

The Visualisation of Application Delay Metrics for a Customer Network

J. Rademan, J.L. Wesson, and D. van Greunen

Department of Computer Science and Information Systems

University Of Port Elizabeth, PO Box 1600, Port Elizabeth 6000

Tel: +27 (0)41 504 4254, Fax: +27 (0)41 504 2831, Email: csbjr@upe.ac.za,

Topic: Network Planning, Sub-Topic: Traffic Engineering

ABSTRACT – Application services are fundamental network components that allow organizations the ability to operate efficiently. It has become essential for organizations to monitor the performance of these critical applications. Traditional network analysis tools, however, cannot cope with the size of today’s network infrastructures and the volume of network data generated. The goal of this paper is to discuss the development of a visualisation system, called AppVis, that uses new information visualisation techniques to enable UPE to effectively visualise the application delay performance of the ITS application implemented on the network.

KEYWORDS – Application performance management, application delay metrics, information visualisation.

I. INTRODUCTION

Large-scale, communication network infrastructures provide the foundation on which E-commerce and other application services are based [1]. Application services form a fundamental part of a network’s ability to operate efficiently. These are facilities that are present within a network’s infrastructure that help users work together collectively while located remotely from each other [2]. Application services are divided into two groups of network-dependent applications, including infrastructure and end-user applications [3]. Infrastructure applications include packaged applications such as E-mail, dns, ftp, nfs and other services that form part of an organisation’s computing infrastructure. End-user applications have several end users who are in-house employees, customers, or partners and include packaged Enterprise Resource Planning (ERP) and E-commerce applications. It has become essential for organizations to manage the performance of network-dependent applications from end-point to end-point and from business function to business function [4].

This paper discusses the goals of Application Performance Management (APM) and the generic problems that occur within the domain. A brief description of application delay metrics and their method of collection are given. An overview of information visualisation is provided which discusses how application delay metrics are currently being visualised and highlights the limitations of existing methods. Based on these findings, the analysis and design of a solution that improves on these methods is discussed.

II. APPLICATION PERFORMANCE MANAGEMENT (APM)

To perform adequate APM for network-dependent

applications requires an understanding of how an application is performing from a user’s perspective [3]. This user perspective is particularly important for end-user applications, while a perspective from a single monitoring server is adequate for most infrastructure applications. A key measure of an application’s performance is its response-time (delay) metrics as perceived by end users. Once users start to experience degraded performance, the following questions need to be asked [3]:

- Is it the network or the application?
- If it is the network, where in the network?
- If it is the application, where on the application server?

Network performance directly impacts application performance, but it is difficult to pinpoint how the network affects an individual application [5]. It’s equally difficult to determine how applications impact overall network performance. This information is important in order to identify the root cause of sluggish delay of applications implemented on the network.

The University of Port Elizabeth (UPE) has an extensive network infrastructure, which supports several LAN-based application services that are distributed across campus. However, there are no effective or efficient means present to visualise the performance of these application services. The goal of this research is to investigate and develop a system incorporating new information visualisation techniques to enable UPE to effectively visualise the application delay performance of the Integrated Tertiary Software (ITS) application implemented on the network. The visualisation techniques will attempt to analyse and explore various delay metrics of ITS across different VLANs based on user specified periods and certain delay statistics, including mean and maximum delay. The delay metrics will be collected from a network monitoring technology known as PacketShaper [6]. The visualisation process will be realised using multiple views and will incorporate various interaction techniques to enhance the analysis and exploration of performance patterns.

III. APPLICATION DELAY METRICS

The delay metrics that the AppVis visualisation system will visualise include various delay components, including the *server delay*, *network delay* and *total delay* of an application service.

- *Server delay*: represents the number of milliseconds (ms) the server uses to process a client’s request after it receives all required data. The server delay is the time after the server receives the last request packet and before it sends the first packet of response.

- *Network delay*: represents the number of milliseconds (ms) spent in transit when a client and server exchange data. It includes the transit time for all packets involved in a request-response transaction. The amount of time the server uses for processing a request is not included.
- *Total delay*: represents the number of milliseconds (ms) beginning with a client's request and ending upon receipt of the response.

