3-1-2006

Analysis and Design of Computer Networks Using a Web-Based DSS

Dale R. Fox
zai.fox@esabocra.com

Richard V. McCarthy

Raymond Papp

ANALYSIS AND DESIGN OF COMPUTER NETWORKS USING A WEB BASED DSS

Dale R. Fox
esabocra.com
zai.fox@esabocra.com

Richard V. McCarthy
Quinnipiac University
richard.mccarthy@quinnipiac.edu

Raymond Papp
University of Tampa
rpapp@ut.edu

Abstract

Analyzing the performance of computer networks to improve their overall performance requires accounting for a large variety of inputs and operating characteristics, such as packet flow rates and sizes, device capabilities and interconnection capacities. A decision support system to analyze these multiple simultaneous variables is presented to assist network developers. Network architectures and demand patterns are accounted for and the underlying model that analyzes these patterns is discussed. A web based approach was utilized which has resulted in an accessible and scalable analytical tool.

Keywords: DSS, networking design and analysis

Introduction

The analysis and design of computer networks requires differentiating and coalescing a huge variety of issues. Considerations of what will truly contribute to an organization's functionality and competitiveness need to be at the source of all designs. Spending the effort to develop the requirements involved in providing appropriate services and security to system users needs to be developed before any sort of more involved analytical and computational tools can be invoked.

But once certain system requirements are developed it is also essential for network designers to determine things such as, required throughputs, bottlenecks, reliability and quality of service all depending on current and anticipated network design and traffic. Collecting existing data on network performance is necessary, particularly when it applies to required services and infrastructure that will be carried over into new designs. But the large variety of issues that impact computer network performance cannot be easily accounted for even while making use of standard modular design approaches.

This paper will give some background on some standard design models and then present a mathematical model that will be used as the basis of a DSS to help a network designer investigate the impacts of particular designs on overall network performance.

Background and Literature

Teare (2004) is the basis of the approaches developed for Cisco certification in designing internetworks. Her models are quite abstract in nature but give a big picture and overall context to the designer. In efforts to deal with the huge complexity involved in developing networks for entire enterprises, Teare (2004) takes a much more modular approach where the modules are components in an overall Enterprise Composite Network Model (ECNM).

The main functional areas within this ECNM are

- Particular enterprise campuses containing
  - Building access
  - Building distribution
  - Campus backbone
  - Server farm components
  - A network management component
- Enterprise Edge/Distribution containing
  - Internet connectivity
  - Ecommerce
  - VPN/remote access
  - Special WAN connectivity
- Service Provider Edge containing
  - ISP connections
  - Frame Relay/ATM/Gigabit connectivity
  - PSTN – Public Switched Telephone Network

Generally, the ECNM is primarily made up of Enterprise Campus Functional Area (ECFA) modules with the other modules/functions relating to interconnecting these. The ECFAs are where on site users gain access to the overall enterprise networks and where most of the user activity takes place. While this diagram ignores the network management component, it represents a typical diagram, Figure 1, of the hierarchy composing an enterprise campus. Any server farms are likeliest to be located directly off the distribution layer, though they might be in a demilitarized zone at the enterprise edge.

