# Network Visualization for Management Purposes

**Christos Tsoleridis, Panayotis Fouliras**

Department of Applied Informatics, University of Macedonia
{xtsoler@gmail.com, pfoul@uom.gr}

## Abstract

In this paper, we present *JNodes*, an open source network monitoring application, which visualizes networks of any size in an enriched form of nodes and arcs. Our approach focuses on displaying the full prospect of a network based on current data: The network administrator can determine the source of the problem efficiently, skipping some of the initial steps in network investigation such as ping requests and telnet sessions to remote entities, having at his disposal a good overview of all of the available data flow graphs. Another advantage of JNodes is that it is written in the Java programming language, allowing access to the displayed data not only through a Java enabled web browser, but also through a Java enabled handheld device (e.g., a common cell phone). This latter aspect of the design expands its usefulness and improves upon the critical reaction time often sought for by network administrators. A complete prototype has been built, tested and is available for demonstration.

**Keywords:** Network Management, Routing, Data flow, Visualization, Mobile.

## *1. Introduction*

The popularity and complexity of computer networks and related services has increased the need for powerful, yet friendly tools to support administrators in their effort to monitor their status and assist in the efficiency of their management. Some of the most common and hard to diagnose problems in a TCP/IP network are congestion, data loss, delay and insufficient bandwidth availability [Hong et al 1997]. Tools capable of indicating such issues are mainly based on data collected intensively. Great efforts have been made to design smart agents that fully automate procedures to track and prevent possible problems on a network. Factors taken into consideration are user needs and habits in a typical network, the available resources such as active entities and bandwidth, as well as the ever present factor of human error. Nevertheless, the effective production of a real time representation of the entities comprising a typical network has not advanced significantly.

In this paper, we present JNodes, an open source network monitoring application, which visualizes networks of any size in an enriched form of nodes and arcs. Our approach focuses on displaying the full prospect of a network based on current data:

The network administrator can determine the source of the problem efficiently, skipping some of the initial steps in network investigation such as ping requests and telnet sessions to remote entities, having at his disposal a good overview of all of the available data flow graphs. JNodes provides a concise map of the managed network with arrows on the links showing not only the traffic rate, but also its direction. This is a unique characteristic of JNodes.

One major advantage of JNodes is that is written in the Java programming language, allowing access to the displayed data not only through a Java enabled web browser, but also through a Java enabled handheld device (e.g., a common cell phone).

The rest of the paper is organized as follows. The related work is presented in the next section, followed by a detailed presentation of JNodes. This is followed by an evaluation of JNodes, also comparing it against some of the most popular network management tools available. Finally, the conclusions and future work are outlined in the last section of this paper.

## *2. Related Studies and Implementations*

### *2.1 Related Work*

Many studies have been conducted regarding applications for network management. [Hong et al (1997)] present an integrated architecture that can be used to manage all the networking and computing resources in an Intranet, focusing on Web based management. [Bieszczad et al (1998)] describe several actual and potential applications of mobile agents in the five OSI functional areas of Network Management. [Raz and Shavitt (1999)] describe a novel active network architecture, which primarily addresses the management challenges of modern complex networks. Its main component is an active engine that is attached to any IP router to form an active node. [Hong et al (1999)] present the design and implementation of a portable, Web-based network traffic monitoring and analysis system called WebTrafMon.

[Brutlag (2000)] outlines a model and software, which work within the architecture of the existing open-source solutions RRDtool and Cricket. He suggests integrating a mathematical model for automatic aberrant behaviour detection in time series into the monitoring software. [Gardner et al (2002)] present Magnet (Monitor for Application-Generated Network Traffic) – a toolkit that captures traffic generated by the application, rather than traffic in the network.

[Kang et al (2002)] present a method and system design for monitoring and analyzing streaming media traffic over RTSP and MMS. Their approach analyzes streaming control messages and extracts information for transferring streaming media data.

[Moore et al (2003)] describe a system for simultaneously monitoring multiple protocols, which perform full linear capture and implements on-line analysis and compression to record interesting data without loss of information. In another paper [Chen et al (2004)], an algebraic approach was proposed that selectively monitors $k$ linearly independent paths that can fully describe all $O(n^2)$ paths. Another interesting study has emanated from the project "A Hybrid Honeypot Architecture for Scalable Network Monitoring" [Bailey et al (2004)]. In the proposed architecture, multiple tiers of sensors combined with a distributed indication and warning system can provide intelligence on brewing threats in a network. This project evolved into another project: "The Internet Motion Sensor: A Distributed Blackhole Monitoring System" [Bailey et al (2005)], where the Internet Motion Sensor (IMS) was introduced – a globally scoped Internet monitoring system whose goal is to measure, characterize, and track threats.

