Performance Evaluation of Scheduling Strategies for Field Force Automation Traffic over GPRS

Niël C Malan

11949244

B. Eng (Electronic & Computer)

Thesis submitted in Partial Fulfilment of the Requirements
for the Degree
Magister Engineering (Electronic & Computer)
School of Electric and Electronic Engineering
at the
North-West University, Potchefstroom Campus,
South Africa

Supervisor: Prof. ASJ Helberg

2004

YUNIBESITI YA BOKONE-BOPHIRIMA
NORTH WEST UNIVERSITY
NOORDWES UNIVERSITEIT

Potchefstroomkampus

Executive Summary

This study makes use of guidelines from other studies together with a lack of appropriate network simulation tools to develop an example data source model successfully that can be used for Field Force Automation network simulations. By following the same procedure, companies can develop models tailored to their own systems, as the model is fully scalable for different sized workforces, as well as different Field Force Automation platforms and operational situations.

The result is that performance modelling can now be done to determine the capability of the support/backbone networks to handle the traffic generated by such systems, as well as to perform traffic optimisations for the fastest, most economical system.

An overview of the networks used to implement Field Force Automation systems is given. A number of queuing algorithms are explained in detail, after which they are used in a network simulation to show their individual performance when combined with the data source model.

Uittreksel

Hierdie studie neem sekere riglyne vanaf ander studies en kombineer dit met 'n gebrek aan toepaslike netwerk simulasie gereedskap om suksesvol 'n voorbeeld data bron model te ontwikkel wat in Veldmag Outomatisering netwerk simulasies gebruik kan word. Deur dieselfde prosedure te volg kan maatskappye hul eie modelle ontwikkel spesifiek vir hul eie sisteme, aangesien die model ten volle skaleerbaar is vir verskillende grotes maatskappye sowel as verskillende netwerk platforms en operasionele omstandighede.

Die resultaat hiervan is dat netwerk werksverrigting simulasies nou gedoen kan word op te bepaal of die ondersteunings netwerke die nodige kapasiteit het om die verkeer te kan hanteer. Netwerk optimerings kan ook gedoen word om die vinnigste en mees ekonomiese netwerk te verkry.

'n Oorsig van die basis netwerke wat gebruik word om Veldmag Outomatisering stelsels op te implementeer word gegee. Daarna word 'n aantal skedulerings algoritmes in detail bespreek. Hierdie algoritmes word dan in netwerk simulasies gebruik om die verskillende algoritmes se werksverrigting te wys wanneer die met die data bron model gekombineer word.

Table of Contents

List of Figures

List of Tables

Acronyms

API	Application Programming Interface
AuC	Authentication Centre
ATM	Asynchronous Transfer Model
BS	Base Station
BSC	Base Station Controller
BSS	Base Station Sub-system
BTS	Base Transceiver Station
BSSGP	BSS Gateway Protocol
CS	Circuit Switched
CS-x	Coding Scheme x (x replaced by a number from 1 to 4)
CLNP	Connectionless Network Protocol
CQ	Custom Queuing
DWRR	Deficit Weighted Round Robin
EIR	Equipment Identity Register
ETSI	European Telecommunications Standards Institute
FCFS	First-Come First-Served (same as FIFO)
FFA	Field Force Automation
FIFO	First-In First-Out
FQ	Fair Queuing
FSM	Finite State Model
FTP	File Transfer Protocol
GGSN	Gateway GPRS Support Node
GMSC	Gateway Mobile Switching Centre
GPRS	General Packet Radio Service
GPS	Generalized Processor Sharing
GSM	Global Systems for Mobile Communications
GSN	GPRS Support Node
GTP	GPRS Tunnelling Protocol
GSM RF	GSM Radio Frequency
HLR	Home Location Register

HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICMP	Internet Control Message Protocol
ISDN	Integrated Services Digital Network
IMSI	International Mobile Subscriber Identity
ISO	International Organisation for Standardisation
IP	Internet Protocol
KB	Kibobyte = 1024 bytes
Kbps	Kilobits per Second
LAN	Local Area Network
LLC	Logical Link Control
LLHP	Low-Latency, High-Priority
MTU	Maximum Transmission Unit
MAC	Medium Access Control
MDRR	Modified Deficit Round Robin
MS	Mobile Station
MSC	Mobile Switching Centre
MM	Mobility Management
NNOC	National Network Operations Centre
NSS	Network Switching Sub-system
OSI	Open Systems Interconnection
PCU	Packet Control Unit
PDA	Personal Digital Assistant
PDCH	Packet Data Channel
PDN	Packet Data Network
PDP	Packet Data Protocol
PDU	Packet Data Unit
PQ	Priority Queuing
PS	Packet Switched
PTM	Point-to-Multipoint
PTP	Point-to-Point
PTM-G	PTM Group
PTM-M	PTM Multicast

PTP-CLNS	PTP Connectionless Network Service
PTP-CONS	PTP Connection-oriented Network Service
PLMN	Public Land Mobile Network
QoS	Quality of Service
Um	Radio Interface
RED	Random Early Detection
RLC	Radio Link Control
RTTI	Road Traffic and Transport Informatics
SDU	Service Data Unit
SGSN	Serving GPRS Support Node
SMS	Short Message Service
SNDCP	Sub Network Dependent Convergence Protocol
TDMA	Time Division Multiple Access
TCP	Transmission Control Protocol
TLLI	Temporary Link Layer Interface
UDP	User Datagram Protocol
VLR	Visitor Location Register
VoIP	Voice over IP
WFQ	Weigted Fair Queuing
WML	Wireless Markup Language
WRR	Weighted Round Robin
WWW	World Wide Web
XML	eXtendable Markup Language

1 Chapter 1 – Introduction

This chapter will give an introduction to the project as well as give the reader some background knowledge of how the project came about.

There are a number of companies in South Africa that have large mobile workforces spread over large areas, even covering the entire South Africa. Managing these large workforces can be a daunting task with huge logistic and communication problems coming to the surface. This is exactly where the need for Field Force Automation systems comes into play. Telecommunication companies that have large networks to maintain are good examples for such systems.

1.1 PROBLEM STATEMENT

The success of Field Force Automation system implementations is mainly based on the speed and efficiency with which network problems and job information can be processed and routed to the correct personnel. The latest Field Force Automation system implementations make use of cellular networks and the *General Packet Radio Service* (GPRS) for information distribution. Most of these implementations assume that the cellular networks they use will always offer high-speed data transfer with little or no congestion problems [1][2]. This assumption has, however, not been tested. This can be risky given the fact that one of the main reasons for using such a system is to provide fast access to the Field Force Management database with short delays in downloading new information.

Implementing such a Field Force Automation system requires planning, part of which is the simulation of the proposed new system to ensure that traffic flow problems do not occur. These simulations require, among a number of other tools, a data source. This model is needed to generate the simulated traffic in a way that closely resembles the real-world traffic.

An initial survey on the feasibility of a study concerning network utilisation of Field Force Automation applications revealed a lack of simulation tools. The first and foremost of which is the lack of a suitable data source model to use in

the simulations [3]-[7]. The development of such a data source model will, therefore, be undertaken as the first part of this project.

Field Force Automation systems that have already been implemented sometimes grow beyond their original planned capacity. Performing the required network capacity upgrades can be a costly undertaking. There are, however, possible remedies to help extend the life of a current system, without a costly upgrade. The second part of the project will be to use the data source model in a network simulation, to study the performance of different packet scheduling strategies and the possible increase in network capacity they might introduce.

1.2 EXISTING WEBFORCE DISPATCH SYSTEM

Telkom is currently using WebForce, a web-based workforce management database containing all the fault information on the network. One problem with the use of this resides in the way the database is accessed remotely. The equipment required to access the database is expensive and requires a dial-up connection (i.e. a phone call) each time the user wants to exchange data with the database [1].

Currently a handheld device, called a Huskey, is used to exchange data with the database (See *Figure 1-1*). This happens either through a dial-up landline connection or through a cellular phone, using a data cable or infrared. The problem lies in the fact that dial-up connections have to be established each time the database needs to be accessed. The dial-up connections are slow and expensive.

With the current WebForce access technology all faults reported to Telkom countrywide are routed to the NNOC (National Network Operations Centre) in Centurion, Pretoria. At the NNOC, these faults are logged into the WebForce database [1].

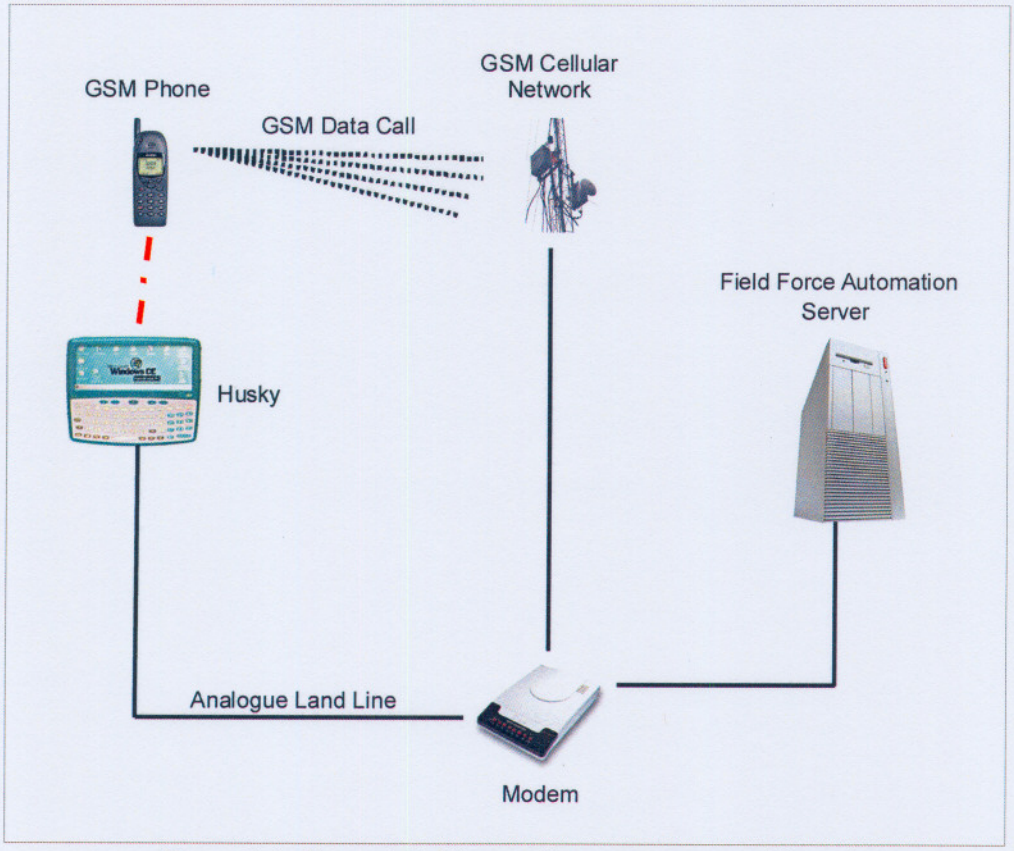

Figure 1-1: Current field operations management layout

Each fault is then assigned to a specific technician according to his location, skills and the equipment he has been issued with. Telkom has approximately 13000 telephone technicians whose daily duties are installation and maintenance of telecommunications network hardware. After the technician has been notified that there is a task waiting for him, he has to make a dial-up connection either from a landline or through a cellular phone, using a Husky (field computer).

After a connection has been established, the technician has to log into the WebForce website, where he receives the tasks that were assigned to him and attends to the faults assigned to him. After attending to the faults, he has to make a dial-up connection again to log the fault report to WebForce. This whole dial-up and login process takes approximately 10 minutes and each technician has an average of 4 assignments per day [1].

The time spent waiting for data to be sent or received is time wasted in man-hours. This cost can roughly be calculated as:

10/60 (10 minutes per session used, converted to hours)
x 4 (Sessions per day)
x R133-00 (Technician hourly rate)
x 13000 (Total number of technicians)
= R 1,15 mil (Daily cost of system access)

The data-calls also cost money. This can be calculated as:

10 (Minutes per session)
x 4 (Sessions per day)
x R1.6 (Call cost per minute)
x 13000 (Total number of technicians)
= R 0.8 mil (Daily data-call cost)

By improving the dial-in access by using modern technology as part of the workforce management solution, significant savings can be made to these daily expenses. The following approach is proposed.

1.3 PROPOSED WEBFORCE DISPATCH UPGRADE

The existing WebForce will stay in place. In the new proposed system (as shown in *Figure 1-2*) remote access of the data will be done via GPRS, which is expected to be faster than mobile data calls. This would be done with a field unit that has GPRS capability. The field unit will communicate through the GPRS network and gateway with the GPRS/Server Interface, where the GPRS/Server Interface would connect to the WebForce Server at the NNOC, which will handle the distribution of tasks [1].

After a GPRS connection has been established, the technician has to log into the WebForce database to receive the tasks that were assigned to him. After

attending to the tasks assigned to him, he has to log the fault report to WebForce.

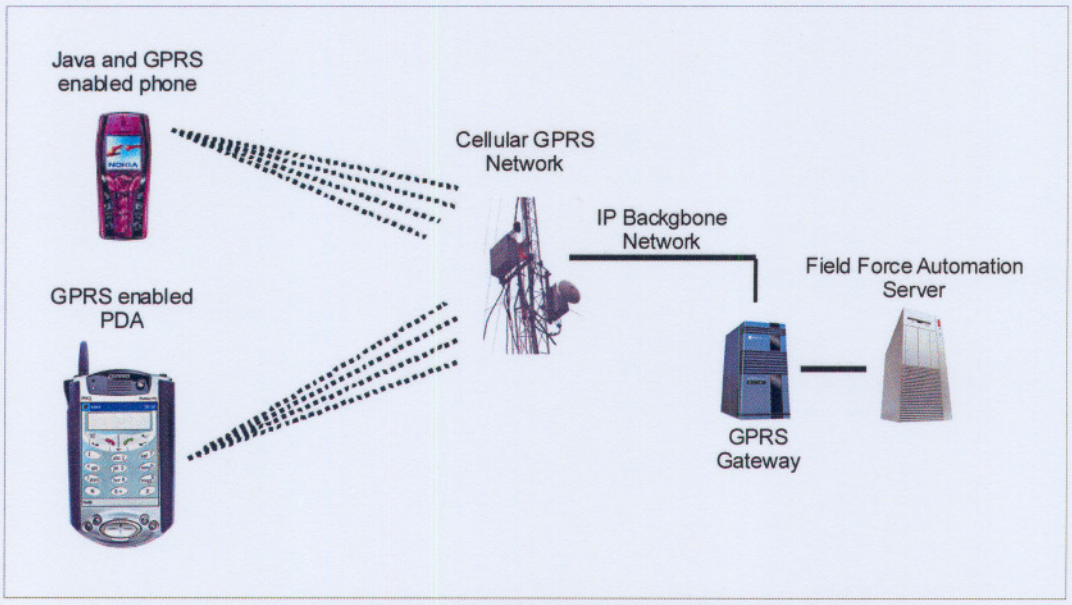

Figure 1-2: New WebForce Access Technology

With this method the tasks would be distributed more efficiently to the technicians in the field. This will enable the field technician to connect to the WebForce Server at any time and any place where there is GPRS coverage. GPRS is used because it is faster than a normal data call and the billing is done according to the data sent and received and not the duration of the connection.

The amount of data that GPRS can deliver for each Technician/Day for the same cost on GSM/Day:

R 832 000 (Data-call cost per day)
÷ R 2 (Cost per megabyte)
÷ 13000 (Total number of technicians)
= 32 MB (Daily traffic allowed per technician)

This shows that if a technician uses less then 32MB of data per day it would be profitable to use GPRS instead of data calls.

1.4 OBJECTIVES AND METHODOLOGY

The following objectives have been identified:

1. Develop a Field Force Automation data source model for use in network simulations.
2. Use the data source model to simulate the effect of different packet scheduling algorithms.
3. Make recommendations on network capacity planning and optimisation.

These objectives will be reached by following standard research procedures, coupled with experimentation for the data source model and simulation for the network congestion.

1.5 CONCLUSION

It is clear that the development of a simulation tool, such as the data source model, can assist with the planning and simulation of Field Force Automation systems. The network infrastructure responsible for the communication between the Field Force Automation units also needs to be understood and will be discussed next.

2 Chapter 2 – Cellular Network Infrastructure

In order to understand the Field Force Automation system, the network on which it is based also needs to be understood. This chapter will introduce the reader to the base network, Global Systems for Mobile Communications (GSM), as well as its upgrade, General Packet Radio Service (GPRS).

2.1 GSM BACKGROUND

GSM was designed by the European Telecommunications Standards Institute (ETSI) [1]. GSM is currently the most widely used mobile system in the world, used by 835 million GSM subscribers, 400 operators, across 195 countries, which is more than one in ten of the world's population [9][10]. In 1994, the first GSM services were introduced to the South African public [11].

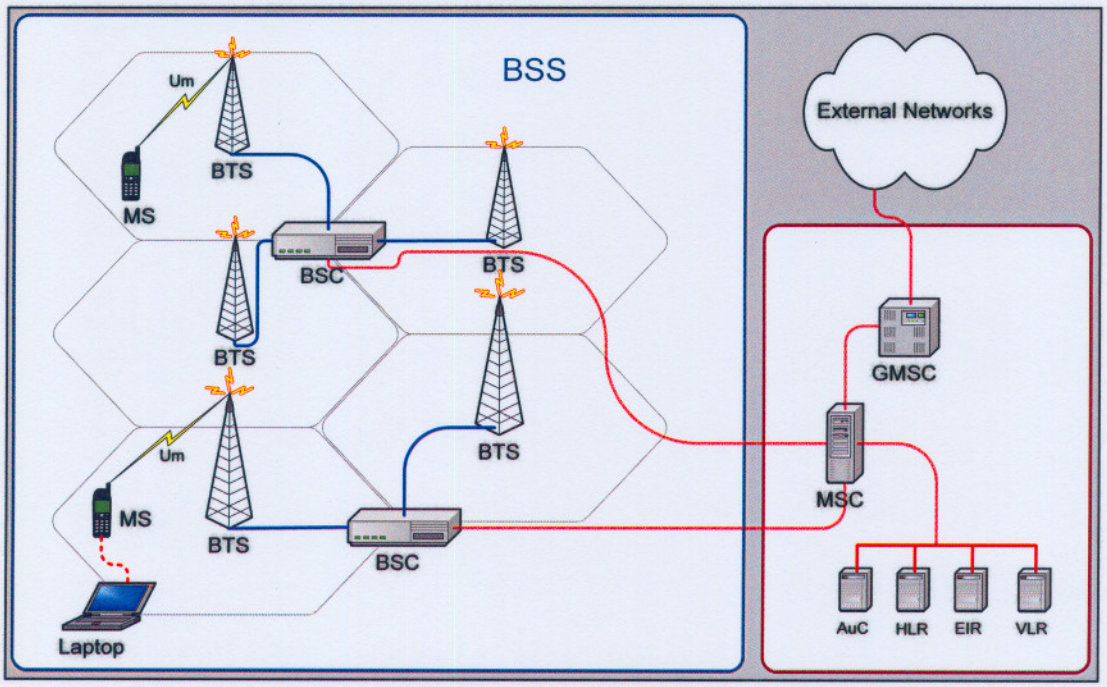

Figure 2-1: Basic GSM Network Architecture [12][13]

2.2 GSM NETWORK COMPONENTS [12][13]

The basic structure of a GSM network is shown in *Figure 2-1*. For users to access the GSM network, they require Mobile Stations (MS), most commonly a cellular phone. These Mobile Stations connect to the GSM network using a Radio Interface (Um). The GSM network consists of a Base Station Sub-system (BSS) and a Network Switching Sub-system (NSS). All the individual components shown in *Figure 2-1* have the following individual functions:

2.2.1 MOBILE STATION (MS)

The MS is a GSM device like a cellular phone, GSM enabled PDA or a notebook with a GSM card. The MS communicates with the Base Station (BS) using the Radio Interface (Um).

2.2.2 HOME LOCATION REGISTER (HLR)

The HLR is a database used to store and manage permanent data of subscribers such as service profiles, location information and activity status.

2.2.3 MOBILE SWITCHING CENTRE (MSC)

The MSC is responsible for telephony switching functions of the network. It also performs authentication using the Authentication Centre (AuC) to verify the user's identity and to ensure the confidentiality of the calls.

2.2.4 AUTHENTICAION CENTRE (AUC)

The AuC provides the necessary parameters to the MSC to perform the authentication procedure. The AuC is shown as a separate logical entity but is generally integrated with the HLR.

2.2.5 EQUIPMENT IDENTITY REGISTER (EIR)

The EIR is a database that contains information about the identity of the mobile equipment. It prevents calls from unauthorized or stolen Mobile Stations.

2.2.6 VISITOR LOCATION REGISTER (VLR)

The VLR is a database used to store temporary information about the subscribers and is needed by the Mobile Switching Centre (MSC) in order to service visiting subscribers. The MSC and VLR are commonly integrated into one single physical node and the term MSC/VLR is used instead. When a subscriber enters a new MSC area, a copy of all the necessary information is downloaded from the HLR into the VLR. The VLR keeps this information so that calls of the subscriber can be processed without having to interrogate the HLR each time. The temporary information is cleared when the mobile station roams out of the service area.

2.2.7 GATEWAY MOBILE SWITCHING CENTRE (GMSC)

A GMSC is an MSC that serves as a gateway node to external networks, such as ISDN or wire-line networks.

2.2.8 NETWORK SWITCHING SUB-SYSTEM (NSS)

The NSS is responsible for call control, service control and subscriber mobility management functions. The NSS consists of the Mobile Switching Centre (MSC), Gateway Mobile Switching Centre (GMSC), Equipment Identity Register (EIR), Authentication Centre (AuC), Home Location Register (HLR) and Visitor Location Register (VLR).