PacketShaper was selected as the network monitoring technology responsible for measuring the delay metrics of ITS for visualisation purposes. PacketShaper is a bandwidth-management solution that brings predictable, efficient performance to applications running over a networking infrastructure [6]. A four-step approach is used for safeguarding application performance, that includes identifying and classifying network applications, analysing their performance, enforcing policy based bandwidth allocation and generating reports on the results.

PacketShaper incorporates a response-time management (RTM) engine for measuring application delay components on the network. It breaks each response-time measurement into *network delay*, *server delay* and *total delay* components.

PacketShaper tracks the course of a client-server transaction and uses information about a TCP connection to differentiate one portion of the exchange from another. Figure 1 illustrates PacketShaper's model of a connection's delay components.

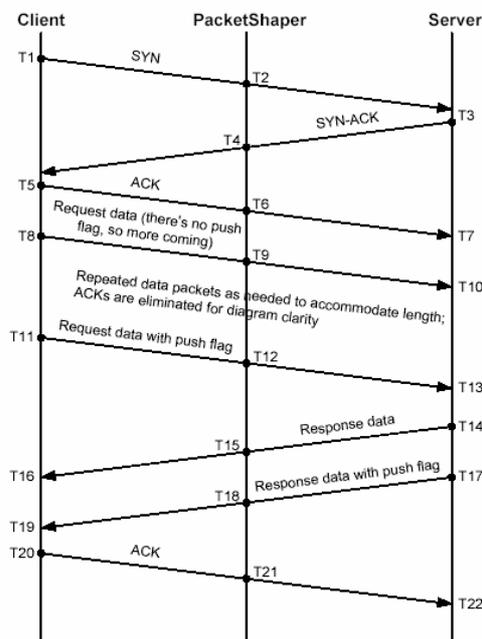


Figure 1: PacketShaper's client-server transaction model

Figure 1 is a standard TCP diagram showing the course of a network transaction over time. Arrows indicate packets travelling the network between client and server. Time increases as you descend, with successive events' times noted as TN, where T1 represents the first event and T22, the last event. Once PacketShaper determines the server delay and the total delay, it can calculate the amount of time the transaction spent in transit, using the following formula:

$$\text{Network delay} = (\text{Total delay}) - (\text{Server delay})$$

IV. VISUALISING APPLICATION PERFORMANCE

Information visualisation (IV) is defined as "the use of computer-supported interactive, visual representations of data to amplify cognition, where cognition is the acquisition or use of knowledge" [7].

The purpose of IV is to gain insight into complex, unexplored data. This process involves the discovery, decision-making and explanation of hidden relationships and patterns within the data. IV can show the big picture by providing an overview of the essential relationships between the data [8].

There are several tools in industry today that monitor the performance of network applications, including Network Vantage [5], SuperAgent [9] and ReportAnalyser [10]. A comparative analysis was performed of these three existing systems and the results were analysed in terms of application delay analysis, visualisation techniques used, interaction techniques available and user interface design. The major limitations that were identified across all the systems were that they used basic two-dimensional graphs for displaying application delay metrics. No interactive graph viewing capabilities were present for in-depth analysis and exploration through the visualisation space. No zooming, panning or rotation mechanisms were present in the visualisation process. The user interface of the systems incorporated single views and resulted in a lot of scrolling. The systems were mainly web-based and slow.

To develop effective and efficient visualisation tools to help users better understand application performance of a network, a user-centred approach to network visualisation is needed. User tasks need to be identified and visualisation techniques designed to support these [11]. Shneiderman [12] presents seven abstract tasks that users normally perform in information visualisation: overview, zoom, filter, details-on-demand, relate, history and extract, which he has incorporated within his visual information mantra. These guidelines were used in the development of the AppVis visualisation system and are discussed in the next section.

V. SYSTEM ANALYSIS

Data Analysis

The application delay components measured by PacketShaper can be represented using a multidimensional model in the form of a snowflake schema using dimensions and facts (Figure 2). The dimensions that are defined within the schema are *Time Period*, *VLAN* and *Department*. The dimensions are associated with a single fact known as a *Delay Transaction*. The schema contains a large central fact table associated with a *Delay Transaction* and a set of smaller dimension tables defined for each dimension respectively. The fact table contains the bulk of the application delay data to be visualised with no redundancy occurring within its records. It contains keys to each of the two dimension tables *VLAN* and *Time Period*, along with three measures, including *network_delay*, *server_delay* and *total_delay*. This allows for a unique application delay record to be measured across any VLAN over a certain time period. Each dimension table contains a set of attributes that describes its associated dimension in more detail.