## *2.2 Related Implementations*

As stated above, some of the most common and hardest problems to diagnose in a TCP/IP network are congestion, data loss, delay and insufficient bandwidth availability. Tools capable of indicating the emergence of such issues are mainly based on data collection in an intense way. Considerable efforts have been made to design smart agents that fully automate procedures to track and prevent possible problems on a network. Major factors taken into consideration are the needs and habits of the users of a typical network, the available resources of both hardware and bandwidth costs and limitations, and the typical human error.

There are many products today offering network monitoring in one way or another. Nevertheless, some of these are particularly popular, thus deserving more attention. These are *CiscoWorks* from Cisco Systems Inc, *HP OpenView* from Hewlett-Packard and *The Dude* from Mikrotik Ltd. After studying the range of features available in these tools, it is possible to summarize the typical features considered necessary for present-day network topology visualization. These features allow the following:

- Display the network structure.
- Display the current state of every monitored network entity.
- Display the delay and availability graphs of every monitored network entity.

More specifically, Cisco offers a collection of tools that provide innovative concretizations of central management of network characteristics in CiscoWorks such as availability, resistibility and safety. Furthermore, CiscoWorks can be organized in several categories according to the type of management required, namely management of entity and infrastructure, IP telephony, and management of security and identity. In terms of its architecture, CiscoWorks is a typical client-server application, utilizing agents running on Cisco appliances.

HP Open View is the version of Hewlett Packard for the depiction and management of a network, capable of visualizing large-scale networks. The same company manufactures and allocates network appliances such as switches (e.g., the HP Procurve series switches) and servers for scaled network services (e.g., the HP Proliant server series). The Network Node Manager visualizes not only the network links, but the network traffic, as well. It guarantees that the administrator will have a more explicit picture of network services and will be able to minimize the cost of maintenance and network extension more easily. Apart from the HP network products, the Network Node Manager is able to operate on any other manufacturer's network appliances. The main characteristic of the OpenView Performance Insight is that it provides the administrator with more precise measurements and details regarding a network node when a problem is identified. Two equivalents tools are the OpenView Network Configuration Manager and the OpenView Route Analytics Management System.

Finally, "The Dude" performs a scan of a WAN, creating a map of the target network with its routers and servers as well as any other network nodes, including the respective links of these nodes. It extracts information on the volume of data that is flowing in the network using the SNMP protocol and then it impresses this movement in each link in real time. Moreover, the user can program numerous operations that depend on the state of some node or network section. For example, it can automatically send a notice to the administrator of the router if some links have been interrupted. Furthermore, it records link traffic and produces graphs to estimate the sufficiency of bandwidth for particular links. The links that present high traffic volume turn red in order to have a wider prospect of traffic flow in the network. All connected users are required to have an account in the related server and it is possible to have more than one administrator managing the produced map concurrently.

A common observation in all cases is that there is a major effort towards the effective production of real time representation of the elements that form a typical network. Nevertheless, targeted depictions of both of the nature of the methods used in network management as well as towards proper training of smart agents are very important. It is, therefore, necessary to advance network visualization further.

## *3. The JNodes Implementation*

JNodes was designed with two main goals that are also the distinguishing factors from all the other relative network management implementations: The first one is visualizing as much information as possible on the network map. A key feature, unique in JNodes, is the existence of directed arrows in the middle of each link (as shown in Figure 1). This feature replaces the classic rx/tx indications in other implementations, which are insufficient when one needs to know the traffic direction.

The second goal is accessibility. More specifically, a user can access the network map using the Internet, provided Java is available on his device. Today Java is available in almost every mobile device. Client-server communication in JNodes is implemented with http compatibility, thus fooling the web proxy that typically is set up by the cell phone service providers to guard against unwanted mobile internet access. Consequently, a typical cell phone can act as the administrator’s client device any time, anywhere.

JNodes depicts network entities in a form of nodes and arcs. However, JNodes is not designed for the absolute beginner: The user must have sufficient knowledge of the monitored network and have some general experience in networks and related tools to be able to handle all the projected data, although the required experience is minimal.

