Open access • Proceedings Article • DOI:10.1109/ICC.2009.5199087

Intelligent Service Monitoring and Support — Source link

Ala Al-Fuqaha, Ammar Rayes, Mohsen Guizani, M. Khanvilkar ...+1 more authors

Institutions: Western Michigan University, Cisco Systems, Inc.

Published on: 14 Jun 2009 - International Conference on Communications

Topics: Service design, Service (business), Service product management, Service management and Service delivery framework

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

Intelligent Service Monitoring and Support

Ala Al-Fuqaha[*], Ammar Rayes[+], Mohsen Guizani[*], Mrinal Khanvilkar[*], Mohammed Ahmed[*]

Abstract— **Intelligent service management techniques play an important role in the continuously and rapidly evolving area of technologically advanced services. High tech companies are looking for better ways to deliver and preserve services to their customers in a competitive way. This paper introduces a new architecture for a scalable service monitoring and support system called Call Home Analysis and Response System (CHARS). The proposed system utilizes data de-noising and filtering techniques to meet the service management requirements of large-scale service deployments. The system utilizes intra and inter-element correlation of events to enhance the services delivered to end-users. Our results demonstrate that the proposed system is effective in covering the performance and fault management aspects for large-scale deployments of advanced services.**

***Index Terms*— Technologically advanced services, service management systems, data filtering and de-noising, flip-flop filter, wavelet transform, rule based expert systems.**

I. Introduction

Technologically advanced services are growing very rapidly worldwide with superb margin. According to Service & Support Professionals Association (SSPA), the industry average margin of software-based services was more than 65% in 2006. High tech companies are looking for better ways to deliver and preserve services to their customers in a competitive fashion. As a result, there is a growing need for intelligent support services that collect information about the conditions of the services delivered to the customer. Such services help to avert potential downtime or unacceptable service degradations.

Normally, tens of logs and alarms can be generated by different services because of a single fault in the network. Most of these alarms can be artifacts of the fault rather than indicating the cause. It is left to the operator to correlate the network events indicating the true failure. Also, as faults rarely occur one at a time, the network operator often has to deal with multiple faults simultaneously. With the increasing size and complexity of networks and the services they deliver, the value of automated tools becomes apparent.

[*] Department of Computer Science, Western Michigan University, Kalamazoo, MI 49008

[+] Advanced Support Technology Group, Cisco Systems, San Jose, CA 95134

The Call Home Analysis and Response System (CHARS) is an intelligent system that receives a stream of events summarizing the status of software-based services and network elements and does what operators used to do traditionally. CHARS functionality includes: collecting a sequence of related network events and identify them as "incident", correlate the collected messages to identify the root cause(s) behind their generation, and associate the collected messages with a resolution procedure. Naturally, the system can deal with multiple faults simultaneously and is capable of making correlations between problems and building problem hierarchies.

The novelty of CHARS stems from its ability to utilize neighborhood, composition, and association relationships between various network elements and software-based services to perform root cause analysis on collected failure messages. Thus, the system correlates network and service logs and events to identify the root causes behind failures. A key design goal of the system has been to present the relevant information to the network operator, thereby eliminating extraneous data. Moreover, CHARS enables the operator to add knowledge to the system by writing scripts that react to the identified root causes. These scripts can change the configuration of the underlying network elements and services. Hence, CHARS helps service providers offer five nine availability (99.999%) by providing mechanisms for real-time root cause analysis and timely and automated resolution of system issues.

The rest of this paper is organized as follows: Section II provides a brief comparison between CHARS and other existing network and service management systems. Section III provides detailed description of CHARS and its architecture and capabilities. Section IV demonstrates use case of how CHARS performs root cause analysis. Section V describes the details of the algorithm used to de-noise and estimate the metrics conveyed in collected performance data. Section VI concludes our study, summarizes our findings, and discusses possible extensions.

II. Related Work and Motivations

The ITU telecommunication standardization sector (ITU-T) has contributed a great deal to standardize a framework for network management tasks through its X.700 and M.3000 series recommendations. These recommendations collectively known as the Telecommunication Management Network (TMN) model define a framework of four logical layers for network management, namely: business, service, network, and element management layers. Furthermore, the TMN model organizes the management tasks in five conceptual areas collectively abbreviated as FCAPS: Fault Management (FM), Configuration Management (CM), Accounting Management

(AM), Performance Management (PM) and Security Management (SM).

