CAPSTONE SYSTEMS MANAGEMENT PROJECT

Geetha Rajgopalan and Wook-Sung Yoo
Department of Software Engineering
Fairfield University
grajgopalan@gmail.com, wyoo@fairfield.edu

ABSTRACT

General Electric Corporate Information Services (CIS) faced the challenge of streamlining its internal system monitoring and management strategy in order to monitor its large scaled systems and provide stable services to the businesses. Current monitoring system with static thresholds at GE CIS, however, resulted in a flood of false alarms and valuable time was lost in responding to these false alarms. The objective of this project was to provide GE a solution by developing a proactive systems management tool to predict any abnormal behavior in the monitored systems and send an alert on time. Algorithms and tools in current market were researched and analyzed to determine what would align with the needs of GE. We compared tools based on several criteria such as accuracy in performance, projected outcomes, accuracy of predictions, maximum data set size, interface capabilities, compatibility, and ease of use. Concluding our research and analysis, Open Forecast Module was selected as the most efficient tool to meet our project needs. We successfully generated multiple performance models for servers that support the business processes of GE CIS. These models can be used to predict future performance and alert if there is a deviation from the expected value. The models will be retrained over time to account for any changes in the behavior of the system. This Systems Management Tool will provide early warnings of degrading performance before the services are completely impacted. The tool based on OpenForecast Module was developed using Java programming and MySQL and provides the following functions: training the model, forecasting based on model, and predicting any impending crisis. Experimental results are also included in the paper.

INTRODUCTION

General Electric CIS lacks an enterprise wide system monitoring and management strategy in its large scaled computer systems. With inconsistent local monitoring tools and processes, CIS is faced with challenges in understanding relationships between configuration items and their capabilities. In addition, inconsistent monitoring tools leaves CIS with no authoritative data source for effective root cause analysis (RCA), automated fault detection or event correlation and analysis (ECA). In the current state, alerting is based on a static threshold, which results in a flood of false alarms and valuable time is lost in responding to these false alarms.

The project initiative is divided into three phases as shown in Figure 1 below: Phase I Standardized Monitoring/Asset, Phase II a Centralized Business Model, and Phase III Custom Views. Phase 1 has previously been completed by GE and Phase II houses the scope of this project.
The goal of this project is to add intelligence and algorithms to understand the normal behavior of the system and be able to compute and predict abnormal system performance. Algorithms and tools available in the market were researched and analyzed to determine what would align with the needs of GE. We analyzed and compared each tool’s compatibility, time investment, ease of use, accuracy in tasks and performance, data set size, projected outcomes, interface capabilities, and accuracy of predictions.

Classical time-series prediction algorithms use history of past observations to predict successive $n$ observations $[1]$. Studies indicate that a model can be created to reflect the behavior of the system and this can be used to extrapolate data any number of steps ahead $[2-4]$. GE has deployed monitoring agents to capture various system variables at periodic intervals. This data can be used to create a model using time-series algorithm. Commercial tools like Netuitive$[5]$ and Integrien$[6]$ provide automated, proactive system management. They use historical data to create models which can predict the system’s behavior at any given time. Since GE was looking at developing an in-house tool that could provide similar capabilities, open-source tools supporting time series algorithms were reviewed.

*R*, an open-source data analysis tool which supports various predictive algorithms like Holtz-Winter or Double Exponential Smoothing, was reviewed as a potential candidate $[7]$. However, the graphical interfaces available for R were not easy to customize and converting the data to a format acceptable by R was not easy either. Similarly, other open-source tools like Zaitun, Cronos, RRD$[8-10]$ etc. were also reviewed.

OpenForecast, a package of general purpose, forecasting models written in Java, provides support for a wide variety of forecasting models such as linear regression, exponential smoothing etc $[11]$. Since the package is written in Java, it supports data formats that can be used with any Java application. OpenForecast also provides support for graphical display using the open source package JFreeChart $[12]$.
OpenForecast met our functional requirements and was easy to integrate with Java application. Concluding our research and analysis, Open Forecast Module was selected as the most efficient and supportive tool to fit our project needs.

