zero value indicates a lack of correlation between the DSSness values occurring at the same time within each day. A positive value of \(r_k\) indicates a conformance of the DSSness trend while a negative value indicates an inverse trend. In general, we deem \(r_k > 0.5\) a strong indication of having a predictable trend. In our experiments, \(r_k\) is 0.65 on average.

#### B. Model Consolidation

In order to analyze the daily scenarios, we transform them into a compact form that represents all days while taking into account the weight of each day. This consolidated scenario is constructed by calculating \(\text{avgDSSness}\), which represents the weighted average of all \(\text{DSSness}_{(d,t)}\) samples collected at time \(t\), in scenario number \(d\):

\[
\text{avgDSSness}_d = \sum_{d=0}^{\text{numScenarios}-1} \text{DSSness}_{(d,t)} \times w_d
\]
where \( w_d = \sum_{i=0}^{2d} \binom{2d}{i} \), and denotes the weight of scenario number \( d \).

As we discuss in subsequent sections, the consolidated scenario is needed by numerous components of the PSP framework. For example, the Psychic builds an off-line prediction model by applying polynomial regression to the consolidated scenario. It also estimates the interval [earliestCheckTime, latestCheckTime] during which the Skeptic works in order to validate the forecasted shifts. In addition, the dominant workload type is determined by analyzing the consolidated scenario.

C. Determining the Dominant Workload

The DBA can run the DBMS with fixed settings that suit the dominant workload type experienced in a business. This type can be systematically determined by analyzing the consolidated scenario derived from the historical data. The TrainingDataModel determines the dominant workload type by scanning the consolidated scenario and constructing a distribution of the DSSness sample types (DSS, MIX, or OLTP). The type with the highest frequency is deemed to be the dominant one.

The TrainingDataModel can be refreshed by inserting a new DSSness scenario in the queue and discarding the oldest one. Consequently, \( numScenarios \), which represents the queue size, poses a tradeoff between the robustness of the off-line model, by analyzing multiple days, and the pace by which the entire training data model is influenced by a newly inserted DSSness scenario. Experimentally, we found that \( numScenarios \) = 3 is a manageable number of scenarios and produces good prediction models.

V. THE PSYCHIC

The Psychic is one of the core components of the PSP framework. It is primarily responsible for producing an off-line prediction model by tapping the cyclic pattern that occurs during the day. More specifically, the Psychic carries out four main tasks in the following sequence:


2. Finding Shifts. The produced polynomial is used to find the potential workload shifts by finding intersection points of this polynomial with the \( dss\_threshold \) or the \( oltp\_threshold \) (see Fig. 4). Therefore, the Psychic calculates the roots of the polynomial \( f(t) \) when \( f(t) = dss\_threshold \) = 0 and when \( f(t) = oltp\_threshold \) = 0. However, we have to exclude the points that are minima and maxima as they almost touch the threshold levels and do not actually embody real shifts. These false shifts can be easily identified by checking the slope of the curve using the first derivative \( f'(t) \). If \( f'(t) = 0 \), then shift \( t \) must be discarded.

So far, the Psychic could find all shifts that may occur but we still lack the semantics of each shift. A shift can be one of four types depending on its trend: OLTUP_UP_TO_MIX, MIX_UP_TO_DSS, DSS_DOWN_TO_MIX, or MIX_DOWN_TO_OlTP. The slope of the shift can determine the direction of the shift by identifying its inclination (slope > 0) or declination (slope < 0).

3. Estimating Shift Check Time. In order for the Skeptic to validate a particular shift that has been forecasted by the Psychic, the Skeptic needs a timeframe during which it monitors the workload and eventually decides whether to endorse this shift or to disregard it. The start and end time of this timeframe, for each shift, is estimated at this stage.

4. Filtering Shifts. Not all shifts are good. The Psychic performs a cost-benefit analysis to determine if a shift is worth consideration or if ignoring it would be more beneficial to the overall system performance.

More details about these tasks are explained in the following subsections.