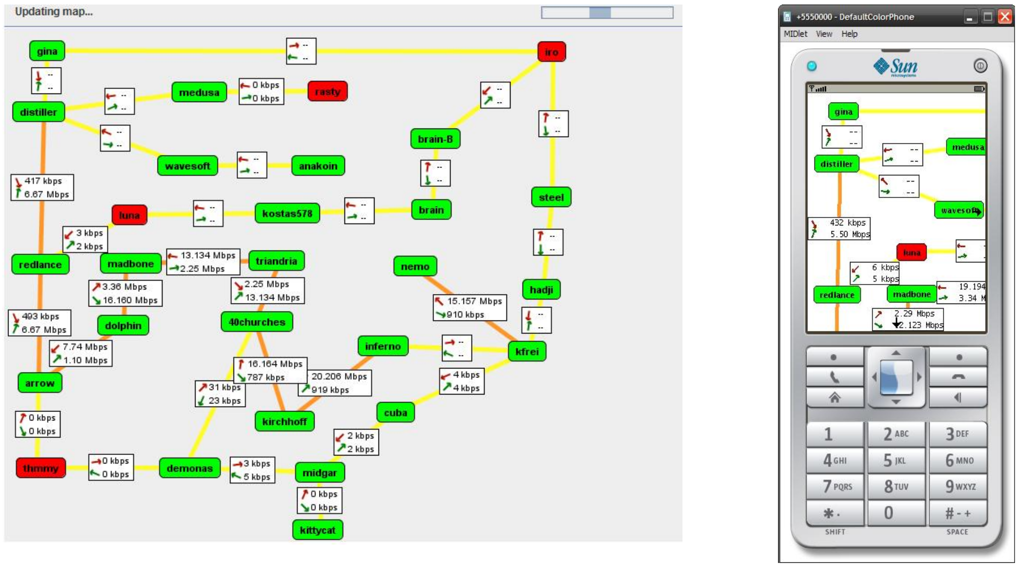
Furthermore, given that JNodes is fully programmed in Java it is not only straightforward for a user to gain access to the displayed data through a common web browser or even a Java-enabled handheld device, but also directly competitive to other Open Source alternatives. In short, the key features of JNodes are:

- A new complete Open Source implementation of Network Monitoring for Management.
- Easy to use application controls for data entry.
- Trivial procedure for gaining access to the displayed data.
- Simple, obvious and comprehensive graphical representation.
- Installation-free and multiplatform application.

JNodes is composed from three separate programs, all built in Java using the Netbeans IDE:

- The JNodes Server that manages all node data, thus collecting traffic information among nodes via SNMP and storing each change attempted by any client on the data map.
- The JNodes Applet Client that displays all data collected by the server and provides the user with the capabilities of altering, removing or adding elements in the displayed map.
- The JNodes MIDlet Client that displays all data collected by the server, on any handheld appliance compatible with Java runtime.

Using JNodes is quite straightforward. Once the server is started, JNodes is ready to receive both instructions for the monitored network entities, as well as the relevant data from the SNMP agents on the nodes. Then the user can start the appropriate client, which can be either an applet or a MIDlet depending on the operating environment.

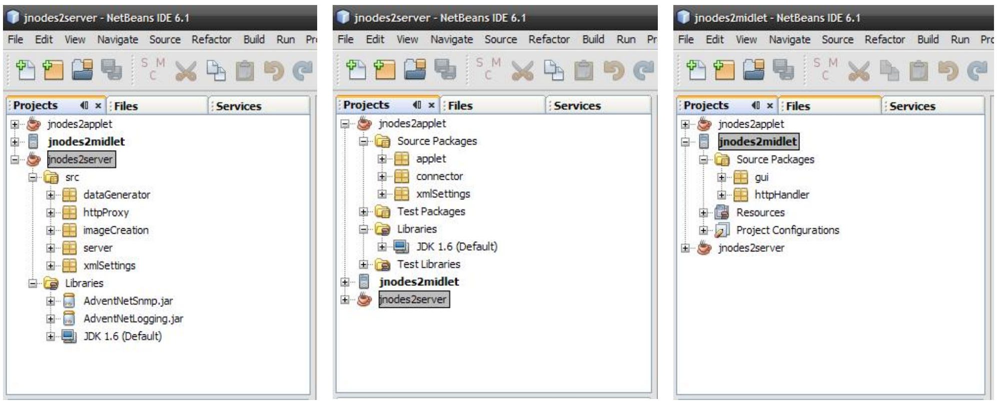


(a) (b)