2.2.9 BASE TRANSCEIVER STATION (BTS)

The BTS handles the radio interface to the MS. It consists of radio equipment (transceivers and antennas) required to service each cell in the network.

2.2.10 BASE STATION CONTROLLER (BSC)

The BSC provides the control functions and physical links between the MSC and the BTS. A number of BSCs are served by one MSC, while several BTSs can be controlled by one BSC.

2.2.11 BASE STATION SUB-SYSTEM (BSS)

The BSS is basically the collection of the Base Transceiver Stations (BTS) and their Base Station Controllers (BSC). The BSS is responsible for radio communications between the Mobile Stations and the Network Switching Sub-system (NSS).

2.3 SHORTCOMINGS OF GSM

In conventional GSM, the connection setup takes several seconds and rates for data transmission are restricted to 14.4Kbps. In circuit switched services, billing is based on the duration of the connection *[12][13].* This is a drawback when considering the normal use of a data connection. There is almost always idle time, when the user is reading information, or typing a response. These idle times incur unnecessary costs.

This duration-based billing structure is unsuitable for applications with bursty traffic profiles such as Field Force Automation systems. The user must pay for the entire airtime, even for idle periods when no information is sent (i.e. when

the user reads the information on a new job). In contrast to this, with packet switched services, billing can be based on the amount of transmitted data.

2.4 GPRS BACKGROUND

Now that the GSM network layout has been discussed, the architecture can be expanded to take into account the changes required to offer GPRS.

GPRS is a new service offered on most GSM networks. GPRS was first commercially offered by Vodacom in October 2002 [11]. The service provides high speed data services to mobile units and billing is done on the amount of data transmitted and not the duration of a session. This billing system results in the cost of the service being less than that of a normal data call when operating with bursty traffic.

The advantage for the user is that he or she can be "online" over a long period of time but only be billed on the data exchanged in the session. GPRS improves the utilisation of the radio resources, offering volume based billing, higher transfer rates, shorter access times and simplifying the access to packet data networks. For most operators GPRS is the easiest and most logical way of offering customers fast data services [13].

GPRS is much better suited for burst-transfer applications such as Web browsing, e-mail and database queries than the normal GSM data-call. These improvements are realised by changing from a circuit-switched service to a packet-switched service, which offers the following advantages [12][13]:

1. Allows reduced connection set-up times and high transfer speeds by allowing a user to access more network resources during peak transfers.
2. Provides efficient usage of radio link resources by assigning resources only when they are required and then returning to an idle mode.

3. Supports existing packet-oriented protocols such as X.25 and IP within the network.
4. Charges customers on the amount of data transferred and not on time spent online.

GPRS is a suitable communications service for applications that need to transfer small to medium amounts of data frequently. A packet switched network service allows more users per network, causing the service to be cheaper than circuit switched alternatives. GPRS is offered as a value-added service and network operators hope to generate new business by offering the service.

Now that the background of GPRS has been discussed, a more detailed look into the architecture of the network and its differences from normal GSM will be taken.

2.5 GPRS ARCHITECTURE

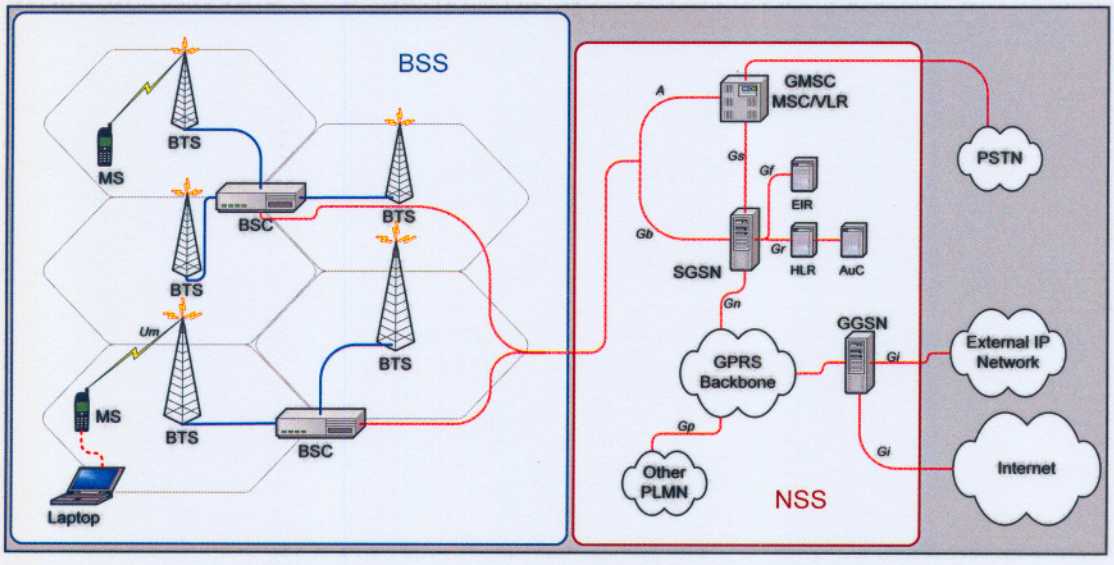

Figure 2-2: Basic GPRS Network Architecture [12][13]

When looking at the BSS part of a GPRS network, there is basically no change with only a Packet Control Unit (PCU) being added to the BSS. The biggest difference can be seen in the NSS part of the network. To be able to offer the new service, a few network elements have to be added or changed. These new elements are called GPRS Support Nodes (GSN). GSNs are responsible for the delivery and routing of data packets between the mobile stations and the external Packet Data Networks (PDN). There are two types of GSN, firstly the Gateway GPRS Support Node (GGSN) and secondly, the Servicing GPRS Support Node (SGSN). The functions of these nodes will be discussed in the rest of the chapter. *Figure 2-2* illustrates the new system architecture.

2.5.1 SERVING GPRS SUPPORT NODE (SGSN)

A Serving GPRS Support Node (SGSN) is responsible for the delivery of data packets to and from the mobile stations within its service area. Different SGSNs service different service areas. Its tasks include packet routing and transfer, mobility management (attach/detach and location management), logical link management and authentication and charging functions [12][13].

2.5.2 GATEWAY GPRS SUPPORT NODE (GGSN)

A Gateway GPRS Support Node (GGSN) acts as the interface between the GPRS network and the external Packet Data Network (PDN). It converts the GPRS packets coming from the SGSN into the appropriate packet data protocol (PDP) format (e.g., IP or X.25) and sends them out on the corresponding packet data network. The GGSN is also responsible for routing incoming packets (from external PDN) to the correct SGSN. For this purpose, the GGSN stores the current SGSN address of the user and his or her profile in its location register. The GGSN also performs authentication and charging functions [12][13].

A GGSN is the interface to external packet data networks for several SGSNs but an SGSN may route its packets over different GGSNs to reach different packet data networks. *Figure 2-2* also shows the interfaces between these new GSN's and the GSM network. All GSNs are interconnected via an IP-based backbone network. Data is exchanged within this backbone by encapsulating the PDN packets and transmitting them, using the GPRS Tunnelling Protocol [12][13].

2.5.3 MOBILITY MANAGEMENT (MM)

A MS connects to the GPRS network by requesting a GPRS attach procedure which establishes a logical link between the MS and a SGSN. This link is identified by a Temporary Logical Link Identifier (TLLI) and changes when the MS moves to another area and is served by a new SGSN.

The MS states are depicted in *Figure 2-3*. In the *Idle* state the MS is not in a direct connection with the GPRS network and can, therefore, only receive broadcast messages intended for all MS's being covered by the same SGSN. The MS needs to perform the GPRS attach procedure in order to connect to the GPRS network. This will change its status from *Idle* to *Standby* and make the MS reachable.

When an MS is connected to the network but not actually exchanging information it is put in the *Standby* state. When the MS wants to transmit data or data arrives at the SGSN, destined for the MS, these intentions are communicated between the SGSN and the MS which cause the MS to enter the *Ready* state.

Data is exchanged when the MS is in the Ready state. The MS disconnects from the network by requesting a *Detach* procedure, which changes the MS back to the *Idle* state. A timer is also activated when the MS changes status

to Ready. When the timer expires (when data is no longer being exchanged), the MS is changed back to the *Standby* state [12].

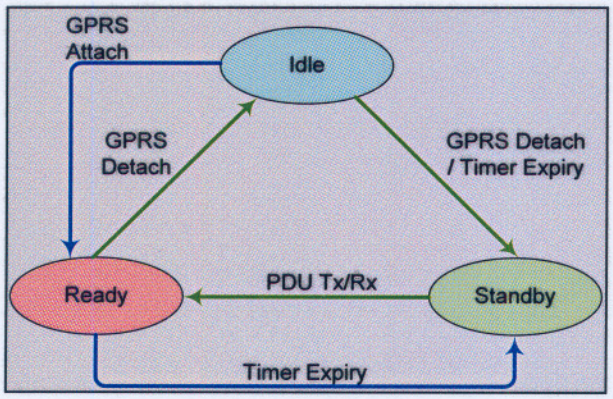

Figure 2-3: GPRS State Model [12]

2.5.4 CLASSES OF GPRS MOBILE STATIONS [12]

GPRS terminals (GPRS MS) are divided into three classes according to their functionality:

Class A is the most demanding class of GPRS terminals. A terminal of this class is able to establish simultaneous connections both with circuit switched (CS) and packet switched (PS) sides of the network.

Class B is able to select automatically either circuit switched or packet switched connection but only one can be active at a time.

Class C terminals cannot be attached to both services at the same time and the selection of the operation mode must be done manually.

2.6 GPRS PROTOCOLS [12][13]

Figure 2-4 shows the GPRS protocol stacks used in data transfer between a server and a mobile client. The GPRS protocols are situated in the lower levels of the International Organisation for Standardisation/Open Systems Interconnection (ISO/OSI) reference model. Above the network layer (OSI layer 3), widespread standardised protocols can be used, for example TCP/IP and X.25.

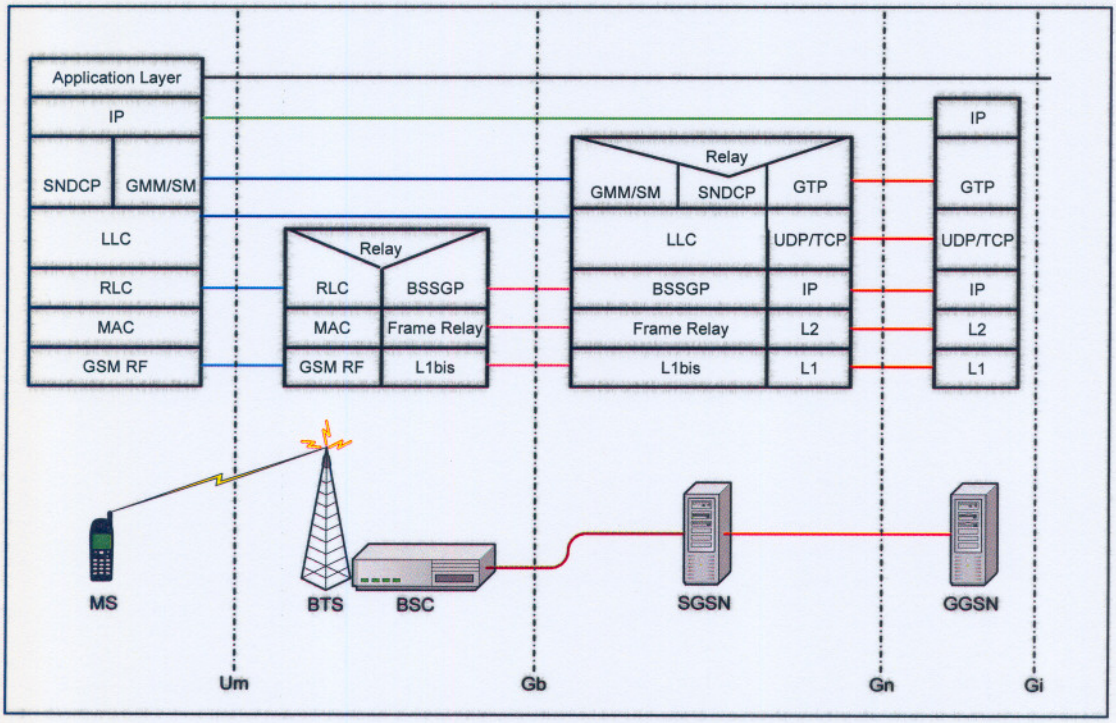

Figure 2-4: GPRS Protocols

It is not of importance for this project to discuss all the protocols shown. It is, however, important to note that an IP based connection exists between the MS and GGSN.

2.6.1 GPRS CODING SCHEMES [13]

Channel coding is a technique used to protect the transmitted data packets from errors. Four channel coding schemes are defined on GPRS standards for packet data traffic channels. These coding schemes are marked as CS-1 to CS-4.

CS-1 has highest error correction and lowest data throughput. The more efficient channel coding used, the smaller the proportion of the payload in the emission. Therefore, higher data rates are achieved by reducing or removing the error correction bits. *Table 2-1* shows the theoretical maximum speeds that can be achieved for the different coding scheme and time slot combinations.

Timeslots	CS-1	CS-2	CS-3	CS-4
1	9,05 kbps	13,40 kbps	15,60 kbps	21,40 kbps
2	18,10 kbps	26,80 kbps	31,20 kbps	42,80 kbps
3	27,15 kbps	40,20 kbps	46,80 kbps	64,20 kbps
4	36,20 kbps	53,60 kbps	62,40 kbps	85,60 kbps
5	45,25 kbps	67,00 kbps	78,00 kbps	107,00 kbps
6	54,30 kbps	80,40 kbps	93,60 kbps	128,40 kbps
7	63,35 kbps	93,80 kbps	109,20 kbps	149,80 kbps
8	72,40 kbps	107,20 kbps	124,80 kbps	171,20 kbps

Table 2-1: GPRS Coding Scheme Speeds

These coding schemes are automatically selected and selection is based on signal quality. When the signal is strong, with little interference, CS-4 is used and the coding scheme is reduced as the MS moves away from the BTS. The number of timeslots is influenced by the number of users using the same BTS, as well as the number of simultaneous slots supported by the MS.

2.7 CONCLUSION

Now that the basics of GSM and GPRS have been discussed, the choices made in the rest of the project will be easier to understand. This network architecture discussion provided background on the infrastructure used by field force automation systems. The next chapter will give a brief description on Quality of Service and will also explain the different packet scheduling or queuing algorithms in detail.

3 Chapter 3 –Quality of Service (QoS) and Queuing

This chapter will explain the concept of Quality of Service (QoS), as well as give an in depth explanation of some of the QoS tools, known as Queuing Algorithms.

Quality of Service (QoS) refers to the capability of a network to differentiate between different classes of network traffic and to provide better service to selected classes of traffic. QoS can be applied over various technologies, including Frame Relay, Asynchronous Transfer Mode (ATM), Ethernet and 802.11 networks, etc. The networks may use any or all of these underlying technologies.

The primary goal of QoS is to provide control over:

- Traffic priority
- Bandwidth
- Jitter
- Delays (required by some real-time and interactive traffic)
- Improved loss characteristics.

Also important to ensure that providing priority for one class of traffic does not cause failure in another. QoS is an important part of the network management toolkit and provides the elemental building blocks that will be used for future business applications. [14]

QoS technology enables complex networks to control and predictably service a variety of networked applications and traffic types. Almost any network can take advantage of QoS for improved efficiency, whether it is a small corporate network, an Internet service provider or an enterprise network.

3.1 QOS CONCEPTS [14]

Fundamentally, QoS enables one to provide better service to certain flows. This is done by either raising the priority of a flow or limiting the priority of another flow. When using congestion-management tools, one tries to raise the priority of a flow by queuing and servicing queues in different ways. The queue management tool used for congestion avoidance raises priority by dropping lower-priority flows before higher-priority flows. Policing and shaping

provide priority to a flow by limiting the throughput of other flows. Link efficiency tools limit large flows to show a preference for small flows.

QoS tools can assist in alleviating most congestion problems. However, many times there is just too much traffic for the bandwidth supplied. In such cases, QoS is merely a bandage. A simple analogy comes from pouring syrup into a bottle. Syrup can be poured from one container into another container at or below the size of the spout. If the amount poured is greater than the size of the spout, syrup is wasted. However, one can use a funnel to catch syrup pouring at a rate greater than the size of the spout. This allows one to pour more than what the spout can take, while still not wasting the syrup. However, consistent overpouring will eventually fill and overflow the funnel.

3.2 QUALITY OF SERVICE (QOS) PARAMETERS

ETSI GPRS recommendations define quality of service for users according to specific parameters [13]:

- Reliability
- Delay
- Peak throughput
- Mean throughput.

In this study the focus will be mostly on delays and ways to minimise it. The reasons for this are as follows:

- Reliability is mostly determined by the cellular network and, therefore, is controlled by the cellular network operator.
- Peak and mean throughput are once again determined by the cellular network infrastructure and hardware used. The Field Force Automation system owner or implementer does not have control over these parameters.

- Jitter (variation in delay times) is a parameter that can be investigated, but jitter only really influences real-time applications like video or Voice-over-IP. Field Force Automation systems are not affected by difference in delay times.
- The average packet delay does influence the performance of a Field Force Automation system and is influenced by both the cellular network and the Field Force Automation server. The Field Force Automation server is under the control of the system owner or implementer and can, therefore, be modified to optimise the average packet delay.

The way to influence packet delay without purchasing more bandwidth or faster hardware is by using the QoS tool known as packet scheduling or queuing. The different packet scheduling algorithms will be discussed next, after which they will be applied in a simulation environment.

3.3 PACKET SCHEDULING ALGORITHMS

Because of the bursty nature of voice/video/data traffic, sometimes the amount of traffic exceeds the speed of a link. At this point the packets start queuing, waiting to be transmitted over the congested link. Congestion-management tools address this situation. Tools include priority queuing (PQ), custom queuing (CQ) and weighted fair queuing (WFQ).

Because queues are not of infinite size, they can fill and overflow. When a queue is full, any additional packets cannot get into the queue and will be dropped. This is a tail drop. The issue with tail drops is that the router cannot prevent this packet from being dropped (even if it is a high-priority packet). So, a mechanism is necessary to do two things:

1. Try to ensure that the queue does not fill up so that there is room for high-priority packets.
2. Allow some sort of criteria for dropping packets that are of lower priority before dropping higher-priority packets.

Weighted early random detect (WRED) provides both of these mechanisms. A thorough explanation of the first six queuing algorithms is given in [15]. The following segments offer summaries of the explanantions in [15], after which Modified Deficit Round Robin (MDRR) and Random Early Detection (RED) are added.

3.3.1 FIFO (FIRST IN FIRST OUT) [15]

First-in, first-out (FIFO) queuing is the most obvious and basic queue scheduling discipline. In FIFO queuing, all packets are treated equally by placing them into a single queue and then servicing them in the same order that they arrived at the queue. This concept is illustrated in *Figure 3-1*. FIFO queuing is also referred to as First-come, first-served (FCFS) queuing.

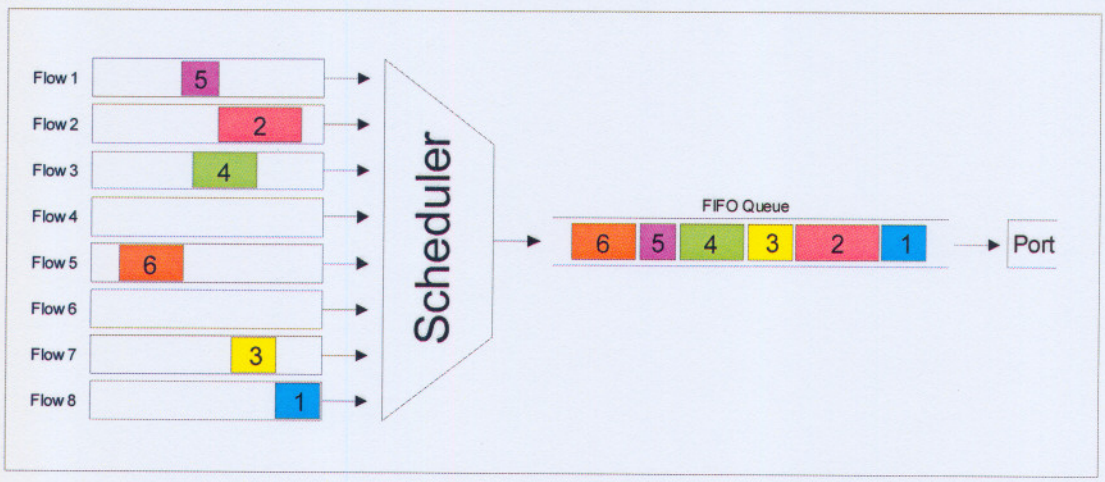

Figure 3-1: FIFO queuing example

3.3.1.1 FIFO Benefits and Limitations

FIFO queuing offers the following benefits:

- For software-based routers, FIFO queuing requires almost no computational power from the system.

- The behaviour of a FIFO queue is very predictable as packets are released in the same order they arrive and the maximum delay can be determined by the maximum depth of the queue.
- As long as the queue depth remains short, FIFO queuing provides simple contention resolution for network resources without adding significantly to the queuing delay experienced at each hop.

FIFO queuing also has the following limitations:

- A single FIFO queue does not allow routers to organize buffered packets and then service one class of traffic differently from other classes of traffic. i.e. Traffic Class Differentiation is not possible.
- A single FIFO queue impacts all flows equally, because the mean queuing delay for all flows increases as congestion increases. As a result, FIFO queuing can result in increased delay, jitter and packet loss in real-time applications traversing a FIFO queue.
- During periods of congestion, FIFO queuing benefits User Datagram Protocol (UDP) flows over TCP flows. When experiencing packet loss due to congestion, TCP-based applications reduce their transmission rate, but UDP-based applications remain oblivious to packet loss and continue transmitting packets at their usual rate. Because TCP-based applications slow their transmission rate to adapt to changing network conditions, FIFO queuing can result in increased delay, jitter and a reduction in the amount of output bandwidth consumed by TCP applications traversing the queue.
- A bursty flow can consume the entire buffer space of a FIFO queue, and that causes all other flows to be denied service until after the burst is serviced. This can result in increased packet delay, jitter and packet loss for the other well-behaved TCP and UDP flows traversing the queue.