A. Off-line Model Generation

The Psychic uses a polynomial as an off-line model. A representative polynomial is generated by applying the polynomial regression algorithm to the consolidated scenario obtained from the TrainingDataModel. In our experiments, the produced polynomials are mostly from the 3rd and 4th degrees. There are many tools that can be used for time series prediction such as neural networks, ARMA/ARIMA (Autoregressive Moving Average/Autoregressive Integrated Moving Average) models, DPLL (Digital Phase Locked Loop), digital filters, or Fourier series [7]. These models can be used to predict the DSSness (dependent variable) at a given time (independent variable). However, the Psychic needs to predict when (i.e., time) the DSSness reaches specific threshold. This requires dealing with the inverse of the prediction function, which is not always easy to derive using the above prediction tools. The extrapolation using polynomial regression, on the other hand, lends itself to the ease of geometric manipulation (i.e., it is easy to find where the polynomial intersects with certain threshold) and it is an intuitive, compact representation for the workload trend.

B. Finding Shifts

The Psychic uses the generated polynomial to find the points of time where the DSSness index intersects with the \( dss\_threshold \) or the \( oltp\_threshold \) (see Fig. 4). Therefore, the Psychic calculates the roots of the polynomial \( f(t) \) when \( f(t) = dss\_threshold \) = 0 and when \( f(t) = oltp\_threshold \) = 0. However, we have to exclude the points that are minima and maxima as they almost touch the threshold levels and do not actually embody real shifts. These false shifts can be easily identified by checking the slope of the curve using the first derivative \( f'(t) \). If \( f'(t) = 0 \), then shift \( t \) must be discarded.

So far, the Psychic could find all shifts that may occur but we still lack the semantics of each shift. A shift can be one of four types depending on its trend: OLTUP_UP_TO_MIX, MIX_UP_TO_DSS, DSS_DOWN_TO_MIX, or MIX_DOWN_TO_OlTP. The slope of the shift can determine the direction of the shift by identifying its inclination (slope > 0) or declination (slope < 0).

C. Estimating Shift Check Time

This task determines the shortest period of time during which the Skeptic will run in order to validate a forecasted shift. This goal is achieved in two steps: 1) determining shift bounds, and 2) estimating earliest and latest check times.
Determining Shift Bounds. The extreme bounds, \([\text{lowerBoundCheckTime}, \text{upperBoundCheckTime}]\), that delimit a shift are determined by the nearest local maximum and local minimum surrounding the shift time as illustrated in Fig. 4. They represent the search space, \([a..b]\), for estimating the earliestCheckTime and latestCheckTime period, \([e..l]\), as explained in step 2 below. The first and last shifts may become special cases. If the first shift is not preceded by a minimum or maximum, the lowerBoundCheckTime is set to zero, which is the beginning of the day. If the last shift is not followed by any minimum or maximum then the upperBoundCheckTime is set to the last minute of the day (\(\text{max_time_scale}\)).

Estimating Earliest and Latest Check Times. In this step, the Psychic tries to find a subset period, \([\text{earliestCheckTime}, \text{latestCheckTime}]\), within the \([\text{lowerBoundCheckTime}, \text{upperBoundCheckTime}]\) of a shift. This is imperative as it reduces the time needed by the Skeptic to validate a shift at run time. The Psychic estimates \([\text{earliestCheckTime}, \text{latestCheckTime}]\) by analyzing the training scenarios stored in the TrainingDataModel. Initially, the Psychic starts with \(\text{earliestCheckTime} = t - (\text{min_check_time}/2)\), and \(\text{latestCheckTime} = t + (\text{min_check_time}/2)\), where \(t\) denotes the expected shift time. This interval is incrementally expanded until the Skeptic’s linear model applied to the consolidated scenario agrees on the trend of the shift. Expansion is performed by decrementing \(\text{earliestCheckTime}\) and incrementing \(\text{latestCheckTime}\) such that the conditions \(\text{earliestCheckTime} >= \text{lowerBoundCheckTime}\) and \(\text{latestCheckTime} <= \text{upperBoundCheckTime}\) are not violated.