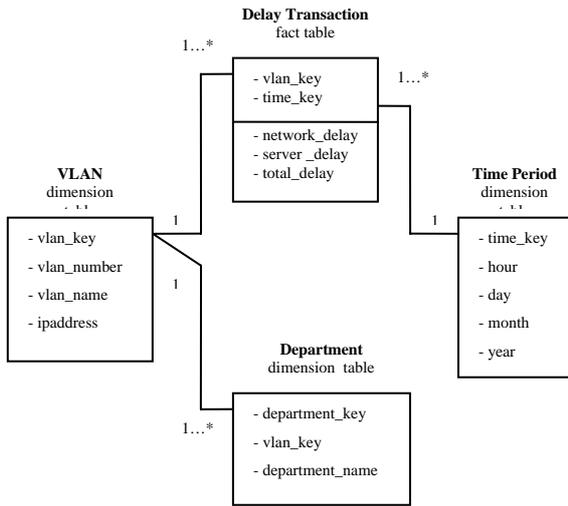


Figure 2: Snow-Flake Schema

As illustrated, a *Delay Transaction* can record application delay data for one *VLAN* at a time. However, a *VLAN* can be associated with more than one *Delay Transaction*. The same can be concluded for a *Time Period*. A *Department* dimension table is defined that is associated with a *VLAN* to cater for the complexity where certain departments share an individual *VLAN*. Since more than one department can belong to the same *VLAN*, *Department* has been normalised into a separate dimension.

Functional Analysis

The three main functions that the system needs to support are the importing of application performance metrics collected by PacketShaper, the filtering of the delay components found within the data and the analysis and exploration of the resultant data. The system's functional requirements are as follows:

- Import application delay metrics;
- Display a network overview of total application delay;
- Display the application delay components for an individual VLAN;
- Display the performance trends of application delay components for an individual VLAN;
- Filter application delay components based on specified periods, VLAN and delay statistics;
- Perform analysis and exploration of application delay components, namely zooming, panning and details-on-demand; and
- Print and save the generated system graphs.

VI. SYSTEM DESIGN

Functional Design

The system functionality is partitioned into seven functional modules that will support the tasks of the system. The identification of each module and its interactivity with other modules is illustrated in Figure 3.

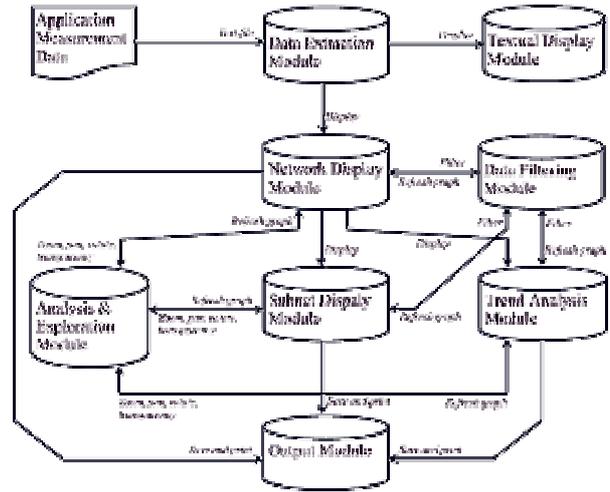


Figure 3: Functional architecture of AppVis

The *data extraction module* is identified as being the first functional component within the architectural design of the system. It is an operational procedure that implements an extraction algorithm that parses the application performance data, found within the configuration file and the PacketShaper measurement file, into the system database. This module will be responsible for performing aggregation operations on the delay components based on different time granularities across different VLANs while they are being loaded into the database. This procedure will assist in structuring delay components in a more compatible form to be displayed on the system graphs.

Once the data extraction module has parsed the application performance data from the measurement file, the system architecture automatically activates the *network display module* and *textual display module* simultaneously. The *network display module* dynamically generates a network overview of the total transaction delay for all VLANs stored in the system database across a default performance period. The *textual display module* is responsible for generating a textual view of the total transaction delay of each VLAN.