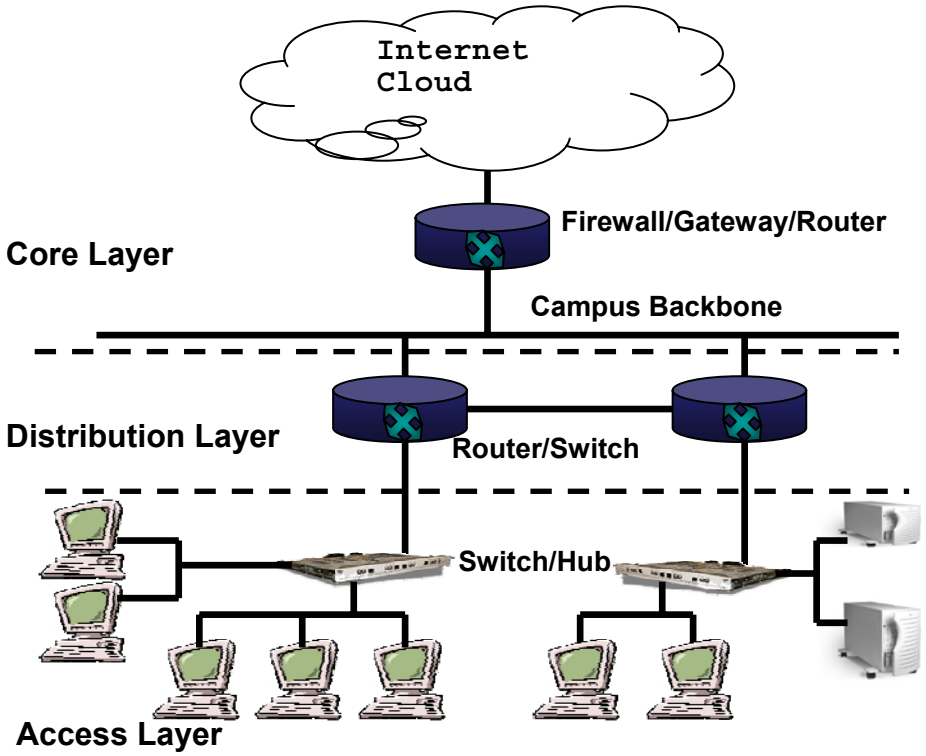

Figure 1. The Hierarchical Enterprise Campus Model

Very quickly, we summarize the basic functions of these layers.

- The **Access Layer** provides user access to the network.
- The **Distribution Layer** provides interconnectivity between the access segments and larger internetworks.
- The **Core Layer** provides the backbone and access to the internet service provider edge

While it is certainly possible to construct other models, this sort of design is probably the most widespread and well known. But rather than hinging all of our modeling underlying our DSS on this particular ECNM, we intend to use it to illustrate how widespread certain structures really are.

Due to the proliferation of Ethernet, of 10/100/1000Mbps, at the access and distribution layers, regardless of whether a particular implementation was developed using this ECNM, the underlying topology of access LAN segments are star wired/wireless. There is almost always some sort of central transceiver – transmits and receives – that acts as the focal point for network connectivity. It is almost always the case that these transceivers are hubs, wireless access points or layer 2 switches.

At the distribution layer you are very likely to find this same sort of star topology linking the access layer transceivers to transceivers that are further up the hierarchy. These distribution layer transceivers are more likely to be routers or layer 3 switches due to their internetworking capabilities via IP addresses. Though, it is becoming more and more common to provide redundant connectivity at the distribution layer. But, at their basis, most networks, due to the proliferation of Ethernet at what are essentially the access and distribution layers in all networks, are hierarchical star topologies. Though this topology almost never holds true once one gets to the core layers and out on the internet cloud. But these underlying topologies are what motivate using the following queueing network model to analyze different designs and the DSS that we have developed in order to facilitate the analysis.

The Model

One specialized model that is useful when analyzing network systems can be called the Open Central Transceiver Queueing Network (OCTQN). In a central transceiver models the internal traffic must flow in and out of a centralized device. The overall star/hub design of the model can help make it appropriate for modeling client connections to hubs, switches, routers, servers and/or other network devices.

The model is given in figure 2. The meaning of each of the symbols is described in the following list.

- There are M stations or work areas
  - Stations 1 through M correspond to client computer areas
  - The 0-th station corresponds to the transceiver
- Each station can have a different number of servers $s_0, s_1, s_2, \ldots, s_M$
  - The number of servers at a station will often be 1
  - The service durations for each station i are exponentially distributed with average rate μ_i.
- The client stations can receive arrivals from outside the system in a Poisson stream with rate λ_i.
- The probability that a service action on a client will result in a request from a client station i to the central server is p_i, where $i = 1, \ldots, M$.
- The probability that a service action on the client will result in another service action is r_i where $I = 1, \ldots, M$.
  - There is some probability that the source of the activity at station i will leave altogether after a service action is $(1 - p_i - r_i)$.
- The probability that a service action by the central server will result in something being sent to the client is q_i for $i = 1, 2, \ldots, M$.
- The probability that a service action by the central server will result in something being sent outside the network is q_0.

You can think of the items flowing in this network as client requests and transceiver responses. You might break it down further and think of this flow of requests as operating at a packet level.