D. Filtering Shifts

Some of the detected shifts might not be beneficial to the performance. A shift might be too short such that it is not worth resetting the DBMS’s configuration parameters. The Psychic performs a cost-benefit analysis for each shift in order to decide whether to accept or reject a shift. This decision is made by comparing the performance difference between the two cases:

- **The shift is accepted.** This implies that the Skeptic causes some overhead due to its validation procedure, and that the DBMS’s parameters are reset.
- **The shift is discarded.** No validation is performed by the Skeptic, and the DBMS retains its current settings.

Note that filtering is not needed if we have fewer than two shifts. The cost-benefit analysis is ultimately based on the global parameters \(\text{performanceMatrix}\) and \(\text{monCost}\).

VI. THE SKEPTIC

The Skeptic’s main function is to validate the Psychic’s forecasted shifts. For each upcoming shift, the Skeptic samples the workload from \(\text{earliestCheckTime}\) to \(\text{latestCheckTime}\). The workload samples are analyzed to confirm whether the trend of a shift conforms to the Psychic’s prediction. The Skeptic builds an on-line prediction model using linear regression that fits the collected samples. The slope of the line is used to determine the trend of the workload. If the on-line prediction model confirms the shift, the DBMS’s settings are reset to suit the upcoming workload type. Otherwise, the DBMS resets its settings to the default, which is the safest resort and can sub-optimally handle MIX workloads of OLTP and DSS.

VII. EVALUATION

A. Experimental Setup

The PSP framework is implemented in Java. We evaluate
this framework by comparing the performance of the DBMS under the PSP to the performance obtained under alternative operation modes. We test our framework using artificially generated data that allow us to examine specific cases as well as arbitrary situations. DSSness data are generated using the notion of Scenarios and ScenarioDescriptors. A ScenarioDescriptor can be perceived as a template for generating scenarios that exhibit a particular daily pattern. Therefore, a particular ScenarioDescriptor is used as a factory to generate multiple daily scenarios that exhibit a particular pattern. A ScenarioDescriptor can represent any workload pattern that may characterize a special event or season (e.g., statutory holidays, Christmas shopping days, weekends, weekdays, etc.) over any window of time (day, week, etc.).

A ScenarioDescriptor consists of a set of pairs (time, DSSness) that play the role of anchors of DSSness values on the final DSSness curve. The time is a minute during the day so its domain is [0, 1440] (24 hours a day), and DSSness ranges from 0 to 100. These anchors enable us to direct and shape the trend of the DSSness in any way we desire. In order to generate a scenario out of this descriptor, a series of DSSness values are automatically generated between every two consecutive anchors. In order to make our scenarios more realistic, we inject a ±(0-5)% of random noise in the DSSness, and ±(0-2)% of random noise in the time, which is equivalent to ±(0-30) minutes. Noise injected in the time dimension affects when a shift may start or end. Noise injected in the DSSness dimension affects whether a shift is likely to occur or not based on its intersection with the threshold lines.

In each experiment, we simulate the performance of a DBMS run under each of the four operation modes based on the empirically-obtained parameters described earlier. We compare the performance of the DBMS under the PSP framework with the other operation modes, namely, Default mode, Dominant Workload mode, and the On-line mode.

B. Experiments

The performance of the DBMS is examined under the DSSness pattern represented by the ScenarioDescriptor = (0, 10), (150, 20), (160, 23), (250, 47), (350, 60), (450, 70), (470, 67), (500, 63), (650, 58), (660, 50), (850, 25), (900, 20), (1000, 17), (1050, 15), (1150, 16), (1200, 17), (1440, 50)). Fig. 5 shows an instance scenario of this daily pattern. As seen, the DBMS experiences a workload that is mostly OLTP in the first two hours of the day. Then it changes to a mixed workload over the next 12 hours. In the next 9 hours, the dominant workload becomes OLTP, and then it shifts back to a mixed workload. We should notice that the MIXed workload is dominant in this scenario. Therefore, the performance obtained under the Dominant mode is expected to be equivalent to the performance under the Default mode.