The Simple Network Management Protocol (SNMP) defined by the Internet Engineering Task Force (IETF) is typically used for the management of IP networks. Most commercial management and planning solutions are based on SNMP, including: HPOpenview, VisionAlert, VisionTrend, OPNET, and AdventNet Web Network Management System (NMS).

Our system, CHARS, is based on the management paradigm which enables vendors to monitor and reconfigure services deployed in customer premises i.e. inter-domain management. This paradigm is based on the services or their proxy generating call home messages. A call home message is a notification message generated when certain events occur in the network elements or user services [4]. Various telecommunications equipment providers, software and hardware vendors including CISCO, IBM, EMC and Microsoft realize variations of the call home feature on their products. In CHARS, performance and fault management messages are interpreted into call home messages. These *Call Home* messages have details of the event that occurred. When received by CHARS, these messages are inserted into the expert system engine. The messages are correlated by the engine and appropriate actions are taken if the activated rules are configured to do so.

CHARS uses a Rule-based Expert System at its core. We believe that expert systems provide an extensible and a natural approach to simulate the judgment of human subject-matter experts. This enables support engineers to define the symptoms of various service failure scenarios and the associated actions that need to be executed as a corrective measure i.e. they can add knowledge to the system. This almost eliminates human intervention. CHARS has the ability to *react* to the events using the knowledge base that has been inserted to the expert system engine, which the "Expert Advisor" [2] lacks.

Our embedded expert system is based on a novel algorithm that provides significant enhancements to the traditional Rete algorithm. It focuses to achieve fast preferential search of the Rete network; thus, minimizing the time needed to trigger the rules when the knowledge base is large. This enables our embedded expert system to be used in environments with hard timing requirements or large-scale knowledge bases such as large-scale enterprise networks, battlefield networks, control area and sensor networks.

III. Proposed Architecture

This research fits within the smart call home framework focusing on the design of a network wide call home message correlation and response system. This enables the Technical Assistance Center (TAC) engineers [8] to write rules that correlate call home messages and identify the root cause issues. The correlation process utilizes topological information at layers *1* through *4* as well as service composition and association information, thus enabling capabilities beyond those obtained through intra-element correlation.

Our proposed Call Home Analysis and Response System (CHARS) offers the following features:

- *Network Wide Correlation*: This enables the call home messages generated by different services and network elements to be correlated together with service composition and association information as well as topological information at layers *1* through *4* to identify the root cause of network and service outages.
- *NMS and Call Home Cooperation*: Current root cause analysis schemes are handicapped by a lack of cooperation between NMS and call home. In this work, we allow fault and performance data to be used efficiently to identify the root causes behind network and service outages.
- *Integrated TCL/Expect scripting*: This enables TAC engineers to write TCL/Expect scripts that react to the identified root causes and changing the configuration of the underlying services and network elements (e.g., changing service parameters, service locations, or routing entries based on correlation results). [10]
- *Expert System based correlation engine*: Our expert system tracks the Probability Density Functions (PDF) of various events to perform preferential search of the knowledge base. This strategy is used to minimize the amount of time needed to search large knowledge bases, thus allowing real-time handling of a large collection of rules and call home messages [9].
- *Fuzzy and Crisp Logic Rules*: This enables TAC engineers to utilize *linguistic* variables and operators to write the correlation rules, thus enabling faster and more intuitive implementation of the correlation rules.
- *Visual Illustration of correlation results*: Correlation results are visually presented to the TAC engineers on the network topology allowing easier and faster identification and handling of the root causes.

CHARS is a rule-based expert system that has been developed using the Java 2 Enterprise Edition (J2EE). It is an intelligent, reactive, highly-scalable and highly extensible call home expert system. Here's an explanation of the design goals and objectives of CHARS:

- *Intelligent*: CHARS utilizes artificial intelligence techniques to perform ***forward chaining*** to identify all relevant documentation and solutions given a set of collected call home messages. This helps support engineers to share their expertise in the field efficiently through an intelligent inference system.
- *Reactive*: CHARS utilizes TCL/Expect scripts to fix and isolate network and service outages (i.e., reactive healing of network and service outages) identified by the forward chaining engine through executing shell commands (e.g., DOS, BSH, IOS, SNMP) on the network elements or servers [5, 11].
- *Highly scalable*: A major goal of CHARS is to build a call home expert system capable of managing a large rule base database of root causes of outages and their corresponding documentation and potential solutions.
- *Highly Extensible*: The proposed system utilizes XML to be highly configurable and extensible. For Example, the rule engine is responsible for associating call home messages with root causes, and relevant

documentation/solutions is configurable through XML rules. Data type Definition (DTD) files was developed to specify the structure of the XML rules.