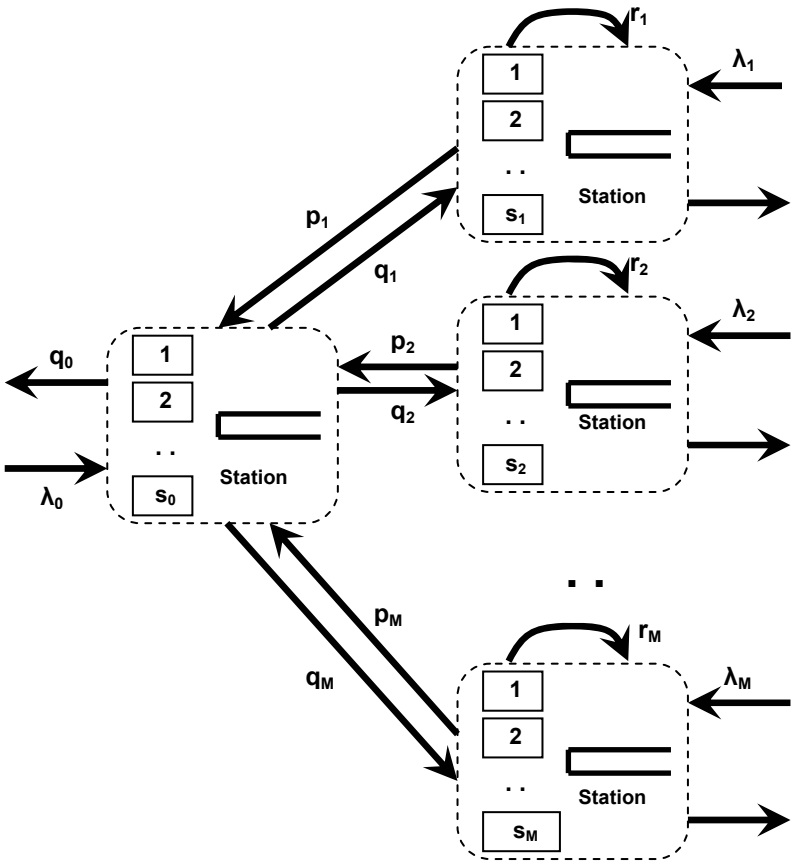

Figure 2. The OCTQN

The unusual aspect of this parameterization of the central transceiver model is that it allows for requests and/or packets to be submitted to the central transceiver with some probability p_i. The times between these submissions are exponentially distributed with the service rate μ_i. The probability a client generates another network request is r_i. Also, clients can leave altogether with probability $(1 - p_i - r_i)$. Or new clients can arrive to generate requests in this same sort of pattern in time intervals that are exponentially distributed with rate λ_i.

Most client stations will consist of a single client, but it is not unreasonable to have more than one client sharing a connection to the transceiver. This is why we need the number of channels/service providers, s_i, to be different at each station when appropriate. The transceiver could be a server farm, have multiple processors in a single component or have multiple components. Thus the transceiver will often have multiple channels/service providers.

The transceiver can either send the results of a request back to a client with probability q_i or send it on to someplace else on a larger network with probability q_M. The times between the requests to the transceiver from elsewhere on a larger network is exponentially distributed with rate λ_M.

The equations associated with this sort of network can be found in Kleinrock (1975) or Jackson (1957). The routing matrix for the routing equations will look like the following.

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ \vdots \\ M \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & \cdots & M \end{array} \\ \begin{bmatrix} 0 & q_1 & q_2 & q_3 & \cdots & q_M \\ p_1 & r_1 & 0 & 0 & \cdots & 0 \\ p_2 & 0 & r_2 & 0 & \cdots & 0 \\ p_3 & 0 & 0 & r_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ p_M & 0 & 0 & 0 & \cdots & r_M \end{bmatrix} \end{array}$$

Notice that all of the entries are zero except along the diagonal, the top row and the first column. This is because all activity moves through the central transceiver.