3.3.1.2 FIFO Implementations and Applications

Generally, FIFO queuing is supported on an output port when no other queue scheduling discipline is configured. In some cases, router vendors implement two queues on an output port when no other queue scheduling discipline is configured: a high-priority queue that is dedicated to scheduling network control traffic and a FIFO queue that schedules all other types of traffic.

3.3.2 PQ (PRIORITY QUEUING) [15]

Priority queuing (PQ) is the basis for a class of queue scheduling algorithms that are designed to provide a relatively simple method of supporting differentiated service classes. In classic PQ, packets are first classified by the system and then placed into different priority queues. Packets are then placed in the output queue by first queuing all the highest priority packets and continuing to lower priorities when the higher ones are empty. Packets of the same priority are scheduled in FIFO order in their own queue. *Figure 3-2* gives a visual example of this algorithm.

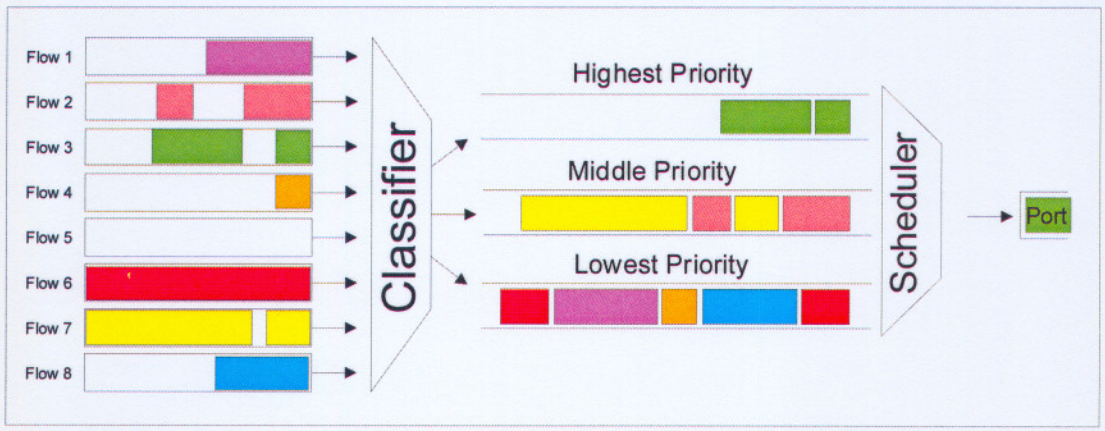

Figure 3-2: Priority Queuing example

3.3.2.1 PQ Benefits and Limitations

PQ offers a couple of benefits:

- For software-based routers, PQ places a relatively low computational load on the system when compared with more elaborate queuing disciplines.
- PQ allows routers to organize buffered packets and then service one class of traffic differently from other classes of traffic. For example, one can set priorities so that real-time applications like interactive video and voice have priority over applications that do not operate in real time.

PQ also has several limitations:

- If the amount of high-priority traffic is not policed or conditioned at the edges of the network, lower-priority traffic may experience excessive delay as it waits for unbounded higher-priority traffic to be serviced.
- If the volume of higher-priority traffic becomes excessive, lower-priority traffic can be dropped as the buffer space allocated to low-priority queues starts to overflow. If this occurs, it is possible that the combination of packet dropping, increased latency and packet retransmission by host systems can ultimately lead to complete resource starvation for lower-priority traffic.
- A misbehaving high-priority flow can add significantly to the amount of delay and jitter experienced by other high-priority flows sharing the same queue.
- PQ is not a solution to overcome the limitation of FIFO queuing where UDP flows are favoured over TCP flows during periods of congestion. If one attempts to use PQ to place TCP flows into a higher-priority queue than UDP flows, TCP window management and flow control mechanisms will attempt to consume all of the available bandwidth on the output port, therefore starving the lower-priority UDP flows.

3.3.2.2 PQ Implementations and Applications

Typically, router vendors allow PQ to be configured to operate in one of two modes:

- Strict priority queuing

- Rate-controlled priority queuing.

Strict PQ ensures that packets in a high-priority queue are always scheduled before packets in lower-priority queues. Of course, the challenge with this approach is that an excessive amount of high-priority traffic can cause bandwidth starvation for lower priority service classes. However, some carriers may actually want their networks to support this type of behaviour. For example, assume a regulatory agency requires that, in order to carry Voice over Internet Protocol (VoIP) traffic, a service provider must agree (under penalty of a heavy fine) not to drop VoIP traffic in order to guarantee a uniform quality of service, no matter how much congestion the network might experience. The congestion could result from imprecise admission control leading to an excessive amount of VoIP traffic or, possibly, a network failure. This behaviour can be supported by using strict PQ without a bandwidth limitation, placing VoIP traffic in the highest-priority queue and allowing the VoIP queue to consume bandwidth that would normally be allocated to the lower-priority queues, if necessary. A provider might be willing to support this type of behaviour if the penalties imposed by the regulatory agency exceed the rebates it is required to provide other subscribers for diminished service.

Rate-controlled PQ allows packets in a high-priority queue to be scheduled before packets in lower-priority queues, only if the amount of traffic in the high-priority queue stays below a user-configured threshold. For example, assume that a high-priority queue has been rate-limited to 20 percent of the output port bandwidth. As long as the high-priority queue consumes less than 20 percent of the output port bandwidth, packets from this queue are scheduled ahead of packets from lower-priority queues. However, if the high-priority queue consumes more than 20 percent of the output port bandwidth, packets from lower-priority queues can be scheduled ahead of packets from the high-priority queue. When this occurs, there are no standards, so each vendor determines how its implementation schedules lower-priority packets ahead of high-priority packets.

3.3.2.3 PQ Implementations and Applications

There are two primary applications for PQ at the edges and in the core of a network:

- PQ can enhance network stability during periods of congestion by allowing one to assign routing-protocol and other types of network-control traffic to the highest-priority queue.
- PQ supports the delivery of a high-throughput, low-delay, low-jitter and low-loss service class. This capability allows one to deliver real-time applications, such as interactive voice or video.

However, support for these types of services requires that one effectively conditions traffic at the edges of one's network to prevent high-priority queues from becoming oversubscribed. If one neglects this, it will be discovered that it is impossible to support these services.

3.3.3 FQ (FAIR QUEUING) [15]

FQ is the foundation for a class of queue scheduling disciplines that are designed to ensure that each flow has fair access to network resources and to prevent a bursty flow from consuming more than its fair share of bandwidth. In FQ, packets are first classified into flows by the system and then assigned to a queue that is specifically dedicated to that flow. Queues are then serviced one packet at a time in round-robin order. Empty queues are skipped. FQ is also referred to as per-flow or flow-based queuing. An example of this queuing algorithm can be seen in *Figure 3-3.*

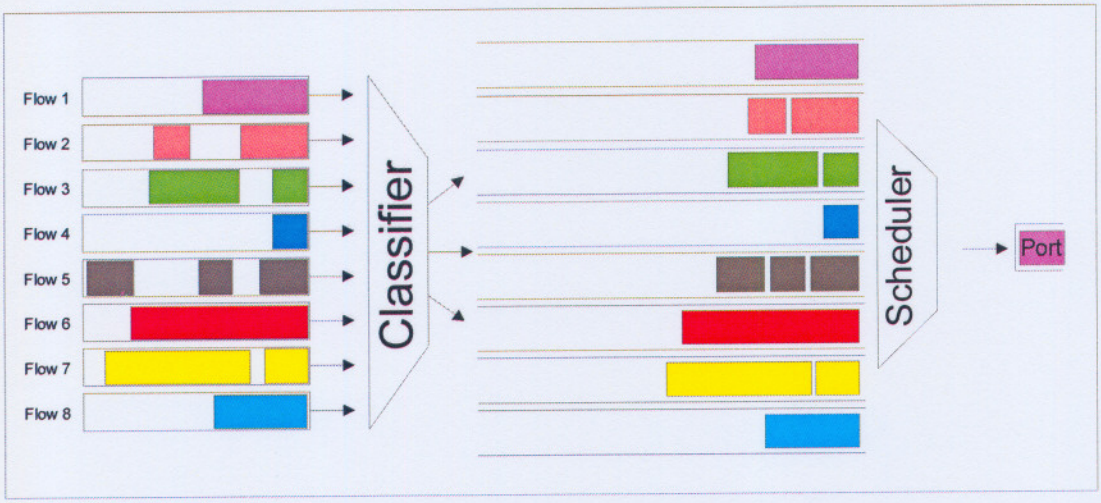

Figure 3-3: Fair Queuing Example

3.3.3.1 FQ Benefits and Limitations

The primary benefit of FQ is that an extremely bursty or misbehaving flow does not degrade the quality of service delivered to other flows, because each flow is isolated into its own queue.

If a flow attempts to consume more than its fair share of bandwidth, then only its queue is affected, so there is no impact on the performance of the other queues on the shared output port.

FQ has several limitations:

- Vendor implementations of FQ are implemented in software, not hardware. This limits the application of FQ to low-speed interfaces at the edges of the network.
- The objective of FQ is to allocate the same amount of bandwidth to each flow over time. FQ is not designed to support a number of flows with different bandwidth requirements.
- FQ provides equal amounts of bandwidth to each flow only if all of the packets in all of the queues are the same size. Flows containing mostly large packets get a larger share of output port bandwidth than flows containing predominantly small packets.

- FQ is sensitive to the order of packet arrivals. If a packet arrives in an empty queue immediately after the queue is visited by the round-robin scheduler, the packet has to wait in the queue until all of the other queues have been serviced before it can be transmitted.
- FQ does not provide a mechanism that allows one to easily support real-time services, such as VoIP.
- FQ assumes that one can easily classify network traffic into well-defined flows. In an IP network, this is not as easy as it might first appear. One can classify flows based on a packet's source address, but then each workstation is given the same amount of network resources as a server or mainframe. If one attempts to classify flows based on the TCP connection, then one has to look deeper into the packet header and deal with other issues resulting from encryption, fragmentation and UDP flows. Finally, one might consider classifying flows based on source/destination address pairs. This gives an advantage to servers that have many different sessions, but still provides more than a fair share of network resources to multitasking workstations.
- Depending on the specific mechanism used to classify packets into flows, FQ generally cannot be configured on core routers, because a core router would be required to support thousands or tens of thousands of discrete queues on each port. This increases complexity and management overhead, which adversely impacts the scalability of FQ in large IP networks.

3.3.3.2 FQ Implementations and Applications

FQ is typically applied at the edges of the network, where subscribers connect to their service provider. FQ requires minimal configuration (it is either enabled or disabled) and is self-optimizing — each of the n active queues is allocated $1/n$ of the output port bandwidth. As the number of queues changes, the bandwidth allocated to each of the queues changes. For example, if the number of queues increases from n to $(n+1)$, then the amount of bandwidth

allocated to each of the queues is decreased from $1/n$ of the output port bandwidth to $1/(n+1)$ of the output port bandwidth.

FQ provides excellent isolation of individual traffic flows because, at the edges of the network, a typical subscriber has a limited number of flows, so each flow can be assigned to a dedicated queue, or else a very small number of flows, at most, are assigned to each queue. This reduces the impact that a single misbehaving flow can have on all of the other flows traversing the same output port.

In class-based FQ, the output port is divided into a number of different service classes. Each service class is allocated a user-configured percentage of the output port bandwidth. Then, within the bandwidth block allocated to each of the service classes, FQ is applied. As a result, all of the flows assigned to a given service class are provided equal shares of the aggregate bandwidth configured for that specific service class.

3.3.4 WFQ (WEIGTED FAIR QUEUING) [15]

WFQ is the basis for a class of queue scheduling disciplines that are designed to address limitations of the FQ model:

- WFQ supports flows with different bandwidth requirements by giving each queue a weight that assigns it a different percentage of output port bandwidth.
- WFQ also supports variable-length packets, so that flows with larger packets are not allocated more bandwidth than flows with smaller packets. Supporting the fair allocation of bandwidth when forwarding variable-length packets adds significantly to the computational complexity of the queue scheduling algorithm. This is the primary reason that queue scheduling disciplines have been much easier to implement in fixed-length, cell-based ATM networks than in variable-length, packet-based IP networks.

3.3.4.1 WFQ Algorithm

WFQ supports the fair distribution of bandwidth for variable-length packets by approximating a generalized processor sharing system. While generalized processor sharing is a theoretical scheduler that cannot be implemented, its behaviour is similar to a weighted bit-by-bit round-robin scheduling discipline. In a weighted bit-by-bit round-robin scheduling discipline the individual bits from packets at the head of each queue are transmitted in a Weighted Round Robin (WRR) manner. This approach supports the fair allocation of bandwidth, because it takes packet length into account. As a result, at any moment in time, each queue receives its configured share of output port bandwidth. If one imagines the placement of a packet reassembler at the far end of the link, the order in which each packet would eventually be fully assembled is determined by the order in which the last bit of each packet is transmitted. This is referred to as the packet's finish time.

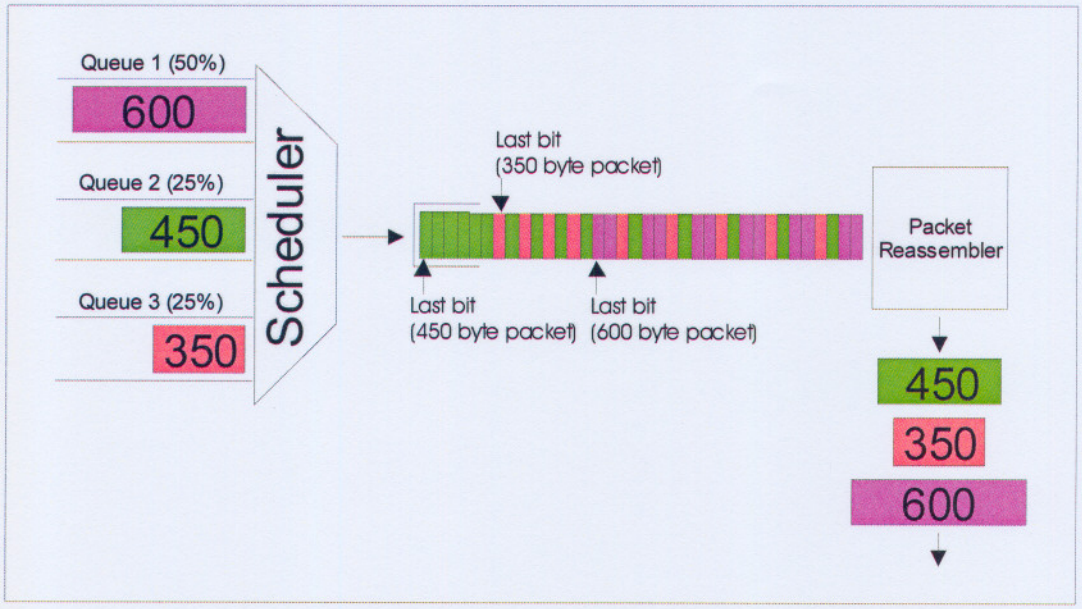

Figure 3-4: Bit-wise Weighted Fair Queuing Example

Figure 3-4 shows a weighted bit-by-bit round-robin scheduler servicing three queues. Assume that queue 1 is assigned 50 percent of the output port

bandwidth and that queues 2 and 3 are each assigned 25 percent of the bandwidth. The scheduler transmits two bits from queue 1, one bit from queue 2, one bit from queue 3 and then returns to queue 1. As a result of the weighted scheduling discipline, the last bit of the 600-byte packet is transmitted before the last bit of the 350-byte packet, and the last bit of the 350-byte packet is transmitted before the last bit of the 450-byte packet. This causes the 600-byte packet to finish (complete reassembly) before the 350-byte packet, and the 350-byte packet to finish before the 450-byte packet.

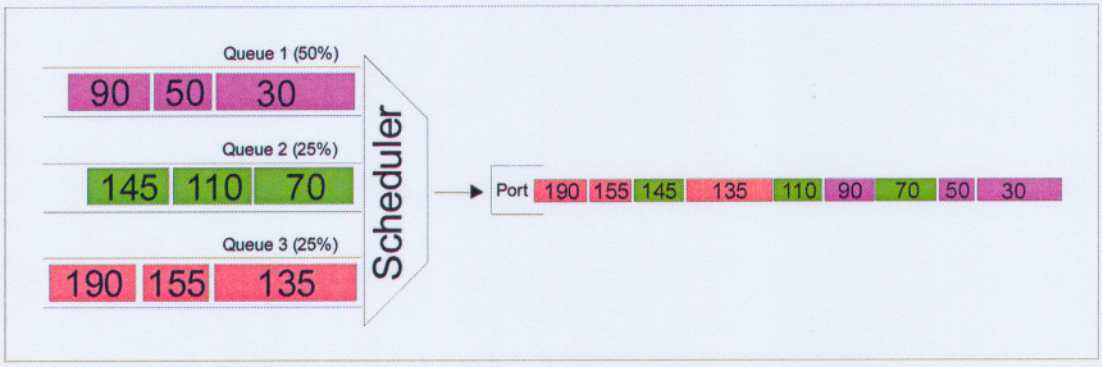

Figure 3-5: Weighted Fair Queuing Example

WFQ approximates this theoretical scheduling discipline by calculating and assigning a finish time to each packet. Given the bit rate of the output port, the number of active queues, the relative weight assigned to each of the queues and the length of each of the packets in each of the queues, it is possible for the scheduling discipline to calculate and assign a finish time to each arriving packet. The scheduler then selects and forwards the packet that has the earliest (smallest) finish time from among all of the queued packets. It is important to understand that the finish time is not the actual transmission time for each packet. Instead, the finish time is a number assigned to each packet that represents the order in which packets should be transmitted on the output port. An example of this can be seen in *Figure 3-5*.

When each packet is classified and placed into its queue, the scheduler calculates and assigns a finish time for the packet. As the WFQ scheduler

services its queues, it selects the packet with the earliest (smallest) finish time as the next packet for transmission on the output port. For example, if WFQ determines that packet A has a finish time of 30, packet B has a finish time of 70 and packet C has a finish time of 135, then packet A is transmitted before packet B or packet C. In *Figure 3-5*, observe that the appropriate weighting of queues allows a WFQ scheduler to transmit two or more consecutive packets from the same queue.

3.3.4.2 WFQ Benefits and Limitations

Weighted fair queuing has two primary benefits:

- WFQ provides protection to each service class by ensuring a minimum level of output port bandwidth independent of the behaviour of other service classes.
- When combined with traffic conditioning at the edges of a network, WFQ guarantees a weighted fair share of output port bandwidth to each service class with a bounded delay.

However, weighted fair queuing comes with several limitations:

- Vendor implementations of WFQ are implemented in software, not hardware. This limits the application of WFQ to low-speed interfaces at the edges of the network.
- Highly aggregated service classes mean that a misbehaving flow within the service class can impact the performance of other flows within the same service class.
- WFQ implements a complex algorithm that requires the maintenance of a significant amount of per-service class state and iterative scans of state on each packet arrival and departure.
- Computational complexity impacts the scalability of WFQ when attempting to support a large number of service classes on high-speed interfaces.
- On high-speed interfaces, minimizing delay to the granularity of a single packet transmission may not be worth the computational

expense if you consider the insignificant amount of serialization delay introduced by high-speed links and the lower computational requirements of other queue scheduling disciplines.

- Finally, even though the guaranteed delay bounds supported by WFQ may be better than for other queue scheduling disciplines, the delay bounds can still be quite large.

3.3.4.3 WFQ Implementations and Applications

WFQ is deployed at the edges of the network to provide a fair distribution of bandwidth among a number of different service classes. WFQ can generally be configured to support a range of behaviours:

- WFQ can be configured to classify packets into a relatively large number of queues.
- WFQ can be configured to allow the system to schedule a limited number of queues that carry aggregated traffic flows. Each of the queues is allocated a different percentage of output port bandwidth based on the weight that the system calculates for each of the service classes. This approach allows the system to allocate different amounts of bandwidth to each queue based on the QoS policy group or to allocate increasing amounts of bandwidth to each queue as the IP precedence increases.
- An enhanced version of WFQ, sometimes referred to as class-based WFQ, can be used alternately to schedule a limited number of queues that carry aggregated traffic flows. For this configuration option, user-defined packet classification rules assign packets to queues that are allocated a user-configured percentage of output port bandwidth. This approach allows one to determine precisely what packets are grouped in a given service class and to specify the exact amount of bandwidth allocated to each service class.

3.3.5 WRR (WEIGHTED ROUND ROBIN) [15]

Weighted round robin (WRR) is the foundation for a class of queue scheduling disciplines that are designed to address the limitations of the FQ and PQ models.

- WRR addresses the limitations of the FQ model by supporting flows with significantly different bandwidth requirements. With WRR queuing, each queue can be assigned a different percentage of the output port's bandwidth.
- WRR addresses the limitations of the strict PQ model by ensuring that lower-priority queues are not denied access to buffer space and output port bandwidth. With WRR queuing, at least one packet is removed from each queue during each service round.

3.3.5.1 WRR Queuing Algorithm

In WRR queuing, packets are first classified into various service classes (for example, real-time, interactive and file transfer) and then assigned to a queue that is specifically dedicated to that service class. Each of the queues is serviced in a round-robin order. Similar to strict PQ and FQ, empty queues are skipped. Weighted round robin queuing is also referred to as class-based queuing (CBQ) or custom queuing.

WRR queuing supports the allocation of different amounts of bandwidth to different service class by either:

- Allowing higher-bandwidth queues to send more than a single packet each time that it is visited during a service round, or
- Allowing each queue to send only a single packet each time that it is visited, but to visit higher-bandwidth queues multiple times in a single service round.