In this work, we extend the call home feature with a rule-based expert system that matches the received call home notifications with the relevant documentation that can help engineers to fix the problems faster. Furthermore, we utilize an XML-based language to associate call home messages with root causes and sequence of actions that can be used to resolve the problem. These actions can be specified as Command Line Interface (CLI) or Simple Network Management Protocol (SNMP) commands. These commands can be automatically executed given the address of the server or network element using a combination of Toolkit Command language (TCL) [15] and Expect (TCL Extension for automating interactive applications such as Telnet, FTP, rlogin, etc.) [10]. This allows service support engineers to find solutions to potential outages faster and share their solution with other fellow engineers. Furthermore, the use of TCL/Expect allows real-time automated resolution of system issues; thus, minimizing the harm that can be caused by network or service outages.

We believe that current implementations of the call home feature are handicapped by a lack of cooperation with the fault and performance management functional areas of the ITU and ISO/OSI recommendations. Our proposed call home analysis and response system is designed to overcome this. We believe that fault and performance monitoring provide a negative view of the network (e.g. a model of failure occurrences that quantifies network dysfunction) which gives us the opportunity to generate call home messages that aggregate collected fault and performance data; thus, enabling back-end components to react to this data to minimize network and service outages.

Figure 1 illustrates the overall architecture of CHARS as an inter-domain architecture. The architecture is composed of following components:

A. Front End: This component serves as an agent that creates and sends call home, fault and performance messages reporting system status to the enterprise-robot component.

B. Enterprise-robot: This component utilizes a rule-based expert system to correlate fault messages and generate call home messages accordingly. This component also utilizes data filtering techniques to generate call home messages that aggregate collected performance messages. Finally, collected and generated call home messages are forwarded using the transport component for handling. The following provides the details of the sub-components of this module:

1) Performance Data De-noising and Aggregation: This module takes the performance data from the devices as input. We have the wavelet transform (DWT) and the flip-flop filters implemented for data de-noising.

2) Call Home Message Generator: There are two ways for generating call home messages: one for FM data and the other for PM data.

- *SNMP FM Data Parsing*: The fault messages here are represented by SNMP traps (SNMPv1 and SNMPv2 traps) which are received when specific network events or errors occur on managed network devices [5, 11]. We convert the SNMP traps into generic call home messages, using XML-based parsing utility, which are then forwarded using the transport component for handling.
- *SNMP PM Data Parsing*: The filtered PM data is fed into this sub-component. Call Home messages are generated depending upon the value of the filtered PM data. These call home messages are either inserted into or cleared from the underlying expert system. This is based on thresholds defined by the user in the parser XML file.

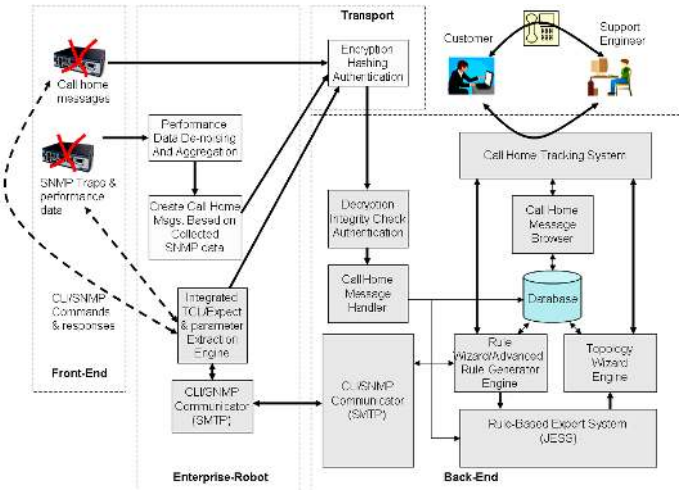

Fig. 1. Overall architecture of the smart CHARS

3) Extraction Engine: The enterprise robot also has an expert system engine which can be used to correlate call home messages. Usually the rules should be setup such that most problems in that network should be resolved by the enterprise robot (i.e. inter-element correlation within the domain). If an issue cannot be resolved locally, a default rule should be setup which generates a new call home message. This call home message contains the details of the environment of the devices which generated the fault or performance message. These details are obtained by the use of integrated TCL/Expect tool in the engine. This new call home message is then forwarded through the transport component to the back-end components (i.e. inter-domain correlation).