***Figure 1****. Netwrok Map using: (a) an Applet, (b) a MIDlet*

For illustrative purposes, let us assume that the user initiates the applet client. The client first receives an xml file containing the current network map from the server. In the simplest case, this map is initially empty. By right clicking on it, the user can add a new node, which plays the role of a “root” graph node in the sense that the rest of the network is depicted gradually by adding the neighbors of the “root” node. The information for a new node comprises of a symbolic name the user chooses, its IP address and the respective SNMP community (typically “public”). Initially a new node is red, turning into green as soon as it replies to a ping request sent by the server. It is also possible to right-click on an existing node in order to delete, edit, inspect its information (such as a list of its network interfaces) and add a link to it. The latter allows the establishment of a link between two existing nodes.

Once such a link is created, a pop-up window appears in which the user selects the SNMP data source from the two nodes in the link and the interface of the particular node corresponding to that link. The link appears in the map, together with a small frame with two entries showing the network traffic on the link. Each entry has two fields: an arrow (red for the SNMP source) showing the traffic direction and the respective traffic rate. The traffic depicted is always real-time as collected by the server. An example appears in Figure 1. This approach has the advantage of being simple for a user to follow as well as quite selective on the network portion the user wishes to monitor. The operation presented above is the basic form for JNodes and is similar if a MIDlet is used as a client. There are other secondary yet useful functions, such as zoom in/out, panoramic network view, etc.



(a) (b) (c)

***Figure 2**. Concise Packages Description: (a) Server, (b) Applet, (c) MIDlet.*

### *3.1 JNodes Server*

The JNodes server is divided in five major packages (Figure 2a). In total, there are 3567 lines of code and two external libraries. The five major packages are outlined below:

- The *dataGenerator* package collects all the classes that complete the SNMP and ping requests.
- The *httpProxy* package implements a custom http server able to feed every client with data in byte array format. The client can be either a Java Standard Edition applet launched within any web browser or a Java Micro Edition MIDlet launched on any Java compatible handheld device. The main reason for this seemingly redundant implementation is that cell phone carriers typically restrict internet connectivity using a proxy. By building a custom http proxy, it is possible to circumvent such restrictions in internet.
- The *imageCreation* package is responsible for the map image creation. When serving a handheld device, the server has to prepare the image and send it as a byte array. On the other hand, when the server serves an applet client, the data is sent in xml format and the applet draws the map for itself.
- The *server* package is the main package that ties all the other packages together.

- The *xmlSettings* package takes care of the xml related procedures. Since all map data and metadata are stored in xml files, there are classes that concentrate the procedures for each file type. There are also classes to assist in querying xml files and to write them out to the disk when necessary.

### *3.2 JNodes Applet Client*

The Applet Project file is divided in three packages. There are 2733 lines of code in total and two external libraries, the latter being the same as for server project (see Figure 2b):

- The *applet* package is the most important. It is able to draw the given data for the user by using the other two packages. It utilizes a Java awt canvas with drag and drop functionality upon the drawn elements of the map.
- The *connector* package takes care of all the network related procedures, which – in the case of the applet – are exclusively http requests, both in polling mode and by user invocation.
- *xmlSettings* is the third package, with xml functionality. Since map data is stored in xml files, the client needs to read, query, alter and write out xml files on the fly.

### *3.3 JNodes Midet Client*

The last project in the JNodes implementation is the MIDlet client. There are 1073 lines of code, organized into two packages (see Figure 2c) with no external libraries, since there is no xml manipulation at this stage. MIDlet clients receive an already drawn image or the part of the image that was modified since the last update and simply print that image on the screen. The two packages are as follows:

- The *gui* package contains a simple form (a canvas) and some basic controls such as zoom out of the map and print console info on the screen. There is also a user driven update trigger for debugging purposes.
- The *httpHandler* package is a Java ME http client that polls the server for new map data.

## *4. JNodes Evaluation*

JNodes is an Open Source application and as such, it offers a significant advantage over commercial ones. Nevertheless, evaluating it in comparison to other similar applications is not straightforward because either the other applications are not freely available or the monitored network is not the same. For this reason, we present only a simple, qualitative comparison between JNodes and “The Dude”, which is another freely available network monitor application. The criteria for our evaluation are: Data collection speed, accessibility, application performance, portability and number of simultaneous users allowed. For both cases, the Thessaloniki metropolitan wireless network was used as a test bed.

Data collection speed is proportional to the SNMP echo requests for both applications; hence, there is no difference in terms of this criterion.