Once the user has performed a drill-down operation within the network overview, the system architecture will activate either the *subnet display module* or *trend analysis module* depending on whether the user is interested in analysing the associated delay components of an individual VLAN or analysing the associated delay trends of that VLAN. The *subnet display module* is responsible for implementing a display algorithm that generates a graph of the delay components of the selected VLAN by receiving the corresponding VLAN key as a parameter. The *trend analysis module* follows the same procedure.

The *data filtering module* is activated by the system architecture once the data extraction module has terminated and the application measurement data is stored in the system database. This module allows queries to be made on the measurement data being displayed in any of the system generated graphs based on VLAN, delay components, delay statistics and performance periods.

The *analysis and exploration module* is activated by the system architecture once the *network display module*, *subnet display module* or *trend analysis module* has been activated.

It incorporates interactivity procedures for zooming in and out of the visualisation space, panning across the visualisation space and performing rotation and transparency filtering on the system graphs to avoid or limit data occlusion.

The *output module* is activated by the system architecture if any of the graphs generated by the system is required to be exported out of the system. The module is responsible for saving and printing system graphs by implementing procedures for printing the respective graph to an available printing device or saving it as an image in a Windows Bitmap format to be used in other applications.

User Interface (UI) Design

An iterative design approach was used in the UI design of the AppVis visualisation system. The design process focused on producing low-level prototype designs of the user interface for evaluation by the users. This iterative process allowed the low-level designs to be refined based on feedback received from the users into high-level designs.

The UI of the system was designed to enable the viewing of application delay components measured across different VLANs on the UPE network over specified periods. The system interface as illustrated in Figure 4 shows the visualisation technique that displays a network overview of application total delay after the user has imported an application measurement file into the system database. Figure 4 also illustrates the multiple view layout of the system interface. The interface is divided into three main views, including a graphical view, a textual view and a filtering view. These views remain fixed during any visualisation analysis and exploration procedures that are performed in the system.

The network overview graph that is displayed in the graphical view of the system represents the total application delay measured across different VLANs over a user specified time period. Each node displayed on the graph represents a defined VLAN. The name of each VLAN is displayed as an associated label attached to the respective node. Two edges run between each subnet displayed on the graph. A solid edge represents the mean total transaction delay for the corresponding VLAN while the dashed edge represents the maximum total transaction delay for the VLAN. The delay values that both edges represent are colour coded according to a standardised colour scheme displayed in the graph legend. The colour scheme is categorised further into delay intervals that describe various levels of delay criticality, including low, regular, minor, major, warning and critical respectively. The user will be able to drill-down to analyse the delay components of an individual VLAN by double clicking on its corresponding node. The total transaction delay displayed across the VLANs can be viewed dynamically over various time periods by shifting the handles of the slider control located below the graph to various time positions.

Filtering can be performed on delay components being visualised in the graphical view of the system based on VLANs, delay components including server, network and total delay, delay statistics including mean and maximum delay and various periods by specifying a start date and ending date. The textual view displays the total mean and maximum transaction delays for each visualised VLAN in

adjacent columns. It provides a textual representation of the total transaction delay being displayed in the graphical view of the interface.

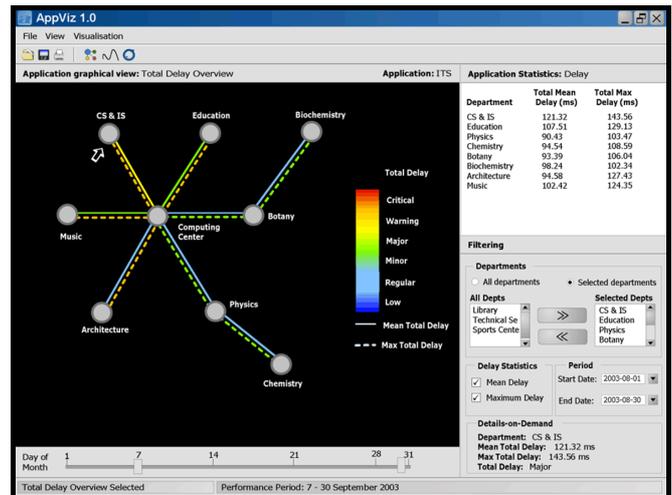


Figure 4: Network overview of application total delay

The system interface illustrated in Figure 5 shows a spiral graph that is used for trend analysis in the system to identify the performance patterns of delay components. The graph visualises server delay for the Embizweni VLAN for September 2003.