In *Figure 3-6*, the real-time traffic queue is allocated 25 percent of the output port bandwidth, the interactive traffic queue is allocated 25 percent of the

output port bandwidth and the file transfer traffic queue is allocated 50 percent of the output port bandwidth. WRR queuing supports this weighted bandwidth allocation by visiting the file transfer queue twice during each service round.

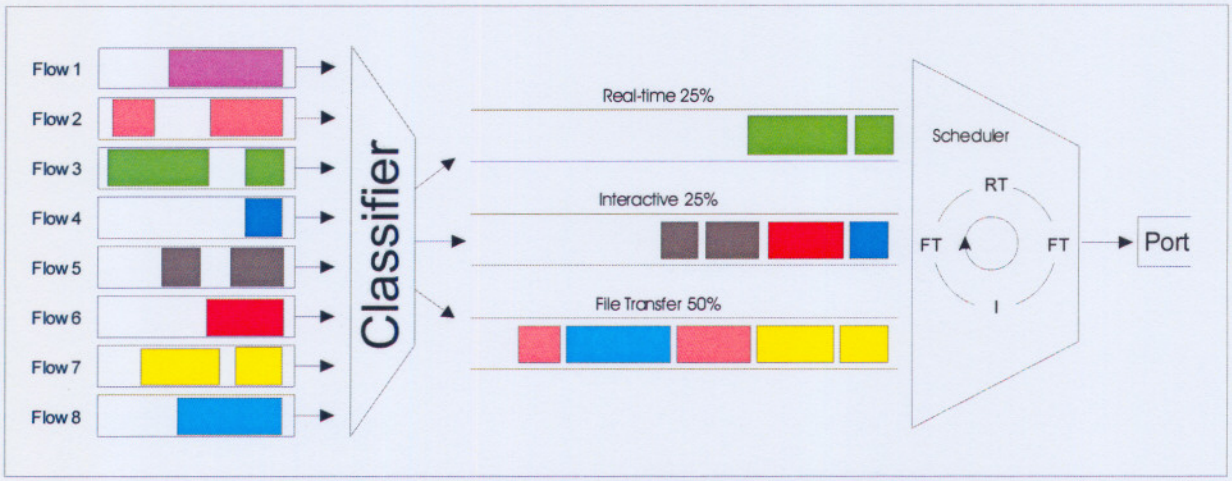

Figure 3-6 : Weighted Round Robin Queuing Example

To regulate the amount of network resources allocated to each service class, a number of parameters can be tuned to control the desired behaviour of each queue:

- The amount of delay experienced by packets in a given queue is determined by a combination of the rate that packets are placed into the queue, the depth of the queue, the amount of traffic removed from the queue at each service round and the number of other service classes (queues) configured on the output port.
- The amount of jitter experienced by packets in a given queue is determined by the variation of the delay in the queue, the variation of delay in all of the other queues and the variation of the interval between service rounds.
- The amount of packet loss experienced by each queue is determined by a combination of the rate that packets are placed into the queue and the depth of the queue. The fill rate can be controlled by performing traffic conditioning at some upstream point in the network.

3.3.5.2 WRR Queuing Benefits and Limitations

Weighted round robin queuing includes the following benefits:

- WRR queuing can be implemented in hardware, so it can be applied to high-speed interfaces in both the core and at the edges of the network.
- WRR queuing provides coarse control over the percentage of output port bandwidth allocated to each service class.
- WRR queuing ensures that all service classes have access to at least some configured amount of network bandwidth to avoid bandwidth starvation.
- WRR queuing provides an efficient mechanism to support the delivery of differentiated service classes to a reasonable number of highly aggregated traffic flows.
- Classification of traffic by service class provides more equitable management and more stability for network applications than the use of priorities or preferences. For example, if one assigns real-time traffic strict priority over file-transfer traffic, then an excessive amount of real-time traffic can eliminate all file-transfer traffic from the network. WRR queuing is based on the belief that resource reduction is a better mechanism to control congestion than resource denial. Resource denial not only blocks all traffic from lower-priority service classes but also obstructs all signalling regarding the denial of resources. As a result, TCP applications and externally clocked UDP applications are unable to adapt their transmission rates correctly to respond to the denial of network resources.

The primary limitation of weighted round-robin queuing is that it provides the correct percentage of bandwidth to each service class only if all of the packets in all of the queues are the same size or when the mean packet size is known in advance. For example, assume that one is using WRR to service four queues that are assigned the following percentages of output port bandwidth: queue 1 is allocated 40 percent, queue 2 is allocated 30 percent, queue 3 is allocated 20 percent and queue 4 is allocated 10 percent. Assume also that all of the packets in all of the queues are 100 bytes.

In the case where one service class contains a larger average packet size than another service class, the service class with the larger average packet size obtains more than its configured share of output port bandwidth. Assume that the WRR scheduler is configured exactly as in the previous example. However, all of the packets in queue 1 have an average size of 100 bytes, all of the packets in queue 2 have an average size of 200 bytes, all of the packets in queue 3 have an average packet size of 300 bytes and all of the packets in queue 4 have an average packet size of 400 bytes.

Assuming these average packet sizes, at the end of a single service round, queue 1 transmits 4 packets (400 bytes), queue 2 transmits 3 packets (600 bytes), queue 3 transmits 2 packets (600 bytes) and queue 4 transmits 1 packet (400 bytes). Since a total of 2000 bytes are transmitted during the service round, queue 1 receives 20 percent of the bandwidth, queue 2 receives 30 percent of the bandwidth, queue 3 receives 30 percent of the bandwidth and queue 4 receives 20 percent of the bandwidth. When faced with variable length packets, WRR queuing does not support the configured distribution of output port bandwidth.

3.3.5.3 WRR Implementations and Applications

Because the WRR scheduling discipline can be implemented in hardware, it can be deployed in both the core and at the edges of the network to arbitrate the weighted distribution of output port bandwidth among a fixed number of service classes. WRR effectively overcomes the limitations of FQ by scheduling service classes that have different bandwidth requirements.

WRR also overcomes the limitations of strict PQ by ensuring that lower-priority queues are not bandwidth-starved. However, WRR's inability to support the precise allocation of bandwidth when scheduling variable-length packets is a critical limitation that needs to be addressed.

3.3.6 DWRR (DEFICIT WEIGHTED ROUND ROBIN) [15]

DWRR is the basis for a class of queue scheduling disciplines that are designed to address the limitations of the WRR and WFQ models.

- DWRR addresses the limitations of the WRR model by accurately supporting the weighted fair distribution of bandwidth when servicing queues that contain variable-length packets.
- DWRR addresses the limitations of the WFQ model by defining a scheduling discipline that has lower computational complexity and that can be implemented in hardware. This allows DWRR to support the arbitration of output port bandwidth on high-speed interfaces in both the core and at the edges of the network.

In DWRR queuing, each queue is configured with a number of parameters:

- A weight that defines the percentage of the output port bandwidth allocated to the queue.
- A Deficit Counter that specifies the total number of bytes that the queue is permitted to transmit each time that it is visited by the scheduler. The Deficit Counter allows a queue that was not permitted to transmit in the previous round because the packet at the head of the queue was larger than the value of the Deficit Counter to save transmission "credits" and use them during the next service round.
- A quantum of service that is proportional to the weight of the queue and is expressed in terms of bytes. The Deficit Counter for a queue is incremented by the quantum each time that the queue is visited by the scheduler. If *quantum[i] = 2*quantum[x]*, then queue i will receive twice the bandwidth as queue x when both queues are active.

3.3.6.1 DWRR Algorithm

In the classic DWRR algorithm, the scheduler visits each non-empty queue and determines the number of bytes in the packet at the head of the queue. The variable Deficit Counter is incremented by the value quantum. If the size

of the packet at the head of the queue is greater than the variable Deficit Counter, then the scheduler moves on to service the next queue. If the size of the packet at the head of the queue is less than or equal to the variable Deficit Counter, then the variable Deficit Counter is reduced by the number of bytes in the packet and the packet is transmitted on the output port.

The scheduler continues to dequeue packets and decrement the variable Deficit Counter by the size of the transmitted packet until either the size of the packet at the head of the queue is greater than the variable Deficit Counter or the queue is empty. If the queue is empty, the value of Deficit Counter is set to zero. When this occurs, the scheduler moves on to service the next non-empty queue.

3.3.6.2 DWRR Example

For this example, as shown in *Figure 3-7*, assume that DWRR scheduling is enabled on an output port that is configured to support 3 queues. Queue 1 is allocated 50 percent of the bandwidth, while queues 2 and 3 are each allocated 25 percent of output port bandwidth.

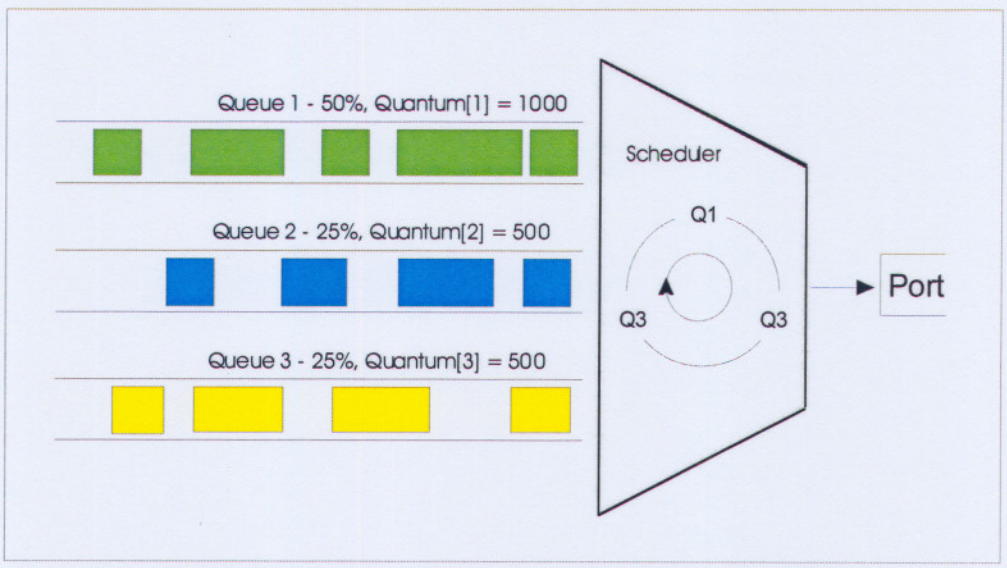

Figure 3-7: Deficit Weighted Round Robin Queuing Example

Initially, the array variable Deficit Counter is initialized to 0. Assume that the round robin pointer points to queue 1, which is at the top of the Active List. Before the DWRR scheduling discipline begins to service queue 1, *Quantum[1] = 1000* is added to *Deficit Counter[1]*, giving it a value of 1000.

Because the 600-byte packet at the head of queue 1 is smaller than the value of *Deficit Counter[1] = 1000*, the 600-byte packet is transmitted. This causes the *Deficit Counter[1]* to be decremented by 600 bytes, resulting in a new value of 400. Now, since the 300-byte packet at the head of queue 1 is again smaller than *Deficit Counter[1] = 400*, the 300-byte packet is also transmitted, and this causes *Deficit Counter[1]* to be decremented by 300 bytes, creating a new value of 100. Because the 400-byte packet at the head of queue 1 is larger than the value of *Deficit Counter[1] = 100*, the 400-byte packet cannot be transmitted. This causes the round robin pointer to point to queue 2, which is now at the top of the Active List.

Before the DWRR scheduling discipline starts to service queue 2, *Quantum[2] = 500* is added to *Deficit Counter[2]* giving it a value of 500. Since the 400-byte packet at the head of queue 2 is smaller than the value of *Deficit Counter[2] = 500*, the 400-byte packet is transmitted. This causes *Deficit Counter[2]* to be decremented by 400 bytes and, therefore, have a new value of 100.

Because the 300-byte packet at the head of queue 2 is larger than the value of *Deficit Counter[2] = 100*, the 300-byte packet cannot be transmitted. This causes the round robin pointer to point to queue 3, which is now at the top of the *Active List.*

Before the DWRR scheduling discipline starts to service queue 3, *Quantum[3] = 500* is added *to Deficit Counter[3],* giving it a value of 500. Since the 600-byte packet at the head of queue 2 is larger than the value *of Deficit Counter[3] = 500*, the 600-byte packet cannot be transmitted. This causes the

round robin pointer to point to queue 1, which is now at the top of the *Active List*.

Before the DWRR scheduling discipline starts to service queue 1, *Quantum[1] = 1000* is added to *Deficit Counter[1]* giving it a value of 1100. Since the 400-byte packet at the head of queue 1 is smaller than the value of *Deficit Counter[1] = 1100*, the 400-byte packet is transmitted. This causes *Deficit Counter[1]* to be decremented by 400 bytes and to have a new value of 700. Now queue 1 is empty. This causes *Deficit Counter[1]* to be set to zero, queue 1 to be removed from the *Active List* and the round robin pointer to point to queue 2, which is now at the top of the *Active List*.

The algorithm will continue operating in this way, thereby providing each queue with its allocated amount of bandwidth.

3.3.6.3 DWRR Benefits and Limitations

DWRR has the following benefits:

- Provides protection among different flows, so that a poorly behaved service class in one queue cannot impact the performance provided to other service classes assigned to other queues on the same output port;
- Overcomes the limitations of WRR by providing precise controls over the percentage of output port bandwidth allocated to each service class when forwarding variable-length packets;
- Overcomes the limitations of strict PQ by ensuring that all service classes have access to at least some configured amount of output port bandwidth to avoid bandwidth starvation; and
- Implements a relatively simple and inexpensive algorithm from a computational perspective, that does not require the maintenance of a significant amount of per-service class state.

As with other models, DWRR queuing has limitations:

- Highly aggregated service classes mean that a misbehaving flow within a service class can impact the performance of other flows within the same service class. However, in the core of a large IP network, routers are required to schedule aggregate flows, because the large number of individual flows makes it impractical to support per-flow queue scheduling disciplines.
- DWRR does not provide end-to-end delay guarantees as precise as other queue scheduling disciplines do.
- DWRR may not be as accurate as other queue scheduling disciplines. However, over high-speed links, the accuracy of bandwidth allocation is not as critical as over low-speed links.

3.3.6.4 DWRR Implementations and Applications

Because the DWRR queue scheduling discipline can be implemented in hardware, it can be deployed in both the core and at the edges of the network to arbitrate the weighted distribution of output port bandwidth among a fixed number of service classes. DWRR provides all of the benefits of WRR, while also addressing the limitations WRR by supporting the accurate allocation of bandwidth when scheduling variable-length packets.

3.3.7 MDRR (MODIFIED DEFICIT ROUND ROBIN) [16]

The Modified Deficit Round-Robin (MDRR) scheduling algorithm is based on the Deficit Round-Robin (DRR) mechanism which implements a number of queues that are served in a round-robin fashion. A configurable value called a service quantum is assigned to each queue in DRR. The service quantum provides a measure of how much traffic should be handled from the queue in each round. Packets from that queue are serviced until their cumulative length (byte count) exceeds the service quantum. A deficit counter, which is a memory mechanism designed to enhance fairness and packet size independence, is used as a credit mechanism. The deficit counter value is

added to the service quantum to determine the measure of service available for each queue during each round.

MDRR extends the DRR mechanisms by including for each set of class-of-service queues a low-latency, high-priority (LLHP) queue designed to handle special traffic, like VoIP, different from the other queues. Except for the LLHP queue, MDRR services all queues in round-robin fashion.

With MDRR configured as the queuing strategy, non-empty queues are served one after the other in a round-robin fashion. Each time a queue is served, a fixed amount of data is dequeued. The algorithm then services the next queue. When a queue is served, MDRR keeps track of the number of bytes of data that were dequeued in excess of the configured value. In the next pass, when the queue is served again, less data will be dequeued to compensate for the excess data that were served previously. As a result, the average amount of data dequeued per queue will be close to the configured value. In addition, MDRR maintains a priority queue that gets served on a preferential basis. MDRR is explained in greater detail below.

Each queue within MDRR is defined by two variables:

- Quantum value - Average number of bytes served in each round.
- Deficit Counter - This counter is used to track how many bytes a queue has transmitted in each round. It is initialized to the quantum value.

Packets in a queue are served as long as the deficit counter is greater than zero. Each packet served decreases the deficit counter by a value equal to its length in bytes. A queue can no longer be served after the deficit counter becomes zero or negative. In each new round, each non-empty queue's deficit counter is incremented by its quantum value.

3.3.8 RED (RANDOM EARLY DETECTION) [16][17]

The RED mechanism was proposed by Sally Floyd and Van Jacobson in the early 1990s to address network congestion in a responsive rather than reactive manner. RED functions on the assumption that most traffic is transported on data networks that are sensitive to packet loss. Protocols like TCP respond to packet loss by temporarily slowing down its transmission rate. This effectively allows RED's traffic-drop behaviour to manipulate TCP traffic, thereby working as a congestion-avoidance signalling mechanism.

TCP is the most widely used network transport protocol. Given the wide use of TCP, RED offers a widespread, effective congestion-avoidance mechanism. RED aims to control the average queue size by indicating to the end hosts when they should temporarily slow down transmission of packets.

RED takes advantage of TCP's congestion control mechanism. By randomly dropping packets prior to periods of high congestion, RED tells the packet source to decrease its transmission rate. Assuming the packet source is using TCP, it will decrease its transmission rate until all the packets reach their destination, indicating that the congestion is cleared. RED keeps the queue depth low in order to allow for spikes in bandwidth use to occur without congesting the network.

In *Figure 3-8*, a queue occupancy state of 25 percent (Minimum Threshold) indicates that there is 0 percent chance that a packet will be dropped, a queue occupancy state of 50 percent indicates that there is a 25 percent chance that a packet will be dropped, a queue occupancy state of 75 percent indicates that there is a 50 percent chance that a packet will be dropped, and queue occupancy state greater than 85 percent (Maximum Threshold) indicates that there is a 100 percent chance that a packet will be dropped from the queue.

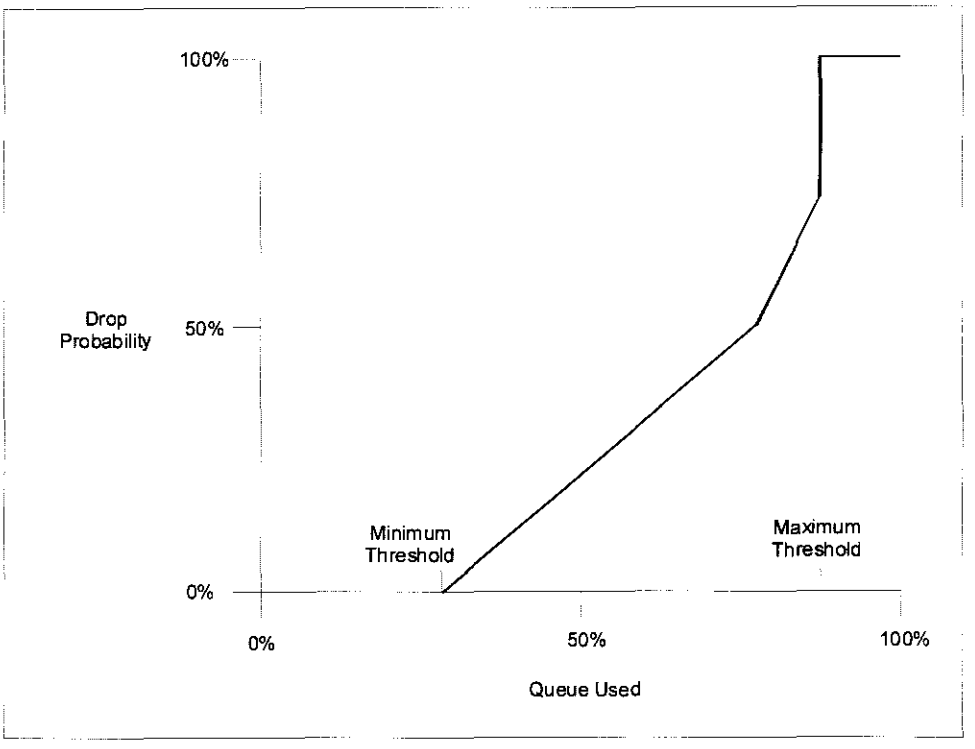

Figure 3-8: RED Packet Dropping Probability

3.3.8.1 RED Benefits and Limitations

The benefits of RED queue management include:

- RED does not require changes to currently deployed host transport-layer protocols.
- RED identifies the earliest stages of congestion and responds by randomly dropping packets. If the amount of congestion continues to increase, RED drops packets more aggressively to prevent the queue from reaching 100 percent capacity, which would result in a complete denial of service. This allows RED to maintain an upper bound on the average queue depth even with non-cooperative transport-layer protocols.
- Because RED does not wait until a queue is 100 percent full to begin discarding packets, RED allows a queue to accept bursts of traffic and not discard all of the packets in the burst. As a result, RED is TCP-

friendly because it does not discard clusters of packets from any single TCP session and helps avoid global TCP synchronization.

- RED allows one to maintain the amount of traffic in a queue at a moderate level. The amount of traffic in a queue is not kept artificially low, causing output port bandwidth to remain under utilized. Nor is the amount of traffic in a queue kept close to maximum capacity, where excessive packet dropping causes large numbers of TCP sessions to reduce their transmission rate, leading to poor utilization of output port bandwidth. RED allows one to maintain queue depths at a level that produces the best utilization of output port bandwidth.
- RED supports the fair discarding of packets across multiple flows without requiring the router to maintain state for the amount of traffic carried by each flow traversing a given queue. For example, RED is configured with a packet drop probability of 20 percent when a queue is 50 percent full. This means that one out of five packets will be discarded when the queue is 50 percent full. This drop profile will affect a flow that is sending 40 percent of the packets in the queue more than a flow that is sending only 5 percent of the packets in the queue.