The system of network routing equations is written in matrix form using transposes in order to make use of column vectors. The vector of Γ_i's contains the unknown derived arrival rates that need to be solved for.

$$\begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \vdots \\ \lambda_M \end{bmatrix} = \begin{bmatrix} 1 & -p_1 & -p_2 & -p_3 & \cdots & -p_M \\ -q_1 & 1-r_1 & 0 & 0 & \cdots & 0 \\ -q_2 & 0 & 1-r_2 & 0 & \cdots & 0 \\ -q_3 & 0 & 0 & 1-r_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -q_M & 0 & 0 & 0 & \cdots & 1-r_M \end{bmatrix} \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \\ \Gamma_2 \\ \Gamma_3 \\ \vdots \\ \Gamma_M \end{bmatrix}$$

Written out in simultaneous equation form, while omitting zeros, they look like the following.

$$\begin{aligned} \lambda_0 &= \Gamma_0 - (p_1\Gamma_1 + p_2\Gamma_2 + p_3\Gamma_3 + \cdots + p_M\Gamma_M) \\ \lambda_1 &= -q_1\Gamma_0 + (1-r_1)\Gamma_1 \\ \lambda_2 &= -q_2\Gamma_0 + (1-r_2)\Gamma_2 \\ &\vdots \\ \lambda_M &= -q_M\Gamma_0 + (1-r_M)\Gamma_M \end{aligned}$$

Isolating the Γ_i's (for I = 1, 2, … , M-1) in terms of Γ_M gives the following.

$$\begin{aligned} \Gamma_0 &= \lambda_0 + (p_1\Gamma_1 + p_2\Gamma_2 + p_3\Gamma_3 + \cdots + p_M\Gamma_M) \\ \Gamma_1 &= \frac{\lambda_1}{(1-r_1)} + \frac{q_1\Gamma_0}{(1-r_1)} \\ \Gamma_2 &= \frac{\lambda_2}{(1-r_2)} + \frac{q_2\Gamma_0}{(1-r_2)} \\ &\vdots \\ \Gamma_M &= \frac{\lambda_M}{(1-r_M)} + \frac{q_M\Gamma_0}{(1-r_M)} \end{aligned}$$

Substituting each of the equations for Γ_i into the last equation allows us to solve to get the following for Γ_0.

$$\Gamma_0 = \frac{\lambda_0 + \frac{p_1\lambda_1}{(1-r_1)} + \frac{p_2\lambda_2}{(1-r_2)} + \cdots + \frac{p_M\lambda_M}{(1-r_M)}}{1-\left(\frac{p_1q_1}{(1-r_1)} + \frac{p_2q_2}{(1-r_2)} + \cdots + \frac{p_Mq_M}{(1-r_M)}\right)}$$

Then this can be back substituted in all of the previous expressions for the Γ_i's (for I = 1, 2, … , M) to get the following.

$$\Gamma_1 = \frac{\lambda_1}{(1-r_1)} + \left[\frac{q_1}{(1-r_1)}\left(\frac{\lambda_0 + \frac{p_1\lambda_1}{(1-r_1)} + \frac{p_2\lambda_2}{(1-r_2)} + \cdots + \frac{p_M\lambda_M}{(1-r_M)}}{1-\left(\frac{p_1q_1}{(1-r_1)} + \frac{p_2q_2}{(1-r_2)} + \cdots + \frac{p_Mq_M}{(1-r_M)}\right)}\right)\right]$$

$$\Gamma_2 = \frac{\lambda_2}{(1-r_2)} + \left[\frac{q_2}{(1-r_2)}\left(\frac{\lambda_0 + \frac{p_1\lambda_1}{(1-r_1)} + \frac{p_2\lambda_2}{(1-r_2)} + \cdots + \frac{p_M\lambda_M}{(1-r_M)}}{1-\left(\frac{p_1q_1}{(1-r_1)} + \frac{p_2q_2}{(1-r_2)} + \cdots + \frac{p_Mq_M}{(1-r_M)}\right)}\right)\right]$$

$$\vdots$$

$$\Gamma_M = \frac{\lambda_M}{(1-r_M)} + \left[\frac{q_M}{(1-r_M)}\left(\frac{\lambda_0 + \frac{p_1\lambda_1}{(1-r_1)} + \frac{p_2\lambda_2}{(1-r_2)} + \cdots + \frac{p_M\lambda_M}{(1-r_M)}}{1-\left(\frac{p_1q_1}{(1-r_1)} + \frac{p_2q_2}{(1-r_2)} + \cdots + \frac{p_Mq_M}{(1-r_M)}\right)}\right)\right]$$

Fortunately, in many instances it is reasonable to parameterize the model so that all of the $p_i = 1 \quad \forall i$ and $r_i = 0 \quad \forall i$. Substituting these values into the equations gives the following much simplified expressions.