4) Integrated TCL/Expect Execution: Based on the received sequence of call home messages, the expert system checks the satisfyability of its rules with the current existing call home messages at the engine. In case a rule is fired, the associated TCL/Expect script is executed to perform automatic corrective actions.

5) CLI/SMPT Communications: This module makes use of Pretty Good Privacy (PGP) encryption techniques to get the encrypted CLI/SNMP commands from the back-end component. These commands are then going to be executed using the TCL/Expect extension.

C. Transport: This component is used by the enterprise-robot component to send secure messages reporting system issues to the back-end components. This component is also used to send CLI/SNMP commands from the back-end

component to the enterprise robot using PGP encrypted e-mails to bypass enterprise firewalls.

D. Back-end: This component forms the core of CHARS offering automated diagnosis capabilities using a rule-based expert system and integrated TCL/Expect capabilities. The rules are expressed using a combination of crisp and fuzzy-logic; thus allowing easy and compact representation of the correlation rules at the network level as opposed to the element level. Further, this component provides a visual illustration of the correlation results through web to enterprise users and support engineers. The sub-components that make up this module are as follows:

1) Decryption Integrity Check Authentication: This component takes the secure messages from the enterprise robot. Verification is done on these messages to confirm whether they are valid. Valid messages are decrypted and forwarded to the Call home message handler.

2) Call Home Message Handler: The decrypted messages are converted into valid call home messages and then fed into the inference engine and the database.

3) Rule Wizard: The user adds knowledge i.e. rules to the engine using this sub component. The rule wizard is simple step by step wizard which any user can use to add rules to the engine.

4) Advanced Rule Generator: This component also generates rules, but can be used only by expert engineers for adding more rules at a time. This module gives the users a friendly UI interface to create rules using XML.

5) Topology Wizard: A web based component that provides a visual illustration of the underlying network and the correlation results.

6) Call Home Message Browser: It maintains the history of the Call home messages. This would allow the engineers to determine the health of the managed underlying network.

7) Service Request Tracking System: This module provides a visual illustration through a web interface to enterprise users and support engineers. Thus, it allows them to track the underlying network and its correlation results.

8) Rule-Based Expert System: This component is the inference engine of CHARS. We have chosen JESS (Java Expert System Shell). It also has the capability of handling fuzzy logic [9].

IV. Illustration Scenario

For example, when a service running on a core router fails, call home messages can be generated reporting the failure and its effect on user's service. In such scenarios, it is very beneficial for network support engineers to have access to an intelligent service that can be searched to quickly find relevant documentation and potential solutions by other engineers who encountered the same scenario in the field. This helps to pinpoint relevant documentation and solutions as quickly as possible. Thus, it is averting potential downtime or unacceptable network degradation.

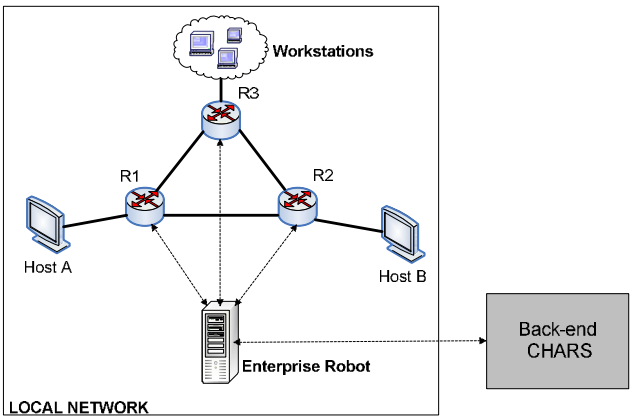

Fig. 2. Topology Example

```
If Router1→Router2 traffic fails & Router2→Router1 traffic fails
   & if Router1.neighbor is also neighbor of Router2
   Then
      reroute traffic Router1→Router1.neighbor→Router2
      cause = Linkdown between Router1 and Router2
End
```

Fig. 3. Rule Example

Figure 3 is an example of a typical rule in layman's term. Notice that the *Router1* and *Router2* in the examples are general. i.e. they are not specified as any particular instance. So this rule can be applied to various routers.