The limitations of RED queue management include:

- RED can be complex to configure to achieve predictable performance.
- There is a lack of operational experience with the specific RED configuration parameters that work best in production networks.
- Incorrect configuration of RED parameters can result in output port bandwidth utilization that is worse than tail drop queue management.
- TCP flows are impacted by RED but non-TCP-based flows — such as the Internet Control Message Protocol (ICMP) and the User Datagram Protocol (UDP) — are not impacted by RED. When a TCP packet is dropped by RED, the source TCP assumes that the packet was dropped due to congestion and responds by reducing its transmission rate. When a non-TCP packet is dropped by RED, the source host does not know that a packet was dropped and does not alter its transmission rate. For this reason, it is recommended that for UDP-

based Voice over IP (VoIP) queues one does not enable RED and simply rely on tail drop. It is also recommended that one configures short queues for VoIP traffic to limit the maximum amount of delay experienced at any single hop in the end-to-end path.

- The dropping of packets by RED wastes network resources used to transmit the packets to their point of discard.

3.4 CONCLUSION

The fact that there is such a wide variety of queuing algorithms available for use shows that there is a lot of interest in the field of packet scheduling/queuing. The reason for this is that it definitely does have a positive impact on network performance.

The large selection of queuing algorithms in itself poses a problem, which algorithm should be chosen for a Field Force Automation system? These algorithms perform differently under different conditions. It is impossible to predict which algorithm will perform best in a certain system without simulating it or implementing it and testing.

For this reason a network simulation will be run to compare the performance of a selection of these queuing algorithms in a Field Force Automation environment. The goal of this will be to provide a guideline for Field Force System implementers for which queuing algorithm to implement and what performance increase they can expect from such an implementation.

4 Chapter 4 – Data Source Model Development

As discussed previously, the entire project consists of two sub-problems. This chapter addresses the first of these two, namely the development of a suitable Data Source Model for use in Field Force Automation network simulations. The approach followed as well as results achieved will be discussed.

4.1 INTRODUCTION

In this chapter the development of a suitable Data Source Model for Field Force Automation systems, that can be used in network and traffic simulations, will be discussed. Most new mobile Field Force Automation systems use mobile access technologies like GPRS, which in reality are still not quite broadband and, therefore, offer limited bandwidth that has to be managed. The development of the model is divided into two parts; the first is the development of a model for a single Field Force Automation transaction. The second is the development of a transaction frequency model which, when used in combination with the first model, gives an overall model for Field Force Automation behaviour. The influence of different model parameters is also addressed.

4.2 PROBLEM STATEMENT

The success of a Field Force Automation system is mainly based on the speed and efficiency with which information can be accessed by personnel. Field Force Automation system implementations are sometimes based on the assumption that the networks infrastructure used will not have any negative effects on the performance of the system. [1] This assumption has, however, not been tested. This can be risky given the fact that one of the main reasons for upgrading to the new system is to provide faster access to the Field Force Management database with shorter delays in accessing new information.

Undertaking an system implementation like this requires planning, part of which is the simulation of the proposed new system to ensure that such problems do not occur. Simulation of such a Field Force Automation system requires, among many other tools, a data source model. This model is needed to generate the simulated traffic in a way that closely resembles the real-world traffic.

An initial survey on the feasibility of a study concerning network utilisation of Field Force Automation applications revealed a lack of simulation tools. The first and foremost of which is the lack of a suitable data source model to use in the simulations [3]-[7].

4.3 GOAL

This project aims to provide companies considering Field Force Automation solutions with a scalable data source model. This model can be used in network simulations to assist with the planning of such a system. These simulations can range from initial simulations to determine the specifications of the bandwidth and server load handling requirements to future upgrade/scalability of the system. The model can also be used to perform network optimisation functions, such as traffic scheduling and load balancing.

The development of a procedure for developing such a model is subject to certain criteria generated by the need for it to be both simple to use and adaptable to different sized workforces [19]. The following criteria were set:

- Both models developed should be suitable for use in GPRS network simulations. The reason for this is that the main implementations of these Field Force Automation systems use GPRS networks as their backbone network.
- The single transaction model should be usable in both Wireless Modelling Language (WML) as well as Hyper Text Mark-up Language (HTML) based systems. This will facilitate different system implementations, instead of developing a fixed model for a certain type of system.
- The single transaction model should accurately represent a single Field Force Automation transaction.
- The incoming call frequency model should accurately represent the expected call patterns during a normal working day of a mobile

workforce. The model should also be scalable to accommodate different sized workforces as well as different workforce parameters.

- Neither models should be bound by rules set to a specific Field Force Automation system, but should be adaptable for different working environments.

The procedures followed to achieve these goals are set out in the following sections.

4.4 PREVIOUS WORK

Previous work on this topic includes the following studies:

- A framework on how a model for a GPRS Network Simulator is presented in [19]. This study was used for basic guidelines on what has to be taken into account when developing a data source model. This study, however, focused more on the control communications between the MS and the BTS than on the physical traffic being transmitted and received.
- A number of studies of the performance analysis of GPRS based networks, which mostly used standard World Wide Web (WWW) Data Source Models for their simulations. These studies can be found in [20]-[24]. The standard WWW data source model was proposed by the European Telecommunications Standards Institute (ETSI) and is suitable for simulating internet web-browsing traffic on the GPRS network.
- Another study also focused on the tools and procedures used in the analysis of network traffic [25]. This study assisted with the method of collecting the traffic information.

These WWW Data Source Models are, however, not usable in Field Force Automation simulations. The reason for this is that the WWW models are more stochastic in nature and rely on larger amounts of data being

exchanged. The following tables (*Table 4-1* to Table 4-3) were taken from three of the GPRS performance articles. They show the parameters of the traffic models used in their WWW traffic models:

Parameter	Model 1	Model 2	Model 3
Max Active GPRS Sesions	50	50	20
Ave session duration	2122s	2075s	312
Ave arrival rate of pkts	8Kbit/s	32Kbit/s	32Kbit/s
Ave duration of packet call	12,5s	3,1s	3,1s
Ave Reading time between calls	412s	412s	3,1s

Table 4-1: WWW Traffic Model Example 1 [4]

Qty	File Type	Size
1	index.html	40KB
20	JPEG/GIF	200B-2KB
20	GIF	2-5KB
4	GIF	5-10KB
1	GIF	>10KB

Table 4-2: WWW Traffic Model Example 2 [20]

Random variable	**Distribution**	**Mean**	**Std. Dev**
Number of pages per web session	Lognormal	25	100
Page viewing time	Weibull	39,5	92,6
Number of inline objects per page	Gamma	5,55	11,4
Size of main object	Lognormal	10kB	25kB
Size of inline object	Lognormal	7,7kB	126kB

Table 4-3: WWW Traffic Model Example 3 [24]

From the preceding tables it can be calculated that the average WWW transaction size is always in excess of 100KB. Because Field Force Automation transfers few, if any images, the transaction sizes are expected to be more in the range of 10KB.

The WWW models are also based on users doing casual internet browsing, therefore having long access times with great variations on reading times on pages. In the case of Field Force Automation, the deviation on reading times

and the session length will be very small as the personnel are working against time and according to a certain pattern.

4.5 MODEL DEVELOPMENT

As discussed earlier, there are two models that have to be developed. This is mainly because there are two different areas that together form the pattern of traffic generated. First is the model of a single transaction. If the personnel were distributed around the world, with no definitive schedule to keep, this model would be enough as there are no defined times when working starts and stops. That is one of the reasons that the WWW model is not sufficient for purposes of Field Force Automation modelling. The model in development here is, however, for a workforce situated in a single country with one time zone.

In the case of WWW traffic, the users are using the network both for business and for pleasure, meaning that WWW network usage, as opposed to Field Force Automation traffic, does not keep office hours. This is then also the purpose of the second model, to give a picture of the pattern in which the transaction will take place during a normal working day with defined office hours. This will help to give an idea of when peak times will be and what they will look like.

It is, however, important to notice that the development of the Data Source Model that will be discussed in the rest of this chapter is intended to serve as an example only and not to be taken as a standard model for Field Force Automation. The reason for this is that every company's implementation of their Field Force Automation system will be unique. The number of steps in every transaction and the amount of information being exchanged in each step will be different and the model will, therefore, have to be adjusted. But following the example will make this adjustment quick and painless.

4.5.1 FIELD FORCE AUTOMATION (FFA) [26]

There are millions of mobile workers performing a wide range of business functions. This field force is composed of sales representatives, service technicians, inspectors and doctors on rounds, among others. In this paper, the emphasis is on one aspect of the field force - those organizations dedicated to delivering field service. Field service, for purposes of this paper, is defined as a technician performing repair or installation work at a customer site. It may also include regularly scheduled preventive maintenance work.

Field service organizations vary greatly in size, target industries and supported technologies. For large international companies like AT&T and IBM, or local companies like Telkom S.A. and South African Breweries (SAB), their field service units number in the thousands whereas smaller companies may have only a few service crews. Although every industry relies on field service to some degree, for industries such as high technology, utilities, telecommunications, insurance and aerospace, field service is critical. Equipment from computers and peripherals to building systems, office equipment and medical equipment depend on field service.

The Field Force Automation Workflow is shown in *Figure 4-1*. It starts with a Customer call which causes a field crew to be dispatched. The crew then drives to the applicable site, where they perform the necessary repairs after which they check the inventory used. A closed work order is then logged with the Field Force Automation system, which is used to calculate the time used and billing information. The crew is then dispatched to their next location.

The three steps from Close Work Order to the next Dispatch are the steps that represent the action of contacting the Field Force Automation Server and exchanging information. This is then also the part that generates the network traffic and will be referred to as a Field Force Automation Transaction.

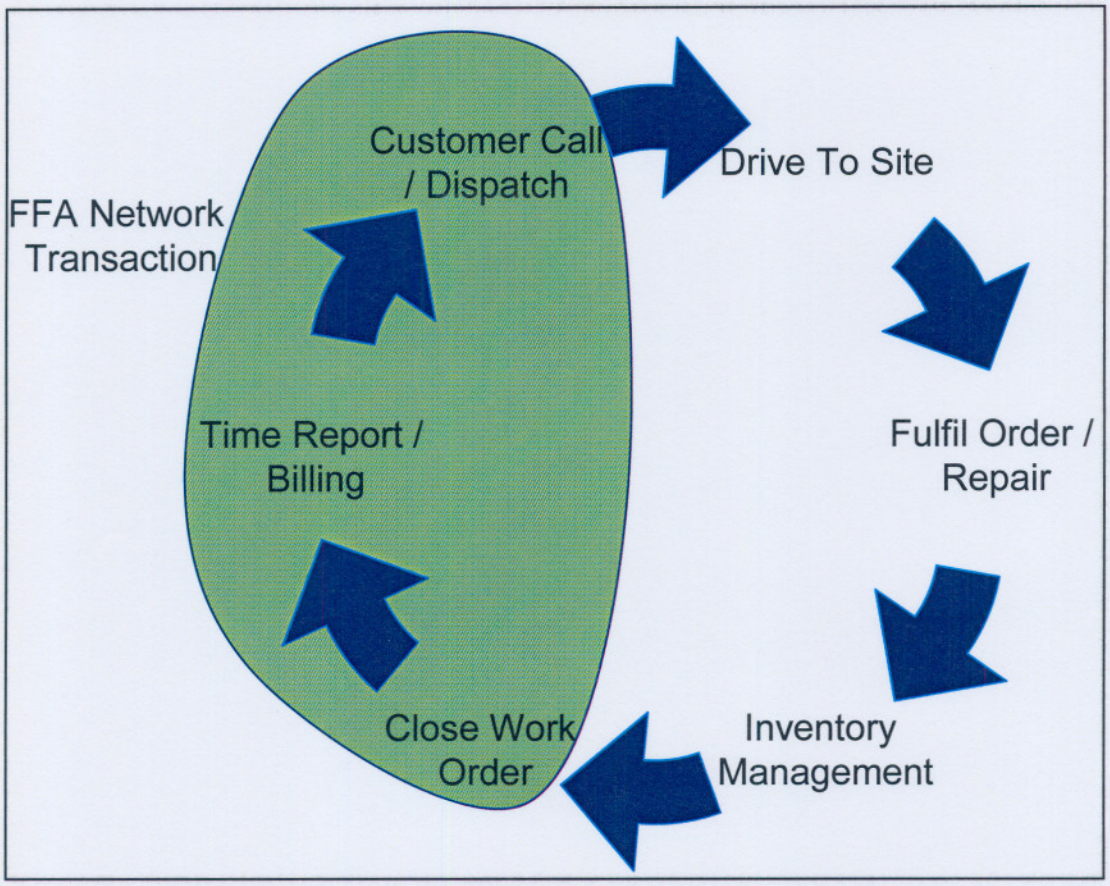

Figure 4-1: Field Force Automation Workflow

4.5.2 SINGLE TRANSACTION MODEL

The Single Transaction Model has to give an idea of what a single Field Force Automation transaction looks like without any influence from the network it is being transported over. This means that the model should not take into account any delays or data transmission which are network related as these delays are the responsibility of the simulator and will be added later on.

From this point forward, the physical development of the Data Source Model is based on a hypothetical system. The purpose is to show how a model can be developed for an individual application and the specific system on which the example is based is not of real importance, as the model has to be adaptable to different systems.

The Single Transaction Model consists of 5 separate steps. Firstly the default page is loaded, which is usually the page that allows the user login and authenticate. The next step is where the user actually types his username and password and submits it back to the server for verification. Upon successful authentication, the user is then forwarded to a page where he/she can submit a close work order for the current job and also submit a status report. When this is finished and submitted to the server, the information on the next job is displayed, containing information on the nature of work to be done as well as the location. *Figure 4-2* show a breakdown of such a Field Force Automation Transaction.

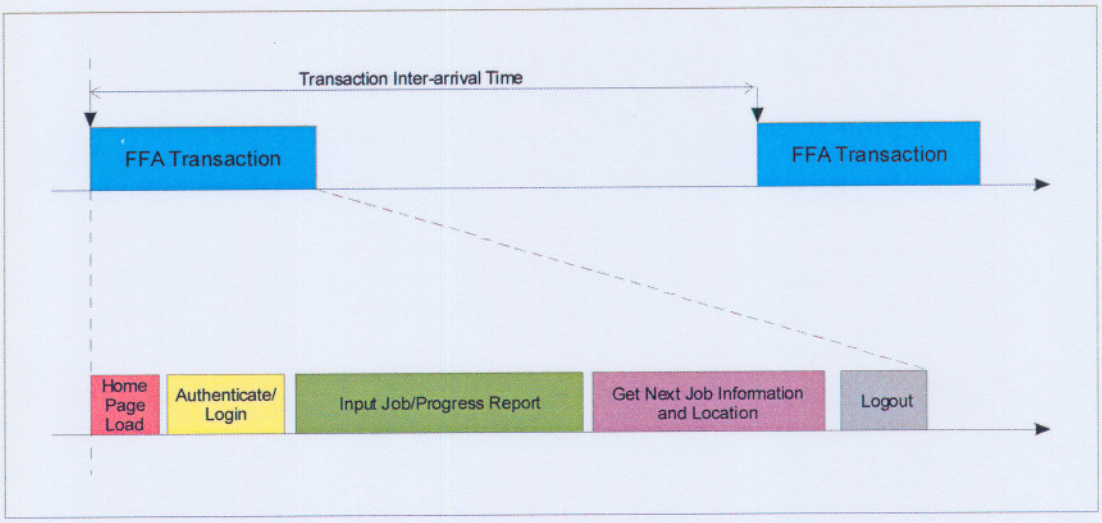

Figure 4-2: Field Force Automation Transaction

4.5.2.1 Data Gathering

The next step in the development of the model is to gather the data required to construct the detail of the model. To accomplish this, physical network data measurements were conducted. The tests were conducted using two cellular phones to conduct Field Force Automation transactions over a GPRS network. The operating system on the cellular phone itself could also have an effect on the measurements. To ensure this was not the case, two different models of phones were used that had different operating systems, with matching results.

The Field Force Automation transactions used WML based pages and included the following steps which are common to these transactions:

- Home Page Load
- User Authentication/Login
- Input Job Report
- Get Next Job Information
- Get Directions to Next Location (when applicable)
- Logout / Disconnect

These tests were conducted in such a manner that clear differentiation was made between network transport/delay times on the one hand and user interaction/read times on the other hand. This was done by taking time measurements of every transaction and splitting the time every time the wait changes from user to phone/network. The precise number of bytes sent and received in each step of the transaction was also recorded.

4.5.2.2 Results

Firstly, 50 transactions were conducted on a custom Field Force Automation system. There was no need for more transactions as they mostly follow the same pattern. Using statistical analysis of the data collected in these 50 transactions, the following results were obtained and used to build the single transaction model:

Parameter	Distribution	Mean	Std Deviation
A: Connect to Start Delay (s)	Normal	3	33%
A: Home Page Request Sent (bytes)	----	389	0%
A: Login Data Received (bytes)	----	796	0%
B: Login Time (s)	Normal	23	27%
B: Login Data Sent (bytes)	Normal	578	41%
B: Job Report Page Received (bytes)	Normal	2068	23%
C: Job Report Time (s)	Normal	117	22%
C: Job Report Data Sent (bytes)	Normal	2662	61%
C: Job & Location Data Received (bytes)	Normal	4936	20%
D: Read Time (s)	Normal	138	23%
D: Logout Request Sent (bytes)	Normal	354	13%
D: Logout Confirmation Received (bytes)	Normal	491	10%
Total Transaction Length (s)	Normal	281	23%
Total Data Received (bytes)	Normal	8291	23%
Total Data Sent (bytes)	Normal	3983	38%

Table 4-4: Table of processed single transaction results

When compared to a WWW data source model, the results in *Table 4-4* clearly show that there is a substantial difference between a Field Force Automation transaction and World Wide Web transaction. The easiest difference to spot is in the data exchanged in a transaction, which are less than 10KB (with 2.5KB deviation) for the entire session in Field Force Automation compared to 100KB (with a 200KB deviation) in the WWW models.

Figure 4-3 shows a graphical representation of the data in *Table 4-4*, with the letters A to D corresponding to steps of the transaction. When generating the graph, only the mean values were used to give a picture of what the average transaction will look like. Taking the deviation into account will introduce slight random changes in the position and height of the spikes.

The spikes will no longer be on top of one another once the network delays and bandwidth restrictions are taken into consideration, but that is where the actual simulation starts.

This model can be scaled to represent a HTML based transaction by simply running some more transactions on a HTML system and adapting the results, by using statistics, to provide the same parameters supplied in *Table 4-4*.

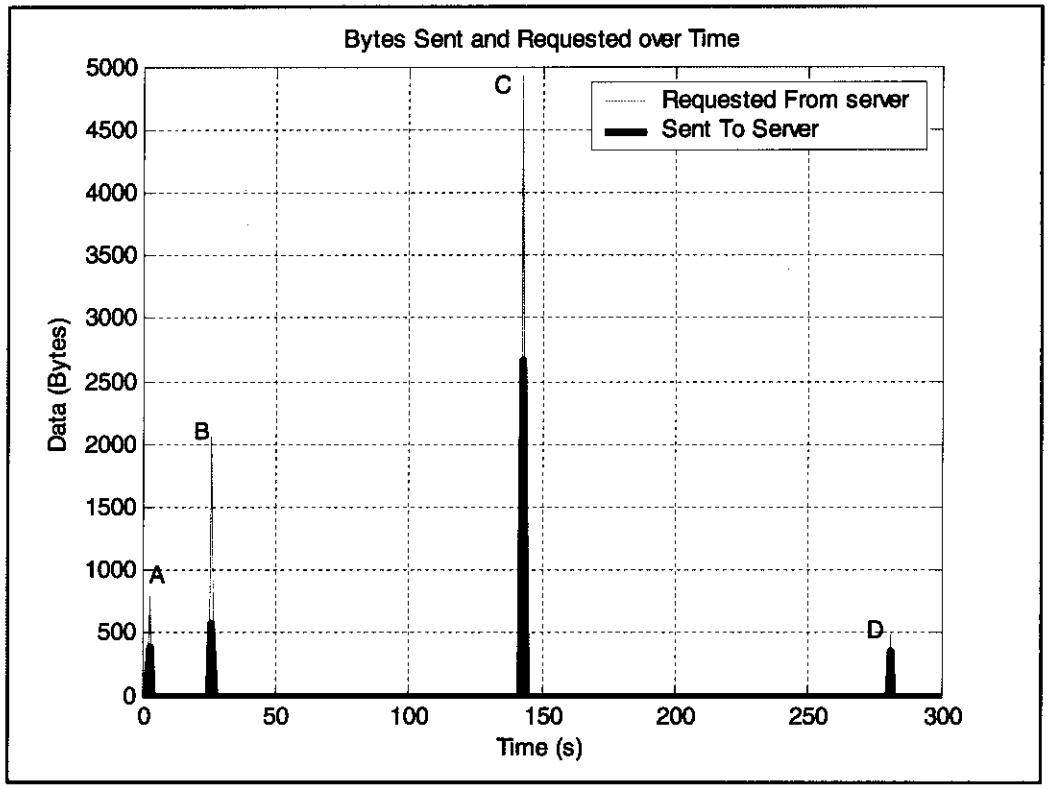

Figure 4-3: Data sent to, and requested from server over time

4.5.3 INCOMING CALL FREQUENCY MODEL

The second part of the study involves the development of a model to predict the incoming call frequency pattern that the server can expect to receive during a normal working day. This model is based on a set of rules that combines some parameters and statistics [27] to form the end result. A number of the guidelines were used from real-world systems, but once again this model is adaptable to different situations.

4.5.3.1 Parameter Selection

The following are model parameters that can be adjusted to suit different company needs:

- Workforce size
- Number of daily jobs

- Daily working hours
- Start of day & delay
- Lunch start
- Lunch duration
- Job length variation.