**METHODOLOGY**

This section will outline the methodology used to develop the Systems Management Tool and the various components of the system.

**Data Overview**

The dataset included data for 6 servers in GE CIS compiled over a month. Data attributes considered for this project included Server Name, Time Stamp, CPU Utilization, and Ping. These data points were collected at an interval of 5 minutes by the Sitescope monitoring agents.

**Open Forecast Module**

Open Forecast supports various Regression and Exponential models. The forecasting model we selected for the alerting system is the Double Exponential Smoothing model. The Double exponential smoothing model also known as Holt-Winters Exponential smoothing takes into account any trends in the data such as value increasing or decreasing over time. In addition weights are given to observations; meaning that a more recent observation will receive more weight in forecasting than an older observation.

There are two equations associated with Double Exponential Smoothing.

\[
\begin{align*}
    s_t &= a.Y_t + (1-a)(s_{t-1}+b_{t-1}) \quad \text{with } 0 < a < 1 \\
    b_t &= g.(s_t-s_{t-1}) + (1-g).b_{t-1} \quad \text{with } 0 < g < 1
\end{align*}
\]

where

- \( Y_t \) is the observed value at time \( t \).
- \( s_t \) is the smoothed value at time \( t \).
- \( b_t \) is the estimated slope at time \( t \).
- \( a \) - representing alpha - is the first smoothing constant, used to smooth the observations.
- \( g \) - representing gamma - is the second smoothing constant, used to smooth the trend.

To initialize the model, the first forecasted value is set to the first observed value. The initial slope is set to the difference of the first two observations. The values of the smoothing constants alpha and gamma are calculated by modeling the data series.

**Systems Management Tool Graphical User Interface**

The user interface for the Systems Management Tool was developed using Swing API, a part of the Java Foundation Classes (JFC). JFC encompasses a group of features for building graphical user interfaces and adding rich graphics functionality and interactivity to Java applications. It emulates the native look and feel of the platform it’s running on. We selected Java Swing since it is a platform-independent, Model-View-Controller framework for Java. It is highly extensible and customizable. JFreeChart, an open-source package, was also used to provide the charting capabilities of the system. The Java
application of user interface has three main parts: training the model, forecasting with the model, and monitoring and alerting system.

A. Training the Model
The first stage of the Open Forecast Model is training the model. The model takes data from past scenarios and learns from past patterns to predict future scenarios. To train the model the user must select the metric to predict, server to train, and start and end date for data. Only one metric can be selected at a time, hence each metric for the selected server will have its own prediction model and will have to be trained individually. After running the training, the training model generates two constants called alpha and gamma which are stored in the model table for future use in forecasting. The mean absolute deviation for the model is also generated and stored with each server and metric.

Training the model allows for the model to be as up to date and accurate as possible. Therefore, if the system is trained every day using previous day’s data, the model will be accurate and effective in predicting the data for the next day or week.

B. Forecasting
The second stage of the Open Forecast Model is running the predictive model. To run the predictive model, the user selects the server and the metric to predict. Using the model coefficients generated during the training process, the model will generate a forecast for the current day in time interval of 5 minutes.

To forecast for one-period ahead the following equation is used:

\[ F_{t+1} = f_t + b_t \]

To forecast for \( m \)-periods ahead the following equation is used:

\[ F_{t+m} = f_t + mb_t \]

C. Monitoring and Alerting
The third part of the system is monitoring and alerting. The model forecasts every 5 minutes; it generates a predictive data value and compares it to the value stored in the database. The forecast error, which is the difference between the actual value and the forecasted value, is determined. The control limit for the forecast error is two to five times the mean absolute deviation. If the forecast error is five times the mean absolute deviation, the behavior is marked as abnormal. Two consecutive abnormal values will trigger an email notification.

**EXPERIMENTAL RESULTS**

In the example below, the sample dataset provided by GE was used to generate a model for a server named \textit{cihcispapp222} for the metric \textit{LoadAverage15minAvg}. The model constants found by the algorithm were: alpha = 1 and gamma = 0.286 as shown in Figure 2 below.
Figure 3 below shows the observed value of $LoadAverage15MinAvg$ i.e. CPU Utilization for the server $cihcispapp224$ in red and the forecasted value for the same in blue. The forecasted values closely follow the pattern exhibited by the actual values. Figure 4 shows the observed value for $Round Trip time$ in red and the values forecasted by the model in blue. The round trip time is the time taken by the server to respond back to a request.