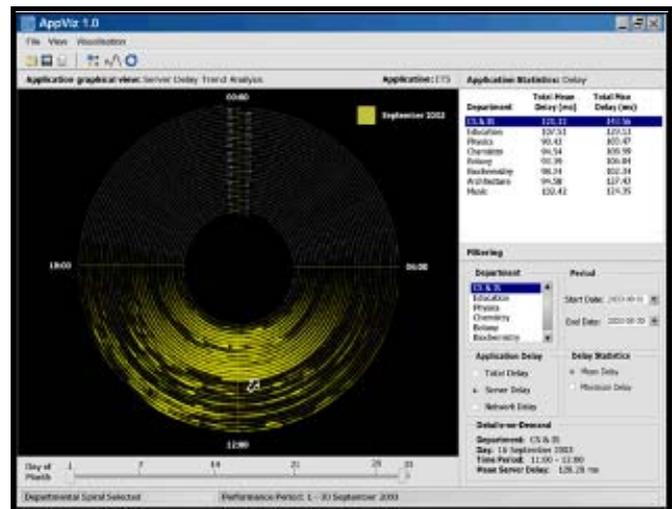


Figure 5: Trend analysis view of delay components

Each revolution of the spiral represents server delay across one day of the month starting from the centre of the graph and culminating outwards at the border of the spiral. The revolutions are partitioned into hour intervals visualising server delay across each day of the month. Line segments within a revolution that are drawn thicker in width indicate a greater delay across that time period. As illustrated, the server delay peaked across most days of the month between nine am and four pm. The graph will be able to visualise more than one delay component at a time by incorporating additional spiral revolutions running next to one another across user specified periods. Each spiral revolution will be colour coded using a different colour.

The AppVis system will allow the user to save and print the graphs generated in the graphical view as image files. The graph images will be saved in a Windows Bitmap

(*.bmp) format for insertion into any presentation document.

Data Design

The application delay components found within a PacketShaper measurement file are arranged in a tabular form, as illustrated in Table 1. The delay components of *network delay (nd)*, *server delay (sd)* and *total delay (td)* are shown for three VLANs, including *main building 16th floor*, *main building 11th floor* and *main building 6th floor* respectively and aggregated over a three month period.

VLAN	Main_building_16			Main_building_11			Main_building_6		
	delay component			delay component			delay component		
time period	nd	sd	td	nd	sd	td	nd	sd	td
Jan	273	4	277	334	2	336	227	1	228
Feb	254	3	257	321	3	324	256	2	258
Mar	266	2	268	289	1	290	283	1	284

Table 1: Example of PacketShaper measurement file

Due to the large amounts of delay data collected from PacketShaper as well as the data being time-series based, a database was developed to store the application performance data and provide quicker query processing and information retrieval. A relational model of the AppVis system is illustrated in Figure 6.

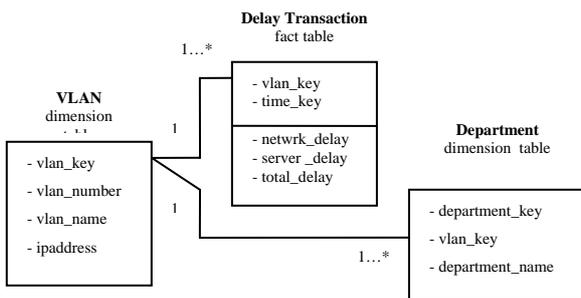


Figure 6: Relational Model of AppViz

The *Time Period* dimension was removed as it is regarded as a primitive key. Its functionality has been folded into the *Delay Transaction* fact table where roll-up and drill-down operations can be performed for delay components based on the *time_key* attribute listed in the table.

VII. IMPLEMENTATION

Implementation tools

The implementation phase of the system involved performing a feasibility study into various implementation tools for implementing the visualisation techniques of the AppVis system. C#.NET [13] and Java with Java3D [14] were investigated as the possible implementation tools to use for system development. Various high-level graphical and charting components such as Nevron [15], that are compliant with C#.NET, were also investigated to determine whether these could assist with the development of the system's visualisation techniques. No suitable charting components for Java were found that could be used for development purposes. Visualisation prototypes of the techniques were developed using certain Nevron charting components to determine whether the graphing capabilities

offered the desired interaction and analysis mechanisms needed for development. The feasibility study concluded that Nevron had a substantial amount of charting mechanisms for developing highly interactive three-dimensional graphing components using C#.NET and it was therefore chosen for implementation purposes.