Workforce Size: For all the results in this section a workforce size of 5000 was chosen to generate the results. This number was chosen to represent Telkom, as they have 13000 technicians, working in groups of two, and some groups having jobs that take longer than a day to complete.

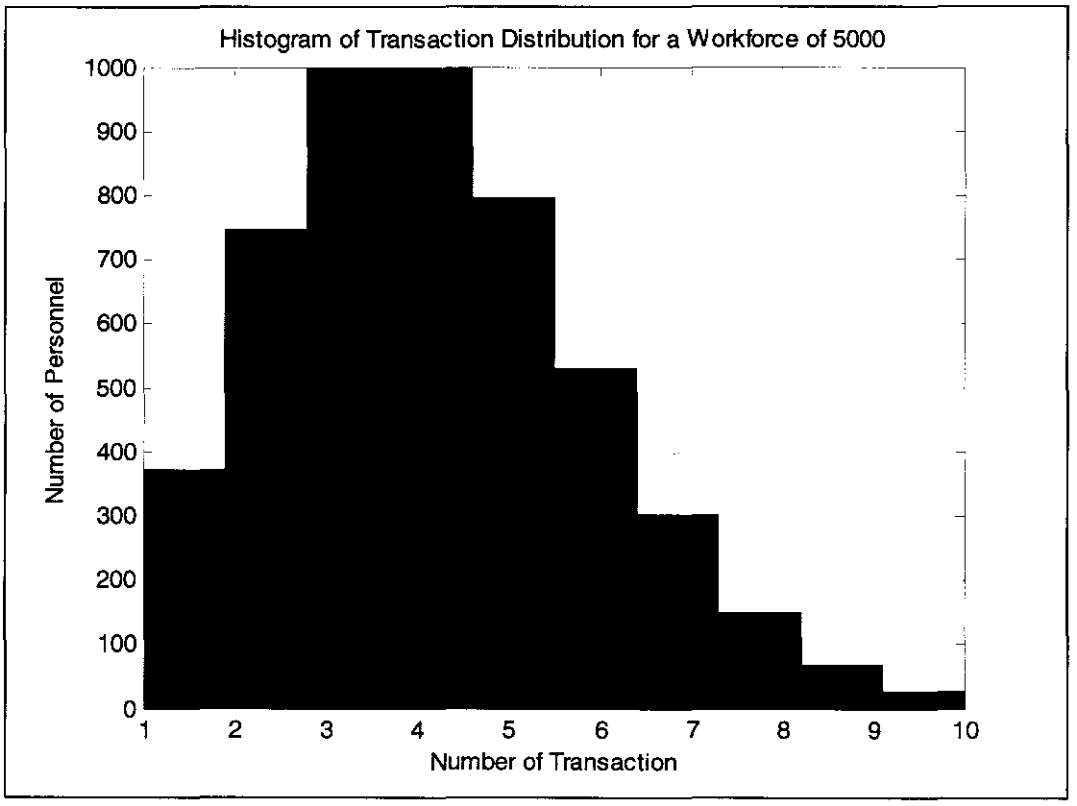

Figure 4-4: Histogram for the Distribution of Daily Transactions

Number of Daily Jobs: The second parameter that has to be defined is the number of transactions that a group will complete during a single working day. For this example, in literature it was found that an average number of 4

transactions are done per day [1]. Assuming that the entire job duration cannot be less than 45 minutes, including travelling, work and transaction, it calculates to a maximum of 10 transactions per day. Some larger jobs can take more than an entire day to complete, resulting in a minimum of 1 job per day as the groups performing less than 1 job per day have already been factored into the workforce size. Using these parameters, a distribution for the number of transactions per day was calculated as in *Figure 4-4*.

Daily Working Hours: The number of working hours per day was chosen as the standard 8 hour working day.

Start of Day and Delay: This is the first part of the day during which people start working. Officially everyone should start working at the same time each day, but realistically that does not happen. The purpose of this parameter is to incorporate that un-synchronised start of day into the model. For this parameter a value of 30 minutes was chosen with a uniform distribution. This means that the employees will start their work between 08:00 and 08:30 in the morning.

Lunch Start: This parameter determines the earliest possible time that an employee can have lunch. For this the value of 12:30 was chosen.

Lunch Duration: The duration of the lunch time was chosen as the standard 1 hour lunch.

Job Length Variation: The variation in job length parameters is intended to help make the model more realistic. If two employees both do 4 jobs for the day, this parameters help vary the job length so that they do not access the server at exactly the same times during the day.

The model was designed to use these parameters together with probability distributions, to give a more realistic representation on what to expect in a real

world system. A summary of the parameters and their values can be found in *Table 4-5.*

Parameters	Distribution	Minimum	Maximum	Mean
Daily Transactions per Person	Poisson	1	10	4
Workforce Size	----	5000	5000	5000
Working Hours per Day	----	8	8	8
Day Start Time	Uniform	08:00	08:30	08:15
Lunch Time Start	Normal	12:30	13:00	13:15
Lunch Duration (minutes)	----	60	60	60
Job Length Variation	Uniform	-10%	10%	0%
Transaction Duration	----	240	240	240

Table 4-5: Table of Model Parameters

4.5.3.2 Frequency Model Rules

The model also has a set of rules that was programmed into MATLAB® together with the parameters as shown above, to calculate the final call frequencies. The following rules were implemented:

- All the employees have to request their first job from the server within the set limits of the start of day delay parameter.
- No employee will be allowed to work less than the set number of working hours per day. If a job is finished at 17:00 and the employee only started work on 08:10, another job will be assigned to that employee.
- An employee will finish each job without stopping, until such time that he has passed the earliest possible lunch time. At that time the employee will have an hour's lunch after which the Field Force Automation Server will be contacted for the next job.

These rules, together with the parameters as they are in *Table 4-5*, were used to generate the Call Frequency Model. This resulted in *Figure 4-5*, which represents the number of new incoming transaction calls per second (frequency) that the server can expect to receive during a normal working day.

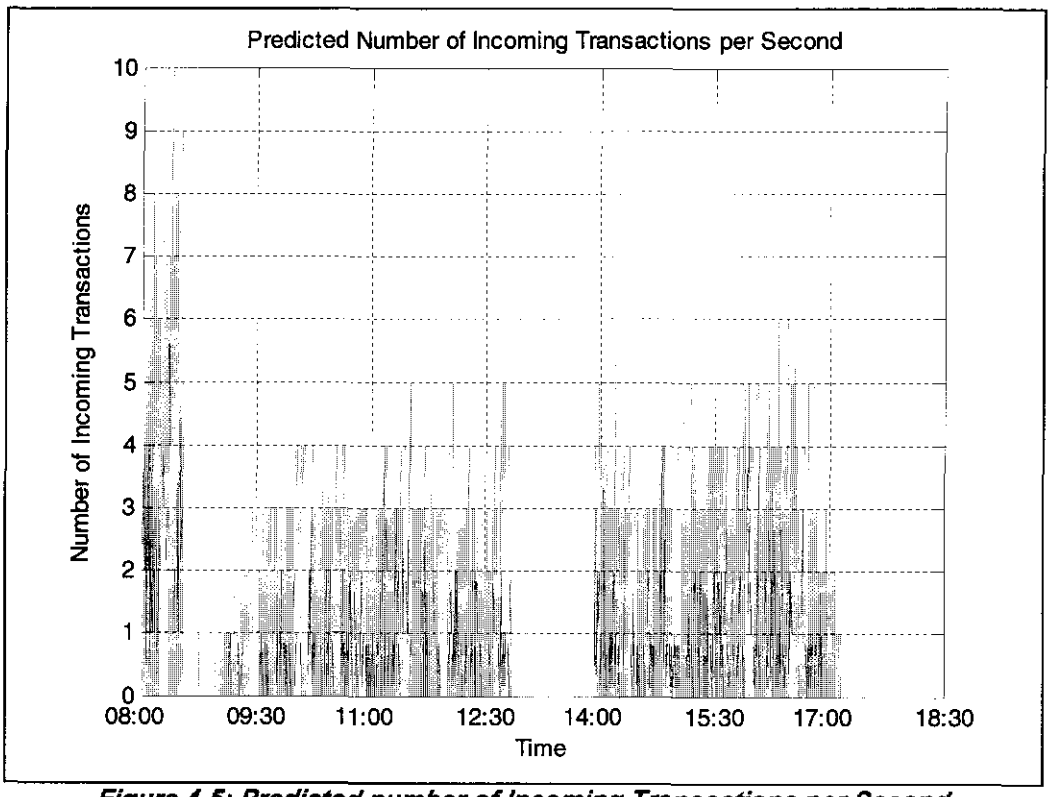

Figure 4-5: Predicted number of Incoming Transactions per Second

The occurrence of some calls after 17:00 in the afternoon is due to the rule that the model takes into account a minimum of 8 working hours per day per person. If the person only started 08:30 and finishes a job at 17:01 then another job is given to that person.

The frequency of incoming transactions were then used to generate a profile of the number of personnel online during any given time of day in the case where a transaction takes 240 seconds to complete. This showed a surprising result in that up to 18% of the workforce will be on-line simultaneously during the mornings. A second, smaller peak is present just after the lunch hour, as can be expected.

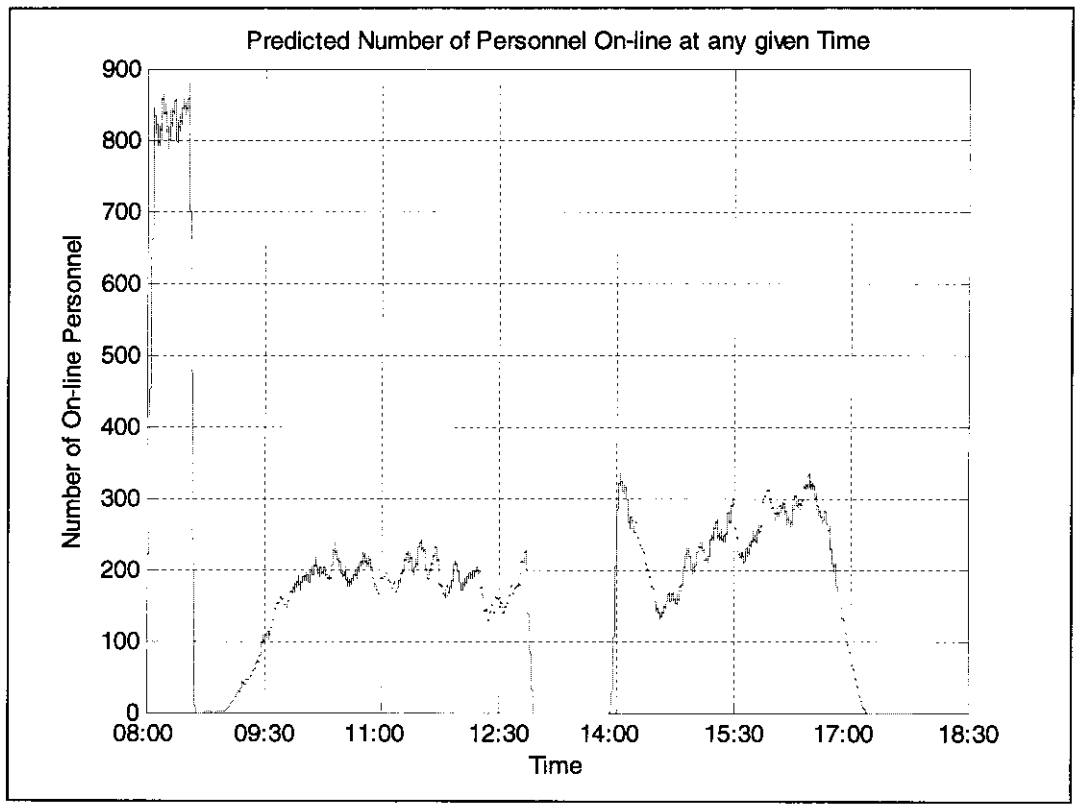

Figure 4-6: Predicted number of Daily On-line Personnel

To test the scalability of the model the number of personnel was first decreased to a small workforce of 1,000 and then increased to a large workforce of 15,000. The only real difference this seemed to have on the model was a change in the variance of the profile. The small workforce generated a profile similar to that of *Figure 4-6* with a little more jitter due to random variations in the different trial runs, while the large workforce gave a smoother profile.

4.6 CONCLUSION

There are two interesting characteristics that can be noted from the model results. The peak caused during the morning session when all the personnel start working causes a peak load of 18% of the workforce on-line at once. When compared to the rest of the day, there are no peaks at more than 7%.

This in itself is an important result, which can help when planning bandwidth requirements for new systems.

Another point of interest is the processor load of the Field Force Automation server. Given the fact that Field Force Automation transactions are mostly database based transactions, they can become quite processor intensive. Keeping in mind the 18% workforce peak, these results could also help specify the processor requirements for such a server.

More insights like these, which can assist with system planning, can be gained by using the models developed in this study and using them in the simulation of Field Force Automation systems. The results of this chapter will be discussed further in the conclusion of this dissertation.

5 Chapter 5 – Simulation Setup and Results

This chapter addresses the second part of the project, namely the simulation of Field Force Automation traffic over a GPRS network, while implementing different queuing algorithms. The approach followed as well as results achieved will be discussed.

5.1 INTRODUCTION

There can be several reasons for simulating a network, some of which include:

- Planning a new network.
- Testing the response of a network to certain changes.
- Testing the sensitivity of the network to several network problems.
- Network capacity analysis.
- Testing to see the effect of certain application traffic on the network.
- Performing network optimisations.

Network optimisations can involve the tweaking of QoS parameters. One of these parameters is the packet scheduling/queuing algorithm used. Delay optimisation of a Field Force Automation system by using different queuing algorithms will be viewed in this chapter.

5.2 PROBLEM STATEMENT

In the previous chapter, a Data Source Model for Field Force Automation systems was developed. The usefulness of this model for companies planning new Field Force Automation implementations, as well as companies who already have Field Force Automation systems can, however, only be illustrated by implementing it in simulation software and using it in an appropriate network simulation.

5.3 GOALS

To make this illustration applicable to real world scenarios, two outcomes were selected based on information that would be useful to both new system implementations as well as existing systems:

1. Network Capacity Planning – The goal of this is to provide network planners with some kind of a guideline for the required bandwidth in relation to their workforce size.
2. Network Resource Optimisation – To offer a good value-for-money solution to Field Force Automation system owners, a way should be found to increase the capacity and efficiency of their network, without significant upgrade costs or any reconfiguration on the field units, software or mode of operations.

5.4 PREVIOUS WORK

Previous work on this topic includes the following:

- A number of studies on performance analysis of GPRS based networks, which mostly used standard World Wide Web (WWW) Data Source Models for their simulations. These studies can be found in [20]-[24].
- Two studies involving the comparison between different scheduling algorithms were used as guideline for the procedures to simulate queuing algorithms. The studies did, however, only focus in the internal scheduling algorithms used within the GPRS network and not on the customer equipment [29],[30].
- The Engineering School of Science at the Simon Fraser University in Canada provided a model of how a GPRS network is built in OPNET. This network could be used as a starting point but had to be expanded due to the large scale of the Field Force Automation simulation [31].

No relevant work on Field Force Automation systems could, however, be found. For this reason it was decided to use the procedures used in the studies mentioned above concerning WWW traffic on GPRS networks and adjust their application to Field Force Automation systems.

5.5 SIMULATION SOFTWARE SELECTION

After reviewing a number of articles about GPRS Network Simulations and QoS Simulations (as stated in the previous section), the two network simulators that were most widely used in these publications and, therefore, academically most accepted, were NS-2 and OPNET.

5.5.1 NS-2 [32]

NS-2 is an object orientated event driven network simulator and is mostly used for small-scale simulations of queuing algorithms, transport protocol congestion and multicast related work.

5.5.1.1 Package Attributes

NS implements key protocol modules for unicast and multicast routing, reservation, transport and session protocols. NS-2 is written in C++ and provides a compiled class hierarchy of objects written in C++ and an interpreted class hierarchy of objects written on Otcl.

5.5.1.2 Simulator Implementation

Controlling, configuring and operating of the simulations are provided through the Otcl class simulator. The class provides the procedure to manage and create the desired network topology as well as to initialize the network and to choose the desired scheduler. The topology is created by the user through the use of Otcl standalone classes, links and nodes that provide simple primitives. The function of the nodes is to receive a packet, examine it and to map it to the relevant outgoing interface. These nodes are composed of classifier objects, with each classifier concentrating on a certain part of the packet. The end points of the network, where packets are received or transmitted, are modelled by agents.

NS links are characterized in terms of delay and link bandwidth. They are built from a sequence of connector's objects. The data structure representing a link is composed of a queue of connector objects, its head and type of link, time to live and an object that processes link drops. Connectors receive the packets, perform a certain function and then transmit the packet to the next connector or to the drop object. Various kinds of links are supported, e.g. point-to-point, broadcast and wireless.

5.5.1.3 Simulation Engine

NS2's simulation engine is a single thread engine and is extensible, programmable and configurable. Events within NS are described by a firing time and handler function. The type of event scheduler used to drive the simulation can be chosen among the four presently available event schedulers: a simple linked-list (default), heap, calendar queue and a special type called real-time. Each one is implemented using a different data structure.

The simple linked-list scheduler provides a list of events kept in time-order from the earliest to the latest. This requires scanning the list to find the appropriate entry upon insertion or deletion. The entry at the head is always executed first. Entries with the same simulated time are extracted according to their order in the list. The heap scheduler code is borrowed from the NETSIM simulator. In the calendar queue scheduler implementation, events with the same "month/day" of multiples "year" are recorded in one "day". The real-time scheduler is still under development and is a subclass of the list scheduler. It is well suited when events arrive with a relatively slow rate.

5.5.1.4 Supplier Statement

The suppliers of NS-2 state the following about the product on their website: "*While we have considerable confidence in NS, it is not a polished and*

finished product, but the result of an on-going effort of research and development. In particular, bugs in the software are still being discovered and corrected. Users of NS are responsible for verifying for themselves that their simulations are not invalidated by bugs.

We are working to help the user with this by significantly expanding and automating the validation tests and demos. Similarly, users are responsible for verifying for themselves that their simulations are not invalidated because the model implemented in the simulator is not the model that they were expecting. The ongoing NS Manual should help in this process."

5.5.1.5 Benefits

NS-2 has the following benefits:

- No licence is required to operate for any application.

5.5.1.6 Drawbacks

NS-2 has the following drawbacks:

- The software still contains bugs.
- Only operates on Linux.
- No built-in results analyser.
- Lack of pre-built models.

5.5.1.7 Summary

To summarise, NS-2 is a freely available simulation package, specifically for Linux users, that can simulate different types of communication networks. NS-2 is used by a number of universities in the world and can be found in a number of studies concerning network simulations. NS-2 is a popular choice for low-cost, non-commercial simulations.

5.5.2 OPNET MODELER 10.5 [32][34]

OPNET Modeller is a commercial network simulator distributed by OPNET Technologies in the United States of America.

5.5.2.1 Package Attributes

OPNET can be subdivided into a few parts, which are the OPNET Modeller, OPNET Planner, Model Library and the Analysis Tool. Some of the features included in the generic simulator are an event-driven scheduled simulation kernel, integrated analysis tools for interpreting and synthesizing output data, graphical specification of models and a hierarchical object-based modeller.

5.5.2.2 Simulator Implementation

The OPNET modeller is intended for modelling, simulating and analysing a large communication network. The modelling technology of OPNET is organized in a hierarchial structure. At its lowest level, process models are structured as finite state machines. State and transitions are specified graphically using state-transition diagrams, whereas conditions that specify what happens within each state are programmed with C++. Those processes, and built-in models in OPNET (source and destination modules, traffic generators and queues, for example), are then configured with menus and organised into data flow diagrams that represent nodes using the graphical editor. Using a graphical network editor, nodes and links are selected to build up the topology of a communication network.

5.5.2.3 Simulation Engine

The analysis tool provides a graphical environment to view and manipulate data collected during simulation runs. Results can be analyzed for any network element. OPNET planner is an application that allows administrators to evaluate the performance of communications networks and distributed systems, without programming or compiling. The modelling libraries are

included in the OPNET planner and modeller and contain protocols and analysis environments.

5.5.2.4 Supplier Statement

The following statement was taken from the official OPNET Modeller product website: "*OPNET Modeller is the world's most powerful modelling and simulation platform, essential for design and analysis of networks, network equipment, and communications protocols. OPNET Modeller provides engineers with a network technology development environment that boosts R&D productivity, improves product quality, and reduces time to market.*"

5.5.2.5 Academical Backing

OPNET Modeller is a commercial environment for network modelling and simulation, allowing one to design and study communication networks, devices, protocols, and applications. Modeller is used by some of the world's largest technology organizations, such as Cisco, Intel, 3Com and Nokia to assist their research and development processes. OPNET Modeller is also used by well-known universities in the United States of America, including [35]:

- MIT
- Stanford
- University of California (Berkeley)
- University of Illinois
- Georgia Institute of Technology.

OPNET also hosts an international conference called OPNETWORK every year for researchers who use OPNET for research purposes. The OPNETWORK '04 conference had attendance of more than 1500 people representing 39 countries [36]. This shows clearly that OPNET is highly trusted among a wide range of international peers.

5.5.2.6 Benefits

OPNET has the following benefits:

- Relatively easy to learn with on-line support and thorough documentation.
- Operates on Windows NT, 2000, XP and UNIX
- Free for academic use.
- Includes specialized editors, analysis tools, and off-the-shelf models, which gives one time to focus on the unique parts of one's project.

5.5.2.7 Drawbacks

OPNET has the following drawbacks:

- Licence required for commercial use.