Modeling and Acquiring Inputs

As with any model, one of the most difficult barriers to its implementation is acquiring appropriate input data. Other very important aspects to implementing the model are how it reflects the actual situation.

The OCTN model requires the following inputs for a station i:

1. Arrival rates to each station - λ_i where i = 0, … M.
2. Service rate for a server at each station - μ_i where i = 0, … M.
3. Number of servers at each station - s_i where i = 0, … M.
4. The probability the source of a request at a client station departs the station – d_i where i = 1, … M.
5. The probability the source of a request at a client station immediately returns to make another request - r_i where i = 1, … M.
6. The probability a request must go to the central server device – p_i where i = 1, … M.
7. The probability a request goes from the central server device to a client – q_i where i = 1, … M.
8. The probability a request goes from the central server device to outside the LAN – q_0.

The first three inputs in the list,

- arrival rate
- service rate
- number of servers

are likely to be relatively well known and/or measurable since they are typical inputs to queueing systems.

For our implementation we consider the following list of probabilities to be what should be measured or estimated.

- The probability a client departs the station after a particular request
- The proportion of requests from a client that need to go outside the network

If we consider these to be the d_i and p_i respectively we can compute the r_i for each station using

$$r_i = 1 - d_i - p_i$$

While it may be unrealistic in some instances to consider the proportion of requests that need to go outside the network to be the same as p_i, adjustments can be made for things like printer requests that stay within the network.

In order to estimate the overall proportion of requests that leave the network we make use of the following weighted average.

$$q_0 = \frac{\sum_{j=1}^{M} \frac{\lambda_j}{d_j} p_j}{\sum_{j=1}^{M} \frac{\lambda_j}{d_j}}$$

Then to get the q_i probabilities for the probability a request result is returned to each client station we need to make use of this estimate for q_0.

$$q_i = \frac{(1 - q_0)}{\sum_{j=1}^{M} \frac{\lambda_j}{d_j} p_j} \left(\frac{\lambda_i}{d_i} p_i \right)$$

The DSS

While it is beyond the scope of this particular paper to develop the algorithms or code for computing the output performance measures based on the inputs, we have a link where the code has been developed as a Java applet at http://esaighu.net/network_performance/OCSN.html. This will work with the more recent Sun Java plug-ins in more recent browsers. The user needs to input the number of client stations and common time units in order to get to the overall GUI.

After the routing equations are solved based on the inputs provided for the network the resulting solutions for the outputs can be obtained by treating each station as if it were an M/M/s queueing system with

- Arrival rate = Γ_i
- Service rate = μ_i
- Number of channels/service providers = s_i

Given particular situations and after appropriate data collection or estimation, the DSS can be used to evaluate alternatives.

Conclusion

This model can be used to study the impact of the devices and interconnections used on network segments of an overall internetwork. In order to account for the variability in demand, routes and packet or request sizes it is important to use probability distributions as the basis of the model. The star network topology represented by the OCTN is also very representative of access LAN segments and access layer transceivers being interconnected through the distribution layer. The topology can also be very relevant even higher up in the ECFA model.

Due to the structure of the model we were able to derive parameterized solutions to the equations for finding the steady state probabilities. This also impacts our ability to compute other performance measures. Having a parameterized solution allows for much more insight in applying the model. It also eases the computation of results within the DSS.

Making use of the DSS allows a designer to test out the consequences of design decisions about things such as the necessary characteristics of the transceivers and the nature that the connections to the clients need to have.

References

Deitel and Deitel (2002). *Java: How to Program*, 4th Edition, Prentice Hall.

Jackson (1957). Networks of Waiting Lines, *Operations Research*, 5, 518-521.

Kleinrock (1975). *Queueing Systems, Volume I: Theory*, Wiley Interscience.

Malik and Nair (2003). *Java Programming: From Problem Analysis to Program Design*, Thomson Course Technology.

Oppenheimer (2004). *Top-Down Network Design*, 2nd Edition, Cisco Press.

Sauer and Chandy (1981). *Computer Systems Performance Modeling*, Prentice Hall.

Teare (2004). *CCDA Self-Study: Designing for Cisco Internetwork Solutions*, Cisco Press.

Wheat, Hiser, Tucker, Neely, and McCullough (2002). *Designing a Wireless Network*, Syngress.