Implementation techniques

After the initial UIs in Figures 4 and 5 were designed they were evaluated by network administrators with respect to the visualisation and interaction techniques developed. Implementing the network node metaphor (Figure 4) was not feasible due to certain data and implementation constraints. PacketShaper cannot measure application delay metrics across individual departments configured within the same VLAN due to the switched nature of the UPE network. The implementation constraints identified were that the node labels and edge colours had to be developed manually and were not consistent with the other graphs developed.

A star visualisation technique was then proposed to display the network overview of the application total delay plotted for each VLAN, across an associated axis emanating from a central point of the graph as illustrated in Figure 7.

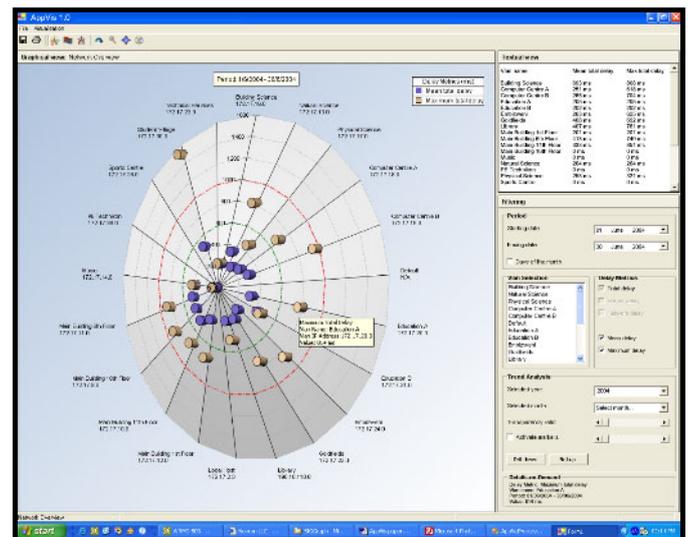


Figure 7: Star visualisation of application total delay

The further the total delay is plotted from the origin the higher its value reads. Therefore, VLANs having delay plots further from the origin will indicate high delay values and those whose delay plots occur close to the origin will indicate low delay values. Threshold bands are displayed on the graph as concentric rings to show the delay criticality of each VLAN on the graph. The graph is responsible for plotting the mean and maximum total delay for each VLAN during a user specified period. The plots are event-enabled allowing the user to drill-down and view the delay metrics for an associated VLAN.

The spiral visualisation technique (Figure 5) was found not to be effective in displaying application delay trends across time periods. The inner and outer rings of the spiral are not of the same length when comparing individual hours across days of the month, resulting in data that could be skewed. It would also be harder to zoom into the graph for hour comparisons across various days of the month. The

graph occupies large amounts of screen real estate when too many delay components are visualised as adjacent spirals.

The system interface illustrated in Figure 8 shows a radar graph that is proposed for the visualisation of trend analysis in the system. The graph allows network, server and total delay to be plotted on top of each other along a series of axes representing various time intervals that can accommodate for the days in a month and the months in a year. This occupies less screen real estate and allows for a clearer comparison between delay components across various time periods