Accessibility here determines the capability of an application to gather data using any device as the client used by the administrator. In this respect, JNodes is superior, since it can be used from a cell phone and even behind a proxy, as opposed to "The Dude" which is limited to socket connection communication.

In terms of actual performance, we tested both applications loading their respective maps with a large number of nodes. "The Dude" was better in the sense that the map was loaded faster; JNodes refreshes the entire map at every client request in order to preserve reliability should a request be missed accidentally – something necessary for connections behind proxies. On the contrary, "The Dude" updates the map only for any new node, something easy when using normal, open socket connections, but very difficult when proxies are involved as in the case of cell phones.

In terms of portability, JNodes is obviously superior because it is completely written in Java, which allows it to run even on simple cell phones, whereas "The Dude" is limited to Microsoft Windows and recently to Linux systems.

Finally, in terms of the number of simultaneous users allowed, both applications are capable of serving large amounts of users without any problem. The relevant data is delivered to all users immediately and the change in the elements of the map is handled with safety.

## *5. Conclusion*

JNodes is Open Source software, completely written in Java for studying, monitoring and analyzing TCP/IP networks. It allows the user to design an appropriate network map and display useful real-time information regarding network traffic and parameters. JNodes can be used by network administrators as well as for a general study of a computer network for educational purposes. It works by using the live feed of traffic information, utilizing the SNMP daemon on every network node. Furthermore, it can be used from any Java-capable device, even a typical cell phone.

JNodes provides a stable and user-friendly yet concise environment with adequate performance. Another important characteristic is that it can easily be extended, augmenting the range of its features. A complete prototype was built and tested with success.

## 6. References

1. Andrew Moore, James Hall, Christian Kreibich, Euan Harris, and Ian Pratt (2003), *Architecture of a Network Monitor*, in Passive & Active Measurement Workshop 2003 (PAM2003).
2. M. Bailey, E. Cooke, F. Jahanian, J. Nazario, and D. Watson (2005), *The Internet Motion Sensor: A Distributed Blackhole Monitoring System*, in Proceedings of the Network and Distributed Security Symposium, San Diego, CA, pp. 167-179.
3. M Bailey, E Cooke, D Watson, F Jahanian, and N Provos (2004), *A hybrid honeypot architecture for scalable network monitoring*. Technical Report CSE-TR-499-04, University of Michigan.
4. Bieszczad, B. Pagurek, and T. White (1998), *Mobile Agents for Network Management. IEEE Communications Surveys*, **1**, No. 1, pp. 2-9.
5. Brutlag J. (2000), *Aberrant behavior detection in time series for network monitoring*, USENIX System Administration Conference, New Orleans, pp. 139-146.
6. Y. Chen, D. Bindel, H.Song and R.Katz (2004), *An algebraic approach to practical and scalable overlay network monitoring*, ACM SIGCOMM Computer Communication Review, **34**, 4, pp. 55-66.
7. Danny Raz and Yuval Shavitt (1999), *An Active Network Approach to Efficient Network Management*, IWAN'99, LNCS 1653, pp. 220–231.
8. Gardner, M., W. Feng, and J. Hay (2002), *Monitoring Protocol Traffic with a MAGNeT*, Passive and Active Measurement Workshop, (PAM2002), Fort Collins, Colorado, pp. 110-115.
9. James W. Hong, Soon-Sun Kwon and Jae-Young Kim (1999), *WebTrafMon: Web-based Internet/Intranet Network Traffic Monitoring and Analysis System*, Computer Communications, Elsevier Science, Vol. 22, No. 14, pp. 1333-1342.
10. W. K. Hong, et al (1997), *Web-based Intranet Services and Network Management*. IEEE Communications Magazine, pp. 100-110.
11. Hun-Jeong Kang, Hong-Taek Ju, Myung-Sup Kim and James W. Hong (2002), *Towards Streaming Media Traffic Monitoring and Analysis*, Proc. of 2002 Asia-Pacific Network Operations and Management Symp. (APNOMS 2002), Jeju, Korea, pp. 503-504.
12. Lopes R. P. and Oliveira J. L. (1999), *Software Agents in Network Management*, Proc. of the 1st International Conference on Enterprise Information Systems – ICEIS'99, pp. 674-681.
13. Oetiker, T. (1998), *MRTG—the multi router traffic grapher*, LISA XII, Boston, MA Proceedings of the Twelfth Systems Administration Conference (LISA '98), pp. 141-148.