5.5.2.8 Key Features

Originally developed at MIT and introduced in 1987 as the first commercial network simulator, OPNET Modeller has many great features. The folowing were relevant to this project:

- Scalable and efficient simulation engine, enabling the fast simulation runtimes using acceleration techniques for wired and wireless models.
- Object-oriented modelling - Nodes and protocols are modelled as classes with inheritance and specialization.
- Clear and simple modelling paradigm - Model the behaviour of individual objects at the "Process Level" and interconnect them to form devices at the "Node Level" interconnect devices using links to form networks at the "Network Level". Organize multiple network scenarios into "Projects" to compare designs.
- Wireless, point-to-point and multipoint links - link behaviour is open and programmable. Accurately account for delay, availability, bit errors and throughput characteristics of links. Incorporate physical layer characteristics and environmental effects using a library that includes

enhanced Longley-Rice and Free Space or interface with custom propagation models.

- Integrated debugger - Validate simulation behaviour or track down problems.
- Integrated analysis tools - Tools to display simulation results. Plot and analyze time series, histograms, probability functions, parametric curves and confidence intervals. Export to spreadsheets or use XML.
- Library of protocol and application models - Including Multi-Tier Applications, Voice, HTTP, TCP, IP, Frame Relay, Ethernet, ATM, 802.11 Wireless LANs, IP Multicast, Circuit Switch, Mobile IP, etc.
- Geographical and mobility modelling - Model cellular, mobile *ad hoc*, wireless LAN and satellite networks or any network with mobile nodes. Control each node's position dynamically or pre-define trajectories. Add maps and other background graphics for context and visual enhancement.
- Models can be shared across hardware and platforms transparently without modification.

5.5.2.9 Summary

To summarise, OPNET is an industry standard simulation package that can simulate most types of communication networks. OPNET has an optimised simulation kernel for faster than real-time simulations.

5.5.3 SOFTWARE CHOICE

After reviewing both software options, OPNET Modeller was chosen for simulating the Field Force Automation network. The three main reasons for this are:

- OPNET is used by well respected peer universities for research purposes and is available at a reduced price for academic use.

- OPNET is easier to learn than NS-2 as it comes with support and comprehensive documentation.
- OPNET can be used on Windows XP.

5.6 DATA SOURCE MODEL IMPLEMENTATION

The first part of the simulation with OPNET included the implementation of the data source model. To be able to do this, one first has to understand how OPNET handles data source models and how to implement a custom built model. Keep in mind that the model was developed before the actual simulation software was selected. The compatibility of the model developed with the simulation software will, therefore, also give a good indication of how well the model was developed and whether or not it succeeded in its goal of being suitable for GPRS network simulations.

5.6.1 OPNET APPLICATION MODELS

The way in which OPNET handles data source models is in the form of application models. There are already a number of predefined models available for use such as: WWW, FTP, Email, Database, etc. In order to implement one's own data source model, the custom application model should be used.

Figure 5-1 shows the breakdown of how OPNET sees such an application model. The first term used is that of a task. A task is a complete client-server interaction, like checking one's e-mail, or a performing a Field Force Automation transaction.

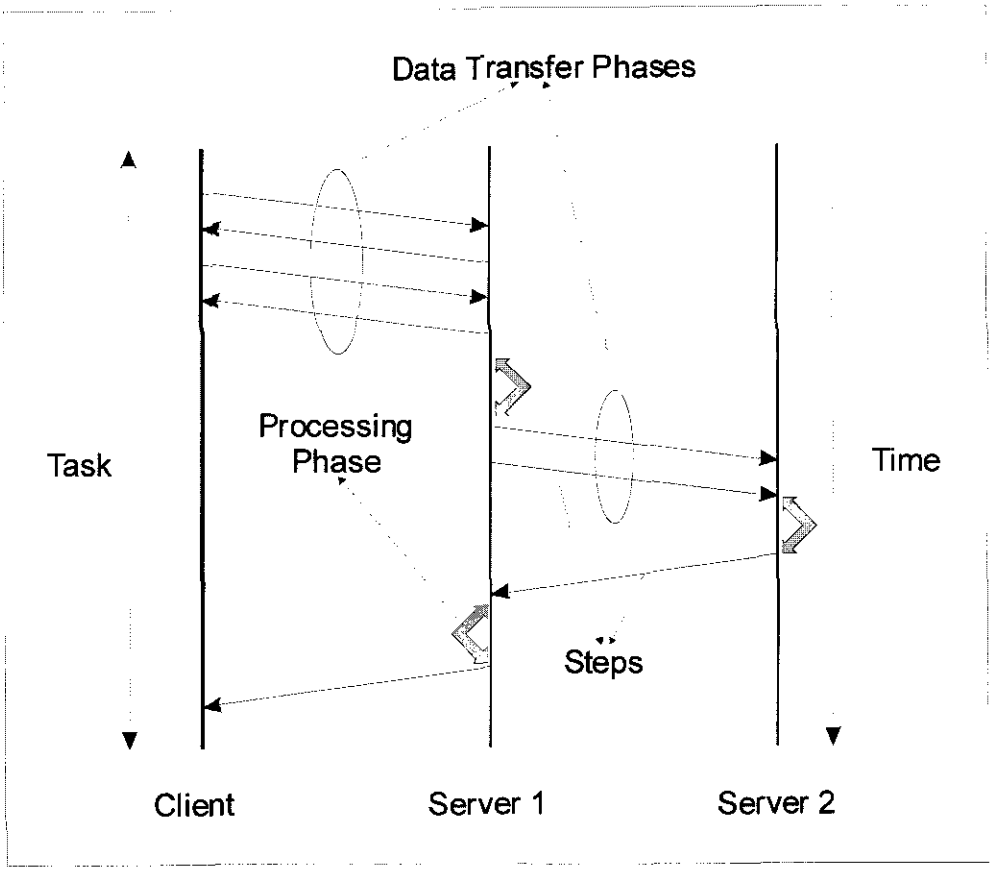

Figure 5-1: OPNET Traffic Model Operation [37]

The task is divided into several different phases. In the case of the E-Mail task, the phases could be:

1. Connect to E-Mail Server
2. Retrieve E-Mails
3. Send E-Mails
4. Disconnect.

Each of these phases is again divided into several steps. As an example use will be made of the "*Connect to E-Mail Server*" phase. This can in turn be divided into the following steps:

1. Send Connection Request to E-Mail server
2. Receive Connection Confirmation and Authentication request from E-Mail server
3. Send Username and Password to E-Mail Server
4. Receive Authentication Confirmation from E-Mail Server.

Icon	Name	Description
	Task Definition	The Task Definition is a collection of a number of phases that make up a certain task. Example: a Login Task will consist of phases: Request Connection, Receive Login Page, Submit Login Information, Receive Confirmation.
	Application Definition	The Application Definition combines a number of tasks to make up an application. Example: a Field Force Automation Application will consist of the following phases: Login, Send Job Report, Get New Job Info, Logout
	Profile Definition	The Profile Definition is used to define which applications are used by a certain network node. In the case of a Field Force Automation unit, the User Profile will just consist of the FFA Application, but Internet browsing, E-Mail and FTP access can also be added.

Table 5-1: Data Source Model components.

In order to define these steps, phases and tasks into an application, certain components have to be included in the simulation project. These components are shown in *Table 5-1* and include the following:

1. Task Definition – The Task Definition component contains all the information about the different tasks involved in the application. When specifying the steps, OPNET requires the following information: Source, Destination, Delay (with statistical variation), Data Transmitted (with statistical variation) and Data Received (with statistical variation). The different phases, as they were entered into the Task Definition, are shown in *Table 5-2*. Note how it matched the model parameters developed in *Table 4-4*. For the data transferred in each step, consult *Table 4-4*.
2. Application Definition – The Application Definition component defines the tasks performed by a certain application. In the case of the Field Force Automation Application, all the tasks are included in the application as well.
3. Profile Definition – The Profile Definition is used to specify which applications are used by which users. It is possible to assign certain

users with internet browsing and e-mail applications while other users use database access and FTP transfers. In the case of the Field Force Automation profile, only the Field Force Automation application was included in the profile, as that is the only traffic that will be travelling between the Field Force Automation Units and Server.

Phase	Task	Phase Type	From	To
Connect Delay	Connect To Home	Delay	FFA Unit	
Home Page Request	Connect To Home	Data Transfer	FFA Unit	→ FFA Server
Process Page	Connect To Home	Delay	FFA Server	
Login Page Reply	Login	Data Transfer	FFA Server	→ FFA Unit
User Login Time	Login	Delay	FFA Unit	
Login Data Sent	Login	Data Transfer	FFA Unit	→ FFA Server
Process Login	Login	Delay	FFA Server	
Reply Job Report Page	Job Report	Data Transfer	FFA Server	→ FFA Unit
Type Job Report	Job Report	Delay	FFA Unit	
Submit Job Report	Job Report	Data Transfer	FFA Unit	→ FFA Server
Process Job Report	Job Report	Delay	FFA Server	
Send New Job Info	New Job	Data Transfer	FFA Server	→ FFA Unit
Read Job Data	New Job	Delay	FFA Unit	
Request Logout	Logout	Data Transfer	FFA Unit	→ FFA Server
Process Request	Logout	Delay	FFA Server	
Confirm Logout	Logout	Data Transfer	FFA Server	→ FFA Unit

Table 5-2: Table of Data Source Model Phases and Tasks

When building the simulation network, each of the Mobile Stations (Cellular Phones) will be assigned a Profile Definition to tell OPNET that a specific unit will be running Field Force Automation transactions. The Server component will also be configured to support Field Force Automation transactions. In that way OPNET knows that the Field Force Automation traffic travels between the Mobile Stations and the Server that supports Field Force Automation transactions.

When all the steps, phases and tasks have been configured, the Custom Application Model is ready to be used in simulations.

5.6.2 TRAFFIC MODEL IMPLEMENTATION VERIFICATION

Before using the implemented Field Force Automation data source model in a full scale GPRS network simulation, the implementation of the data source model has to be tested first. To do this, a simple client-server network was constructed, using a Server node and connecting it to a Desktop Client node using a 10baseT network cable (See *Figure 5-2*).

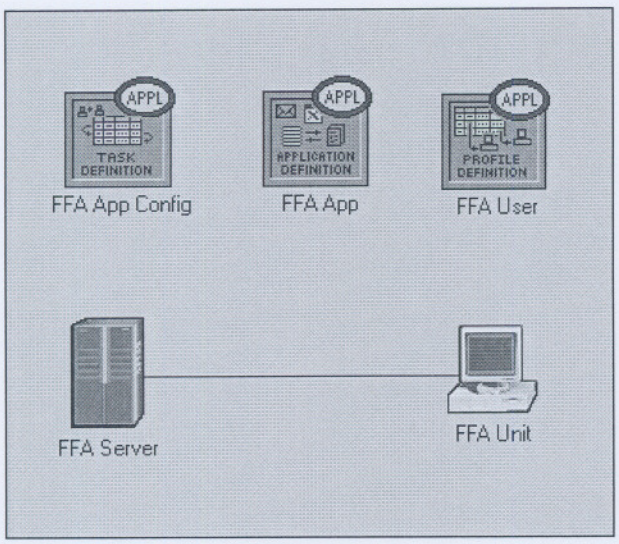

Figure 5-2: FFA Transaction Model Test Network

A single transaction was then simulated between these two nodes and the traffic was monitored. The result of this simulation is shown in *Figure 5-3*. When compared to the model developed in *Figure 4-3*, it is clear that the model was correctly implemented as the two figures match closely. Note that *Figure 4-3* is specified in bytes, while the output from OPNET in *Figure 5-3* is in bits.

The Field Force Automation data source model was, therefore, correctly implemented in the OPNET environment and is ready for use in larger simulations.

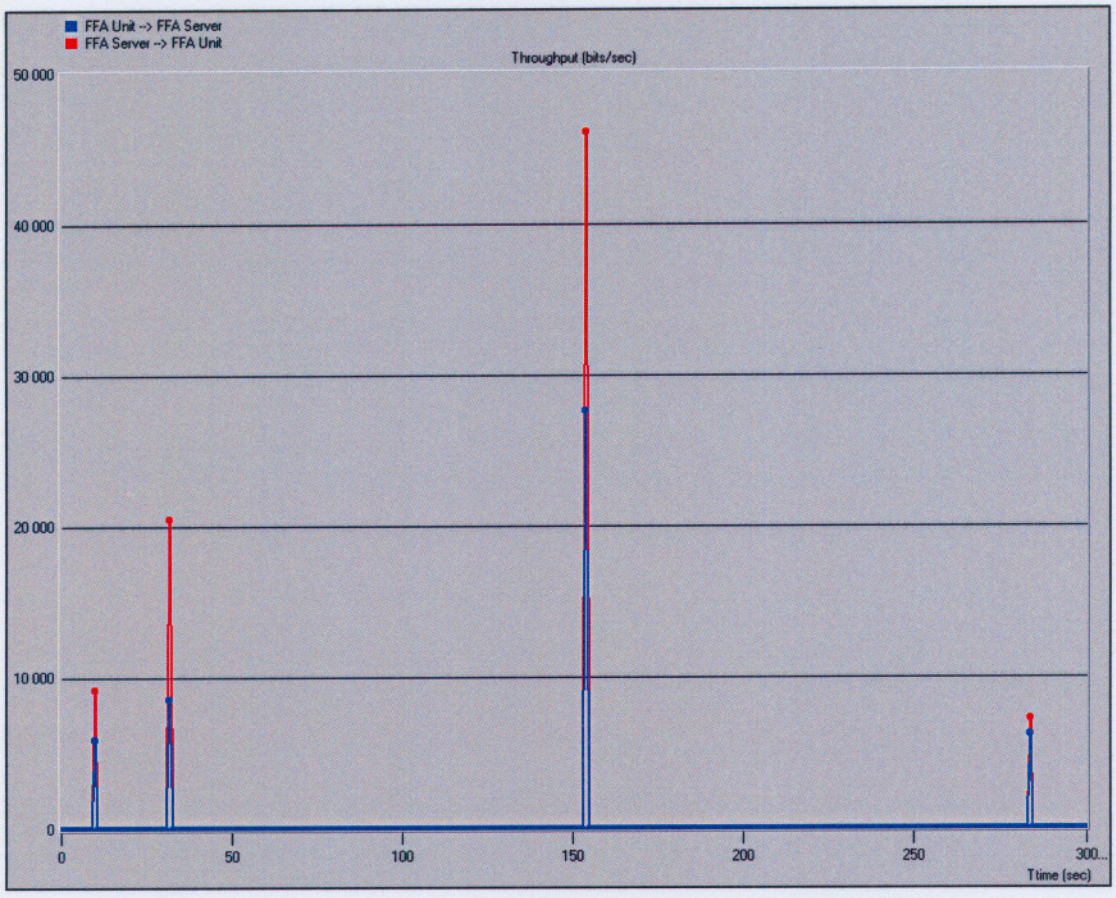

Figure 5-3: Single FFA Transaction as Simulated in OPNET

5.7 SIMULATION NETWORK COMPONENTS

The next step in the simulation part of the project is to construct a GPRS network using OPNET. To do this, the GPRS network in [31] was used as a baseline network. The components used to construct the network can be seen in *Table 5-3*. These include most of the same components as first described in Chapter 2.5, which are:

1. Mobile Stations (MS) – The cellular phone, or PDA used by the field technicians.
2. Base Transceiver Stations (BTS) – The cellular phone tower, to which the mobile station connects using radio signals.
3. Base Station Controllers (BSC) – The node responsible for managing the calls and data from and to the different BTSs.

4. Servicing GPRS Support Nodes (SGSN) – The node that is responsible for routing the necessary traffic between its GGSN and a number of BSCs.
5. Gateway GPRS Support Node (GGSN) – The node responsible for routing the traffic between the external packet data network and the SGSNs.

Icon	Name	Description
	Mobile Station	These units represent the GPRS and Java enabled cellular phones that the technicians carry with them from location to location. These units form the one end of the Field Force Automation system.
	Base Transceiver Station	These units represent the Base Transceiver Stations, or the physical cell phone towers.
	Base Station Controller	The Base Station Controllers control a number of BTSs.
	Servicing GPRS Support Node	The Servicing GPRS Support Node connects to a number of Base Station Controllers and route the necessary traffic to each of them.
	Gateway GPRS Support Node	The Gateway GPRS Support Node connects the cellular network to the internet or an outside network. On the one side it connects to a packet data network and on the other side to a number of SGSNs and it routes the packets to the correct SGSNs.
	Router	The Router is used to connect the GGSN to the Field Force Automation Server.
	Field Force Automation Server	The Field Force Automation Server forms the other end of the Field Force Automation system, to form an End-to-End connection between the FFA Units and the FFA Server. The FFA Server provides the information required by the FFA personnel.

Table 5-3: Table of Simulation Network Components

To complete the network, two extra components had to be added:

1. Router – The network node responsible for interfacing the GGSN with the Server.
2. Server – The computer responsible for communications to all the MS's on the network.

These components, together with the interconnecting cables, are all that are required to build a GPRS simulation network.

5.8 SIMULATION NETWORK SETUP

Now that the components have been selected, they have to be placed, connected and configured. An example of a simple GPRS network can be found in [31]. This network has to be expanded to a larger scale in order to simulate a Field Force Automation system.

The first issue that has to be addressed is the number of mobile stations. In order to prevent slow simulation times, the number of field units has to be kept as low as possible. Fortunately the simulation will approach the problem from another side, it will start with one transaction and gradually increase the number of transactions until the network is congested. When the network is congested to the point where it starts reducing productivity, the number of active transactions at that point will be used to determine the maximum number of employees for the specific bandwidth.

To start with, an educated guess would be sufficient. Start with the bandwidth available. Start with 512kbps of bandwidth; this should be enough to support a relatively large workforce. Given the fact that the average transaction exchanges about 15KB of data, it works out to a theoretical maximum of 300

simultaneous transactions. In practice this number could, however, be far less.

It is possible for each mobile station to run multiple non-synchronised transactions, in which case the number of mobile stations can be further reduced without any further effect on the network. When looking at the Field Force Automation data source model and its bursty nature, it is possible to run 6 non-synchronised transactions, without the transactions impacting on each other.

This means that the number of mobile stations can be set to 50 and that would still allow for a simulation of up to 300 transactions. The number of Base Transceiver Stations was then chosen to have an average of 2 mobile stations per BTS. The BTS:BSC ratio was kept relatively standard at 4:1, while there is basically one SGSN for every BSC. All the SGSNs in turn connect to a single GGSN which is connected to a Router. The router in turn connects to the Server with a connection limited to 512kbps. *Table 5-4* shows the components used in the GPRS network, as well as the number used for each.

Up to the Router, the connections between the nodes were specified as high bandwidh connections (10Mbps or more) to make sure that congestion does not occur within the GPRS network itself. The reason for this is that the primary goal of this simulation is to show the effect of scheduling algorithms on an over-utilised GGSN-Server link. Seeing as the system developer cannot change queuing algorithms on the cellular network itself, the only nodes that can be configured are the server and the mobile stations.

Component	Name	Node Count
MS	Mobile Station	50
BTS	Base Tranceiver Station	24
BSC	Base Station Controller	6
SGSN	Servicing GPRS Support Node	5
GGSN	Gateway GPRS Support Node	1
Router	Router	1
FFA Server	Field Force Automation Server	1

Table 5-4: Node Count for each Network Component

The final result can be seen in *Figure 5-4*, which shows all the network components and how they are interconnected. This is then also the network which will be used to simulate the Field Force Automation system. Each of the mobile stations as well as the server are configured to use the Field Force Automation profile, which in turn is configured to perform the necessary steps used in a Field Force Automation transaction.

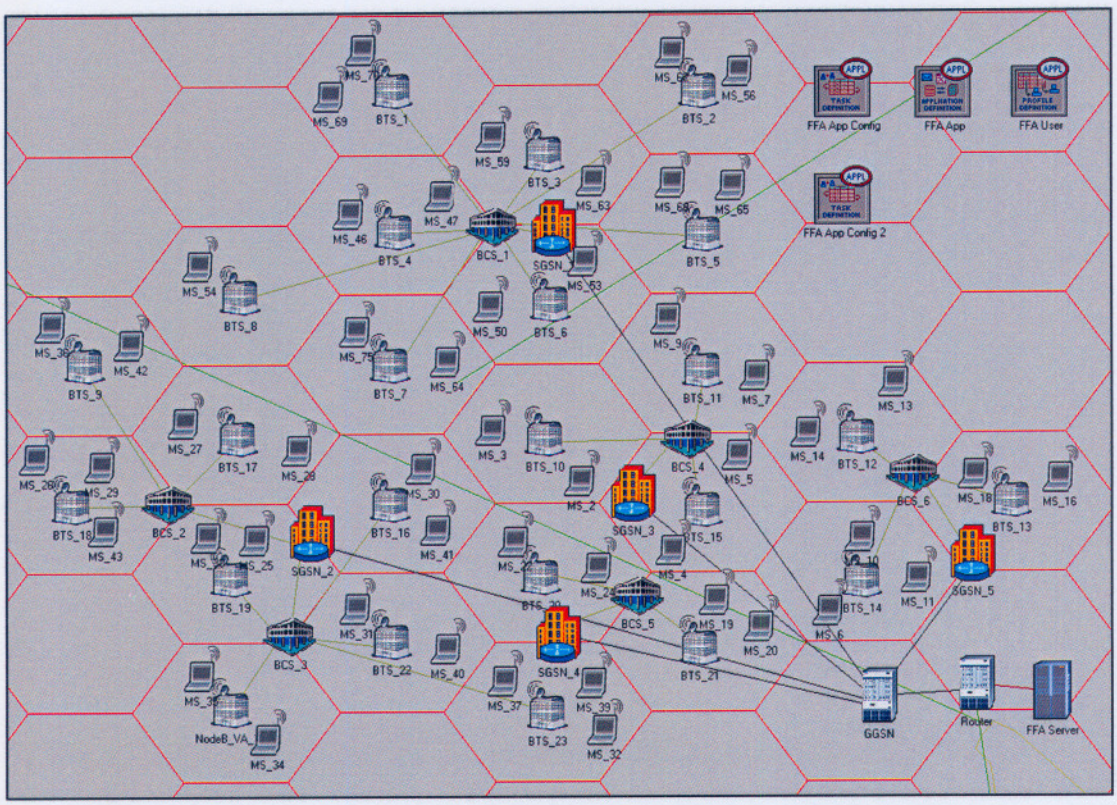

Figure 5-4: GPRS Simulation Network

Another issue which has to be addressed it that of the queue which will be scheduled. For this simulation, the outgoing server queue was selected for the following reasons:

1. The outgoing server queue is the only one that will become congested, as the incoming network traffic is less than half that of the outgoing

traffic, which will cause congestion on the outgoing queue of a symmetrical link.