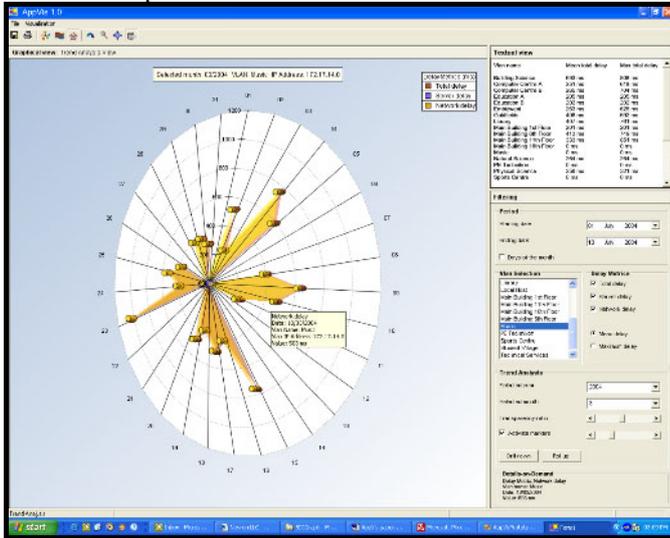


Figure 8: Trend analysis view of delay components

Figure 8 illustrates that the server and network delay for ITS over the Music VLAN peaked during the first half of March 2004 and tapered off during the end of the month. The user will also be able to analyse and explore the graph interactively by zooming, panning and rotating the graph in its visualisation space. The user will also be able to use transparency filtering on the plotted delay components to avoid data occlusion. Details on demand can be dynamically displayed for each delay plot by performing a mouse over operation.

VIII. EVALUATION

The evaluation of the system will determine the usability of the system in terms of its supported tasks. The usability metrics that will be measured are the effectiveness and level of user satisfaction with which the system visualises application delay metrics. Informal user testing will take place where users will be given questionnaires to evaluate the system based on several heuristics. System changes will be implemented based on the qualitative results obtained from the evaluation, thus supporting the iterative design process of the system's development.

IX. CONCLUSION

APM is essential to enable organisations to monitor network-dependent applications. UPE currently has no effective or efficient means of visualising the performance of the application services implemented on the university network. A more flexible and interactive information visualisation system to visualise and explore application delay of the UPE network is needed. The application delay metrics to be visualised, namely server, network and total

delay, will be collected by means of a network monitoring technology, called PacketShaper. An iterative design approach was adopted for the development of the system where feedback was received from the users to refine the system functionality and UI. This process resulted in the design of novel interfaces to visually represent application delay metrics in a dynamic and interactive manner (Figures 7 and 8).

REFERENCES

- [1] Erbacher, R.F. (2001): Visual Traffic Monitoring and Evaluation, Department of Computer Science, University of Albany, Albany.
- [2] eSoft Technologies Ltd. (2003): The Solutions Provider. URL: <http://www.esoftech.com>
- [3] Garg, A. and Schmidt, R. (1999): Network Management: Is it the Network or the Application. URL: <http://www.comnews.com>
- [4] Chataut, S., McClellan, S. and Gemmill, J. (2002): Tools for Application Performance Management, University of Alabama, Birmingham, Alabama.
- [5] Compuware Corporation (2003): NetworkVantage: Isolate Network and Application Performance Problems Quickly. URL: <http://www.compuware.com>
- [6] Packeteer, Inc. (2002): Response-Time Technology, Packeteer Technical White Paper Series. URL: <http://www.packeteer.com/>
- [7] Card, S.K., Mackinlay, J.D. and Shneiderman, B. (1999): Readings in Information Visualisation: Using Vision to Think. Morgan Kaufmann Publishers, Inc., San Francisco, California.
- [8] Young, P. (1996): Three Dimensional Information Visualisation, Computer Science Technical Report No 12/96, Centre for Software Maintenance Department of Computer Science, University of Durham.
- [9] NetQoS SuperAgent (2003): SuperAgent: Monitor application response times across the network. URL: <http://www.netqos.com/solutions/superagent/>
- [10] NetQoS ReportAnalyser (2003): ReportAnalyser: The Netflow Solution. URL: <http://www.netqos.com/solutions/reporter/>
- [11] Carr, D.A. (1999): Guidelines for Designing Information Visualisation Applications, Proceedings of 1999 Ericsson Conference on usability Engineering.
- [12] Shneiderman, B. (1996): The Eyes Have It: A Task by Data Type Taxonomy for Information Visualisation, Proceedings of 1996 IEEE Visual Languages, pp. 336-343.
- [13] Microsoft .NET Framework (2004): Microsoft Visual C#.NET. URL: <http://msdn.microsoft.com/vssharp/>
- [14] Sun Microsystems (2004): Java Technology. URL: <http://www.java.sun.com/>
- [15] Nevron LLC (2004): Enterprising Charting Solutions. URL: <http://www.nevron.com>

BIOGRAPHY

Justin Rademan received his B.Sc Hons degree in 2002 from the University of Port Elizabeth. He is presently doing his M.Sc in Computer Science at the University of Port Elizabeth. His current field of research involves using new information visualisation techniques to visualise the application performance of a customer network.