It is hard to predict a time-series perfectly; it will usually have prediction errors. The mean absolute percentage error (MAPE), a commonly-used criterion to determine forecast errors, is 6% for CPU Utilization. Since MAPE below 10% is considered good prediction, 6% in our system is better than acceptable. Similarly, the MAPE for the round trip time metric is 0% which indicates that the forecasted value follows the observed value.

To further validate the results generated by the Systems Management Tool, we used the Paired t test. A paired t test is usually used to compare two groups of data which have matching data points. In this case, it will help us determine if there is any significant difference between the
observed values for a server metric against the values forecasted by the model. The two tail p-value helps determine if the two groups are statistically different or similar. If the p-value is less than 0.05, there is a significant difference between the forecasted and observed values. As shown in Table 1, the p-value for the \textit{LoadAverage15minAvg} is 1.96. This signifies that forecasted and observed values follow similar trend.

Table 1: t-Test: Paired Two Sample for Means of LoadAverage15minAvg

<table>
<thead>
<tr>
<th></th>
<th>Forecasted</th>
<th>Observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.143754</td>
<td>0.143798799</td>
</tr>
<tr>
<td>Variance</td>
<td>0.002443</td>
<td>0.002451863</td>
</tr>
<tr>
<td>Observations</td>
<td>666</td>
<td>666</td>
</tr>
<tr>
<td>Pearson Correlation</td>
<td>0.999849</td>
<td></td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>665</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>-1.34245</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>0.089954</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.647148</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td>0.179909</td>
<td></td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td>1.963538</td>
<td></td>
</tr>
</tbody>
</table>

Similarly, Table 2 shows the two tail p-value of \textit{Round Trip Time} to be 2.048 which further confirms that our model is able to forecast pretty accurately.

Table 2: t-test: Paired Two Sample for Means of Round Trip Time

<table>
<thead>
<tr>
<th></th>
<th>Forecasted</th>
<th>Observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.01</td>
<td>0.010344828</td>
</tr>
<tr>
<td>Variance</td>
<td>1.2467E-35</td>
<td>3.44828E-06</td>
</tr>
<tr>
<td>Observations</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>Pearson Correlation</td>
<td>2.65544E-15</td>
<td></td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>0.162937353</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.701130908</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td>0.325874707</td>
<td></td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td>2.048407115</td>
<td></td>
</tr>
</tbody>
</table>
Based on the test results shown above, we can conclude that the model developed with Double Exponential Smoothing Algorithm can predict server metrics with reasonable accuracy.

**CONCLUSION**

To resolve the issue of monitoring huge computer systems in GE CIS, many data mining tools in the market have been studied and multiple models for the servers supporting the business processes of GE CIS were generated successfully using OpenForecast with Java and MySQL database. Systems Management Tool was delivered to GE CIS in 2010 for further development and is being integrated with current systems for phase III development. However, there were several restrictions in current Systems Management Tool related to the data and the model. Systems Management Tool assumed data in consistent time intervals. But, the data provided to the system might have inconsistent time intervals between successive measures of metrics, meaning that data point A will be taken at time interval 5 minutes while data point B will be taken at time interval 7 minutes. Therefore, the data should be properly massaged before using in current Systems Management Tool. An automotive process that takes a consistent time interval or adjusts the time interval would make the system more easy to use.

Current Systems Management Tool requires human intervention to train the model for each metric, which can be automatically done as a daily scheduled job to run every fortnight to capture any changes in the server behavior and hence make the model more adaptive. The number of servers and metrics per server forces the application to handle huge sized data leaving room for possible inconsistencies or issues with management and upkeep.

**ACKNOWLEDGMENT**

Our thanks to GE Team and instructors at Fairfield University who have helped us develop this tool and provided the data.

**REFERENCES**