As shown in Fig. 2, there are three routers. Router R_1 and R_2 are connected directly together with a shared link. And all the routers are in direct connection with the centralized enterprise robot that is in turn connected to the back-end CHARS. R_1 and router R_2 and configured in such a way the traffic between host *A* and host *B* flows directly through Routers R_1 and R_2. So, the route of the packets generated by host *A* to host *B* is: (host *A*) ➔ R_1 ➔ R_2 ➔ (host *B*).

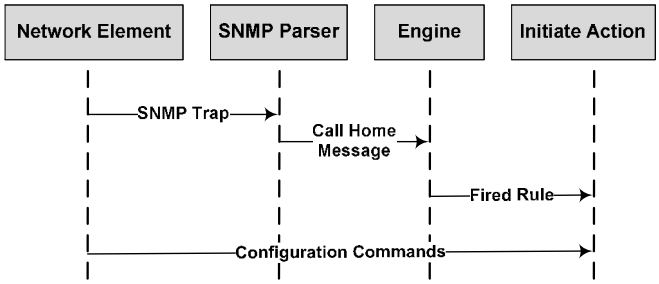

Fig. 4. Sequence of actions taken upon receiving traps at robot

When the connection from R_1 to R_2 fails for some reason, the above traffic route will be broken and both R_1 and R_2 will send SNMP traps to indicate the failure. In such scenario, if the enterprise robot has prior knowledge (predefined rule), then, the sequence of events that can occur is shown in Fig. 4.

When the system gets two call home messages routers R_1 and R_2 indicating the failure of the underlying link, traffic is rerouted through router R_3. The rerouting of the traffic is done by the use of TCL/Expect scripts which the support engineers have in place. In this case, multiple messages are generated to report a single root cause (i.e., link failure). CHARS inter-element correlation engine identifies the root causes associated with the received messages.

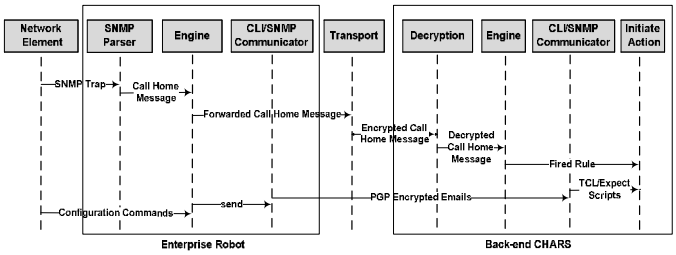

Fig. 5. Sequence of actions taken upon receiving traps at robot and back-end

If the enterprise robot lacks the knowledge to handle this scenario, it will forward the generated call message to the back-end component of CHARS. Figure 5 shows the series of events that will take place in this scenario.

The reply commands change the configuration of R_1 so that it reroutes its traffic through R_3. This makes the route path of the packets generated by host A to host B as follows: (host A) ➔ R_1➔ R_3 ➔ R_2 ➔ (host B).

V. Performance Evaluation

A. Introduction:

Our goal is to explore the advantages of using data filtering techniques to advertise changes to the back-end component of CHARS. In this work, we focus on techniques to reduce the overhead in the network, while ensuring that network performance does not deteriorate.

Usually, producing quality estimates is challenging because network observations are noisy. Current systems depend on simple, exponentially-weighted moving average (EWMA) filters [6]. These filters are either able to detect true changes quickly or to mask observed noise and transients, but cannot do both.

In this work, we utilize these filtering techniques for de-noising and estimating performance data to detect and report true changes to CHARS's back-end component. Our approach uses a combination of flip-flop filters and the wavelet transform to remove noise from raw performance measurements. Both approaches serve to provide more accurate estimates (compared to raw performance measurements) for later use by CHARS back-end component. Since the performance data changes frequently, a smart strategy that decides when the aggregated performance data should be sent to the back-end component of CHARS. We believe that fuzzy-logic can implement this strategy efficiently; thus minimizing the amount of traffic forwarded to the back-end component of CHARS.

B. Analysis:

This study compares traffic overhead and imprecision of advertised QoS parameters when using periodic or filtered updates. In our simulation study, the traffic model is assumed to be Poisson. The QoS parameter monitored is the traffic overhead represented in terms of the number of advertisements sent from the enterprise robot to the back-end. Also, the QoS metrics include the imprecision of the data transferred between the two parties.