The autocorrelation coefficient, $r_s$, of this workload is 0.6562, which indicates a predictable cycle of DSSness across multiple days. The Psychic’s off-line prediction model for this daily pattern is:

$$f(t)=-6.61+0.32\times t-4.07\times 10^{-4}\times t^2+2.64\times 10^{-6}\times t^3+8.73\times 10^{-11}\times t^4$$

The shift schedule for this daily pattern is shown in Table II. The performance of the DBMS under all modes is simulated and examined for at least 30 days. Fig. 6 shows the DBMS’s performance under different operation modes. Note that the reported performance refers to the percentage that can be achieved with respect to the maximum, theoretical performance resulting from the perfect matching of the DBMS settings with the workload type for each minute. As shown in the figure, the best performance is achieved under the PSP (avg. 97.08%), followed by the On-line mode (avg. 87.39%), then the Dominant mode (avg. 76.23%), which is equivalent to Default mode. This implies that the PSP achieved an average 27.37% performance improvement over the Default mode compared to 14.65% performance improvement achieved by the On-line mode. All performance estimates are based on workload classifier overhead of 10% (monCost = 10%).

We obtained similar results experimenting with different DSSness patterns generated by different ScenarioDescriptors.

### VIII. CONCLUSION

Monitoring systems and constant on-line analysis cause performance penalties that may hinder the adaptation of tools such as the workload classifier. Luckily, the overhead of such tools can be mitigated by exploiting characteristics in the workload. In this paper, we introduced the Psychic-Skeptic Prediction (PSP) framework. The Psychic analyses historical data and produces a shift schedule. Each shift indicates whether the workload is heading to the DSS, OLTP, or MIX region. These regions are delimited by two DSSness thresholds. The DBMS does not put full trust in the off-line predicted shifts. Therefore it asks the Skeptic to validate each shift at run-time by sampling the workload for a small interval around the expected shift time in order to confirm its direction. If the shift is approved, the DBMS resets its configuration to suit the new workload type. Experiments with different DSSness patterns show that the PSP outperforms the typical modes.

<table>
<thead>
<tr>
<th>Shift</th>
<th>Shift Type</th>
<th>EarliestCheckTime</th>
<th>Time</th>
<th>LatestCheckTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shift 1</td>
<td>OLTP UP TO MIX</td>
<td>152</td>
<td>167</td>
<td>182</td>
</tr>
<tr>
<td>Shift 2</td>
<td>MIX_DOWN TO OLTP</td>
<td>804</td>
<td>819</td>
<td>834</td>
</tr>
<tr>
<td>Shift 3</td>
<td>OLTP_UP_TO_MIX</td>
<td>1278</td>
<td>1293</td>
<td>1308</td>
</tr>
</tbody>
</table>
IX. CURRENT AND FUTURE WORK

Part of our ongoing work with the PSP framework is augmenting it with another component called the MUM (Model Update Mechanism). The goal of the MUM is to provide an automatic means of keeping the prediction models of the Psychic and the Skeptic up to date by performing regular, pre-scheduled sampling throughout the day in such a way that the PSP framework remains the best mode under which the DBMS can operate. This should make the PSP framework robust and able to adapt to changes that may occur in the workload pattern because the new DSSness samples are used to patch historical data in the TrainingDataModel component. Consequently, all prediction models are refreshed based on the newly patched training data. However, the MUM poses a challenging tradeoff between rendering the PSP framework adaptive by carrying out regular on-line sampling, which is associated with monitoring overhead, and keeping the PSP performance superior. The MUM achieves this goal by automatically optimizing its internal parameters in light of the characteristics of the workload.

Our preliminary experimental results with the MUM demonstrate the adaptability of the PSP framework and its robustness against changes in the workload pattern (characteristics) while the PSP performance remains superior in comparison with their counterpart operation modes.

Our prediction framework is generic and we speculate that this approach can be effective in other situations where workloads exhibit some trend that makes them relatively predictable. As a future work, we intend to apply our approach to a number of DBMS scheduling problems such as when to make incremental backups, re-build indexes and refresh materialized views, update statistics, or reorganize data on the disk in order to enhance performance and utilize idleness of DBMS resources.

REFERENCES