The weight of the EWMA is controlled by a fixed-weight EWMA filter (Error based filter and weight = *0.6*) that estimates the error. Advertisements are generated when the difference between the previously advertised estimates and the current is more than $3 * \overline{MR} / 1.128$. QoS parameters are measured every ΔT. $\overline{MR}$ is calculated based on the last *10* measurements.

The performance advantages obtained through the use of CHARS are as follows:

- Using a conventional network management tool causes an overhead on the traffic of the network. In CHARS we are using a combination of the flip-flop filter, the wavelet filter and the update policy to minimize this overhead.
- As we have earlier seen, CHARS can be operated as a single independent network management (i.e. managing a single network, ROBOT), or it can be operated in a more diversified mode (where many networks can be handled by CHARS). The latter architecture involves the use of CHARS as a ROBOT for one network which forwards the call homes generated in its network to the back-end CHARS which can handle many ROBOTS and hence multiple networks at a time.
- Any network parameter can be forwarded to the back-end CHARS as a performance parameter. The delivery of such a parameter continuously provides inefficient use of the network resources.
- CHARS uses filtered updates to reduce the number of advertisements from the robot to the back-end CHARS. The process of fixing up the threshold is explained in Section III.
- Tests have been conducted to illustrate that the update strategy utlized by CHARS serves to decrease the advertised changes from the robot to the back-end while not lowering the precision of the monitoring process.

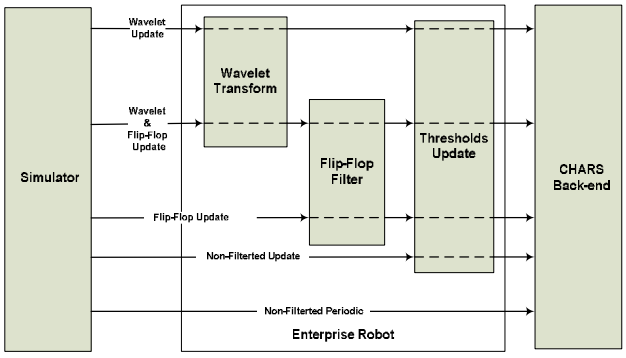

Fig. 6. Performance evaluation experiments

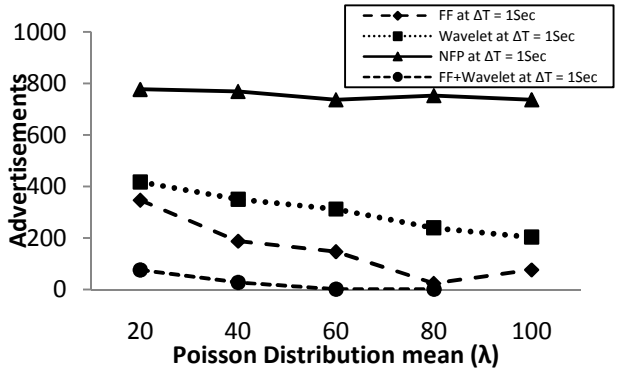

Fig. 7. Advertisements: Non-filtered periodic vs. filtered update schema

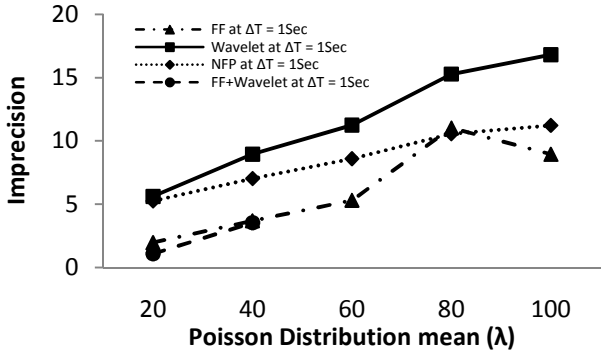

Fig. 8. Imprecision: Non-filtered periodic vs. filtered update schema

Figures 7 and 8 above illustrate the advantage of using the flip-flop filter and the wavelet transform to minimize the traffic overhead. These figures demonstrate that the advertised changes in the Non-Filtered Periodic mode (NFP) almost remain constant since it keeps on generating advertisements at a constant rate. The problem of this mode is that it can lead to an overloaded network. Through the filtered update policy, our aim is to de-noise the data (using wavelet filter) and predict a better estimation of the value (using flip-flop filter). This serves to minimize the number of advertisements as shown in the above graphs. Only changes which are above or below a certain threshold are advertised.

Figures 9 and 10 illustrate that our filtered updates policy (the flop-flop filter, the Wavelet filter along with the update ploicy) produces less advertisements and have lower imprecision over the others. The upper and the lower thresholds for the update policy are normally fixed by the user. We are fixing them up at 10% upper/lower than the selected mean of the Poisson distribution (λ). For e.g. for λ = 20 the upper threshold is 22 (20 + 10% of 20) and the lower threshold is 18 (20 – 10% of 20).

VI. Conclusion

Maintaining any network is a complex process and requires constant human intervention. However, our proposed CHARS system serves to reduce this human intervention by using an expert system at its core. By integrating TCL/Expect scripts with the system, we have opened wide-range of possibilities where CHARS can perform a variety of actions. These possibilities reflect the field experience of service, system and network engineers.

By using the update policy and the combination of the flip-flop and the Wavelet filters we are able to reduce considerably the traffic overhead which is generally present in other network management systems. Furthermore, the ability to use CHARS as an enterprise-robot and as a back-end component has made the monitoring process of large networks more feasible. Moreover, each network along with its enterprise-robot can have its own specific knowledge.

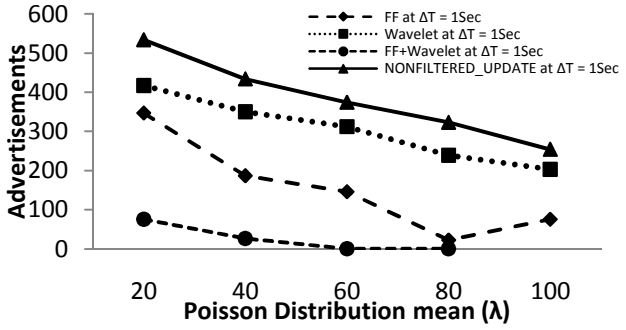

Fig. 9. Advertisements: Non-filtered update vs. filtered update policy

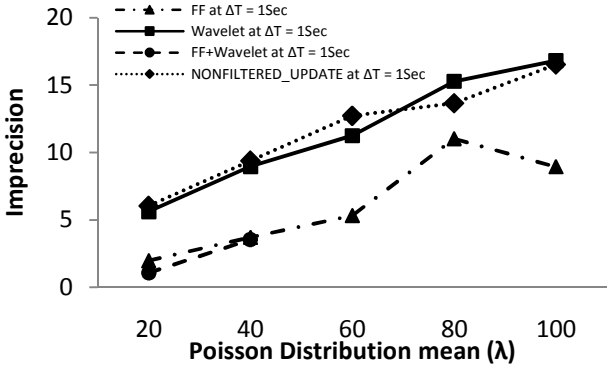

Fig. 10. Imprecision: Non-filtered update vs. filtered update schema

References

[1] Yongjian, Yang., & Songyang, Han. "Expert System-Intelligent Management for ATM Network". IEEE, 1999

[2] White, T., & Bieszczad, Andrzej. "The Expert Advisor: An Expert System for Real Time Network Monitoring". Northern Telecom

[3] Lindqvist, Ulf., & A. Porras, Phillip. "Detecting Computer and Network Misuse through the Production-Based Expert System Toolset (P-BEST)". *IEEE Symposium on Security and Privacy*, Oakland, California, May 9–12, 1999.

[4] Cisco Systems Inc. (May 2007). "Cisco MDS 9000 Family Troubleshooting Guide, Release 3.x". San Jose, CA: Cisco Press.

[5] Cisco Systems Inc. and ILSG Cisco Systems. (August 15, 2003). "Internetworking Technologies Handbook, Fourth Edition". San Jose, CA: Cisco Press.

[6] Kim, Minkyong, & Noble, Brian. "Mobile Network Estimation". ACM Conference on Mobile Computing and Networking, Rome, Italy, June 2001.

[7] Oh, Y., Ju, H., Choi, M. and James Hong. "Interaction Translation Methods for XML/SNMP Gateway". Springer-Verlag London, UK, 2002.

[8] Technical Assistance Center (TAC) Quick Reference Guide. Cisco Systems, Inc.

[9] E. Friedman-Hill, Jess in Action, Greenwich, CT: Manning, 2003

[10] D. Libes, Exploring Expect: A Tcl-based Toolkit for Automating Interactive Programs, First Edition, O'Reilly & Associates, Inc., 1995

[11] W. Stallings, SNMP, SNMPv2, SNMPv3, and RMON 1 and 2, Third edition, Reading, MA: Addison-Wesley, 1999.