2. The outgoing server queue is also the most important queue, as this one queue will influence the delay experienced by all the field units.
3. Trying to change the queue on the mobile stations themselves will involve quite complicated software development as well as a recall of all the field units. This will cost money and the original goal of this project was to provide a guideline for optimising a link's usage, without incurring extra costs.

5.9 QUEUING ALGORITHM SELECTION

The final step in the setup of the GPRS Simulation network was the choice of queuing algorithms to implement. Seeing as the goal of the simulation is only to see if different queuing algorithms would have an effect on the network delay, it was not necessary to implement all the queuing algorithms. This would have been time consuming, so a representative selection was made. The following is a list of all the queuing algorithms described in Chapter 3, together with reasons for being chosen or left out:

1. FIFO (First In First Out) – Included – This is the most basic of queuing algorithms and also widely implemented. FIFO should serve as a good baseline performance algorithm.
2. PQ (Priority Queuing) – Not Included – The Field Force Automation traffic is not divided into different priorities which means that PQ will in the end just become FIFO as all the traffic is added to one queue.
3. FQ (Fair Queuing) – Not Included – FQ is the base class for WFQ, which in turn is an upgrade to FQ. The better one of the two will, therefore, be included.
4. WFQ (Weighted Fair Queuing) – Included – WFQ is one of the most popular queuing algorithms. It is part of the FQ class of algorithms, so it was chosen to represent Fair Queuing algorithms.

5. WRR (Weighted Round Robin) – Not Included – Once again WRR is a base algorithm for the WRR class of algorithms. Upgrades DWRR and MDRR are available and will be chosen instead.
6. DWRR (Deficit Weighted Round Robin) – Included – Upgrade to the WRR algorithm, but still part of the Round Robin range of algorithms, chosen to represent this range.
7. MDRR (Modified Deficit Round Robin) – Included - This algorithm has been slightly adapted from DWRR and is also implemented in certain Cisco routers. For interest sake this algorithm will also be included to be compared with DWRR.
8. RED (Random Early Detection) – Included – RED is the most popular algorithm in a class known as "Congestion Avoidance Algorithms" and was then also chosen to represent that class of algorithms.

The simulation will, therefore, have the following queuing algorithms: FIFO, WFQ, DWRR, MDRR and RED.

5.10 SIMULATION RESULTS

The network was set up as discussed, equipped with the Field Force Automation data source model, together with the selected queuing algorithms. For each of the queuing algorithms, two simulation runs were conducted, taking the average of the results of the two runs. Each run was allowed to increase the number of concurrent transactions to a maximum of 300. The following results were obtained:

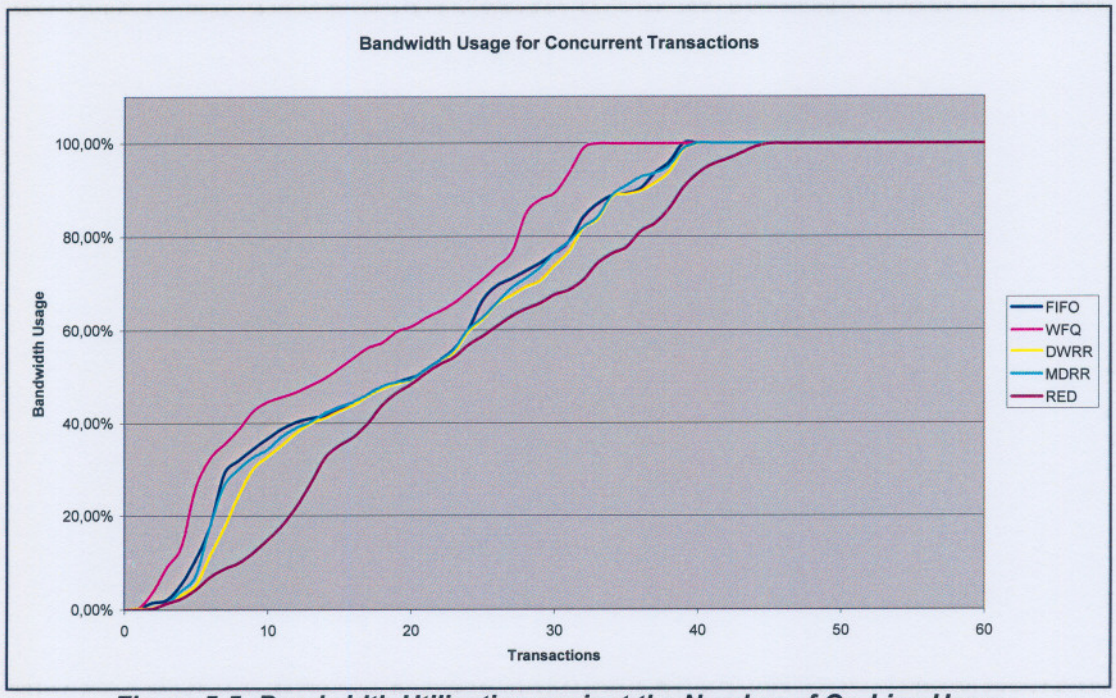

Figure 5-5: Bandwidth Utilisation against the Number of On-Line Users

Figure 5-5 shows the increase in bandwidth utilisation as the number of concurrent transactions increase. The effect of the different queuing algorithms can be clearly seen.

Algorithm	Maximum	Score
WFQ	32	100%
FIFO	38	119%
MDRR	39	122%
DWRR	40	125%
RED	46	144%

Table 5-5: Maximum Number of Concurrent Transactions

Table 5-5 shows the maximum number of concurrent transactions that can be performed without utilising 100% of the bandwidth available, using each individual queuing algorithm. There is also a normalised score to show that the RED algorithm can allow 44% more concurrent transactions than WFQ without using all the available bandwidth.

The best performing algorithm (RED) is clearly doing what it is designed to do, which is "congestion avoidance", as it is the algorithm which is most capable of avoiding the use of 100% of the bandwidth for as long as possible. The fact

that it is the best at avoiding 100% bandwidth use does not automatically mean it will have the smallest delay. The logic behind this statement is that if one has more transactions together, using the same amount of bandwidth, then one has less bandwidth per transaction and, therefore, probably a higher packet delay.

The next simulation recorded the average packet delay as the number of concurrent transactions increased. *Figure 5-6* shows the simulation for the entire range of up to 300 concurrent transactions, at which time the packet delay is ridiculously high at between 4500 and 5500ms seconds per packet. At this stage enormous amounts of packet loss will occur and the network will be almost unusable.

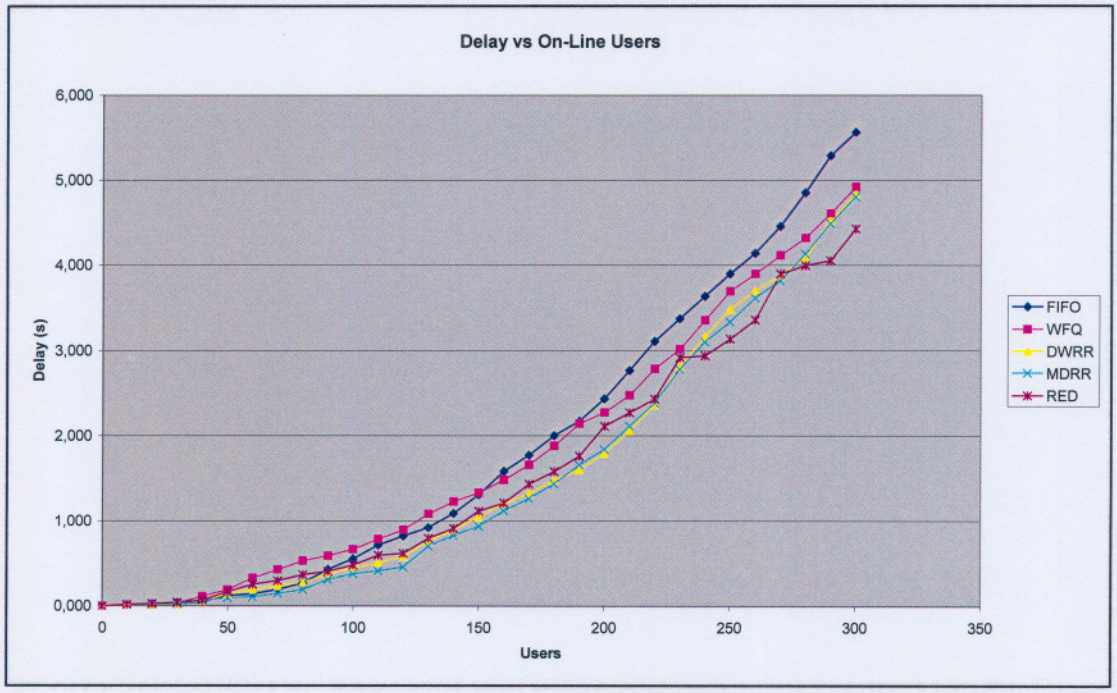

Figure 5-6: Queuing Delay for up to 300 Concurrent Transactions

The actual area of interest is where the delay is below 1000ms per packet. *Figure 5-7* shows a enlarged version of that part of the simulation results. It is interesting to note that the RED algorithm which performed the best in the congestion simulation, now only manages the middle of the field.

It is, however, surprising that the WFQ algorithm performed far worse than would be expected. Logic suggests that FIFO should be the worst algorithm, with the other algorithms increasing in performance. The reason this does not happen is not clear, but it is suspected to be due to the increased complexity of the algorithm, which takes extra time but doesn't really offer any benefit as all the traffic are of the same class. Another reason could be the large number of active queues that constantly have to be adapted as a new transaction is started. This probably causes the first queues to be serviced at ever increasing intervals, thereby increasing the average packet delay.

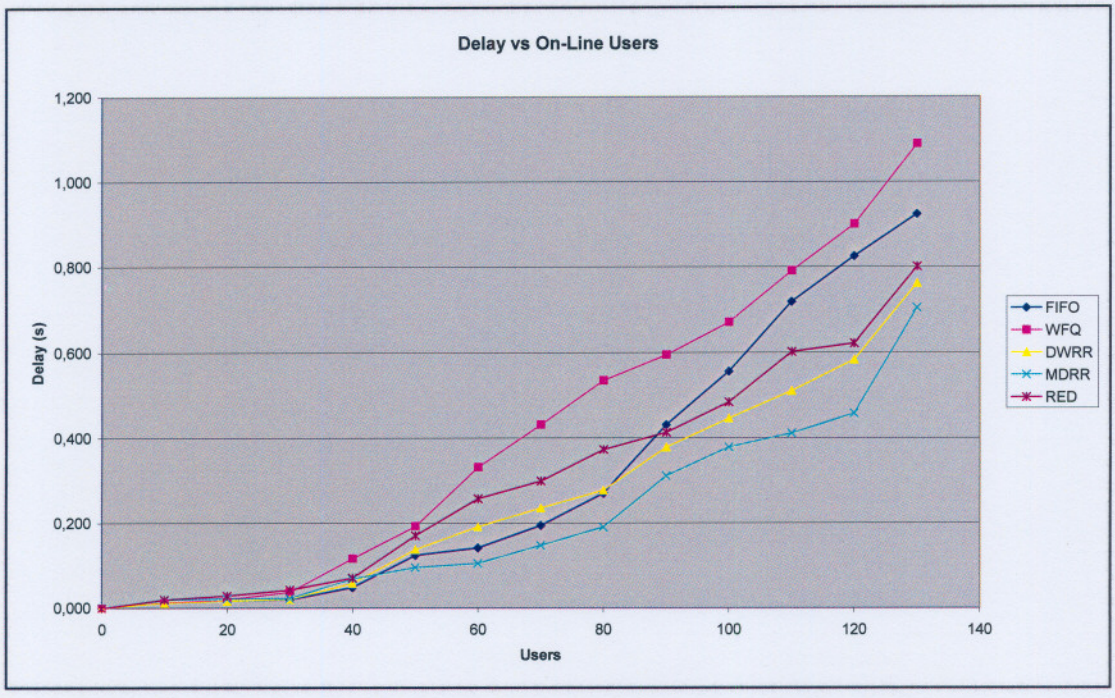

Figure 5-7: <1000ms Queuing Delay versus Server User Load

The results obtained are summarised in *Table 5-6*, where different packet delay thresholds are specified. The capacity of the network with each of these queuing algorithms is shown in order to give an idea of the performance increase that can be obtained by implementing a certain algorithm. It is important to note that between 0 and 100ms the different algorithms perform

relatively the same. Between 100ms and about 1000ms they diverge to give substantial differences in performance, after which they start converging, with the percentage performance gain available decreasing as the average packet delay increases.

Consider for example that a company stipulates in their SLA that the Field Force Automation system should always operate at a packet delay of 700ms or less. Given that the cellular network adds about 200ms of packet delay, the other 500ms can be queuing delays. If this company had implemented WFQ on their servers outgoing queue, they could realise a 58% increase in their maximum workforce size by implementing MDRR.

	250ms Delay		500ms Delay		1000ms Delay	
	Transactions	Score	Transactions	Score	Transactions	Score
FIFO	75	142%	93	122%	134	108%
WFQ	53	100%	76	100%	124	100%
DWRR	73	138%	106	139%	146	118%
MDRR	84	158%	120	158%	153	123%
RED	58	109%	100	132%	144	116%

Table 5-6: Queuing Algorithm Capacity for Different Delay Thresholds

According to these results, a general rule of thumb can be provided that for a similar Field Force Automation system, with a similar daily work pattern, one needs about 1kbps of bandwidth for every Field Force Automation unit out in the field. This should, however, be recalculated if the Field Force Automation system differs from the example system in any way.

5.11 CONCLUSION

The simulations were successfully conducted, providing some interesting and useful results. The initials goals set to provide a Network Capacity Planning guideline as well as assist with Network Resource Optimisation, have been achieved. For Network Capacity Planning a rule of thumb was provided and

the percentage increase in network capacity was also provided for Network Capacity Optimisations.

This could help Field Force Automation implementers to offer a good value-for-money solution to Field Force Automation owners, as well as to provide a low-cost way to extend the lifespan of current Field Force Automation systems.

The choice of queuing algorithms was simulated with results that show the best algorithms for this specific implementation of a Field Force Automation system.

6 Chapter 6 – Conclusions and Recommendations

In this chapter the results obtained in the dissertation will be viewed, discussed and summarised. Future work on this subject will be discussed briefly.

Reviewing this project, two major storylines can be identified. The first is the development of a data source model specifically targeted at Field Force Automation systems. The purpose of this model is to empower Field Force Automation system integrators to simulate such a network, thereby providing a helpful planning tool.

The second part of the project was directed at using this data source model in a GPRS network simulation. The first goal of this was to prove that the data source model can be used in such simulations and the second goal was to show that it can produce useful results, both for systems being planned, as well as systems already in operation.

6.1 SUMMARY OF TRAFFIC MODEL RESULTS

A number of surprising results were obtained during the development of the data source model. The most important part of the development was the example of how to develop a customised data source model for Field Force Automation. This example can be used by Field Force Automation system owners and developers to construct a data source model that closely matches their own implementation, thereby allowing them to simulate their system accurately.

There are also some aspects from the example data source model worth noting. The first is the marked difference in traffic volumes and behaviour between die WWW traffic model and the Field Force Automation model. This clearly showed that WWW traffic models are not suitable for Field Force Automation simulations.

The second aspect of interest is the peak in the number of concurrent transactions in morning rush, when all the personnel start working. This causes a peak load with 18% of the workforce on-line at once. When compared to the rest of the day, there are no peaks of more than 7%.

Depending on the system parameters, this means that if bandwidth was purchased to handle the maximum load which occurs in the first 10% of the day, the bandwidth would be more than 50% underutilised for the remaining 90% of the day.

The third aspect, or point of caution, is the processor load of the Field Force Automation server. Given the fact that Field Force Automation transactions are mostly database based transactions, they can become quite processor intensive. If each on-line user needs only 1% of the server's processing power, the server will only be able to handle 100 concurrent users. This is something to note for during the start of the day when the most people are on-line simultaneously.

The data source model was then used in GPRS based Field Force Automation network simulations.

6.2 SUMMARY OF QUEUING SIMULATION RESULTS

The results obtained during the network simulations showed that the data source model developed in the previous section can be used to generate useful results. The results obtained are useful for both system planners as well as system owners.

The first part of the simulation section showed that the data source model could easily be implemented in simulation software, without the model being developed specifically for the software. The model simulated a single transaction and produced results which matched those of the originally developed data source model.

The second part of the simulation section showed how the GPRS network was constructed and configured in order to be simulated as a Field Force

Automation network. The network was built using previously proved models and produced a network capable of simulating a wide range of situations.

The final part of the simulation section involved simulating Field Force Automation traffic over a throttled link between the cellular network and Field Force Automation server, while using different queuing algorithms. The results of these simulations showed that Field Force Automation systems in certain conditions can obtain up to a 58% increase in network capacity by implementing different scheduling algorithms.

It was also found that with a custom data source model and the correct simulation tools, the capacity specification required when planning a field force automation network can easily be calculated.

6.3 RECOMMENDATIONS

The results have made it possible to make several recommendations to system designers in order to optimise their systems. Certain areas of caution can also be highlighted. They are as follows:

- During the first part of the day, a peak occurs in which the network can become congested. Depending on the system parameters, this means that if bandwidth was purchased to handle the maximum load which occurs in the first 10% of the day, the bandwidth would be more than 50% underutilised for the remaining 90% of the day. This is not economical and can be largely overcome by, for example, allowing half the workforce to start at 08:00 and the other half at 08:30, which in effect cuts the morning peak in half. This can also extend the working day by 30 minutes as the people who start at 08:30 will then stop 30 minutes later.
- The load on the server processor during the morning peak also needs to be monitored. If the server starts slowing down due to processor overload, this can also be alleviated using the same approach as for

the previous problem, but other solutions like database caching and processor upgrade can also be considered.

- Implementing a queuing algorithm like MDRR can greatly increase the longevity of an existing Field Force Automation system.

6.4 FUTURE WORK

Future work in the area of Field Force Automation could include research on the following scenarios or topics:

6.4.1 DIFFERENTIATED CLASSES OF SERVICE

One scenario that can be investigated is that of differentiating traffic based on different classes of service. Most mobile operators these days have the capability of offering clients mobile access to their corporate Virtual Private Networks (VPNs). The Field Force Automation server can then be connected to the VPN which also has a connection to the mobile operators' network. The limiting factor in this case will be the connection to the mobile operators' network.

In this case the VPN will probably support different classes of service, such as Real Time, Business Interactive, Business Bulk and General Data. These classes are handled with different priorities by the queuing algorithms. Assigning different parts of each Field Force Automation transaction to different classes could help more important information reach the technician first in cases where congestion does occur.

The effect of these different classes of service in mobile access environments can be studied.

6.4.2 INTERFACE APPLICATION

The models in this document were developed for a system based on HTML transactions. These transactions could however make use of other application types, like a Java application front-end that interfaces with a database on the Field Force Automation server.

Other web technologies like PHP and XML could also behave differently and therefore generate different models compared to than of the HTML based system. A study that compares the behaviour of different interface applications for Field Force Automation systems would also be useful.

This study can also take into account the new generation of mobile technologies (EDGE, 3G, UMTS, WiFi, WiMAX, etc) and their capabilities and use that as guiding parameters to show which application type best suits which access technology.

6.4.3 QUEUING ALGORITHM PARAMETER ADJUSTMENTS

Another useful piece of work could include a study into the most widely used queuing algorithms and a qualitative study on the effect the adjustment of queuing parameters have on the performance of the different algorithms.

Another possibility for this is the adjustment of the queuing parameters for the study done in this document. The study has already proved that for the model developed here, the MDRR queuing algorithm is best suited and it can have a 58% increase in bandwidth efficiency to effect. This percentage might however still be increased by the adjustment of the MDRR parameters.

6.5 A FINAL WORD

This study has used guidelines from other studies together with a shortcoming in the tools used for network simulations to develop a model that can be used for Field Force Automation network simulations successfully. By following the same procedure, the model is fully scalable for different sized workforces, as well as different Field Force Automation platforms and situations.

The result of this is that performance modelling can now be done to determine the capability of the support/backbone networks to handle the traffic generated by such systems, as well as perform traffic optimisations for the fastest, most economical system.

This was proven when the developed data source model was used in a simulated network to show the effect of different queuing algorithms on Field Force Automation network traffic.

7 References

[1] C. Chiba, R. Volkwyn, "Field Force Automation Project", Telkom-Enhanced Services, 2003.

[2] B. Ghribi, L. Logrippo, "*Understanding GPRS: the GSM packet radio service", Computer Networks*, Vol. 34, p763, 2000

[3] J. Huang, S. Szu-Lin, C. Jiann-Horng, "*Design and performance analysis for data transmission in GSM/GPRS system with Voice Activity Detection*", IEEE Transactions on Vehicular Technology, Vol.51 Issue 4, p648, Jul. 2002.

[4] C. Lindemann, A. Thummler, "*Performance analysis of the general packet radio service",* Computer Networks, Vol. 41, p.1-17, May 2002

[5] T. Irnich, P. Stuckmann, "*Fluid-flow modelling of internet traffic in GSM/GPRS networks*", Computer Communications, IN PRESS, 2003, Available From: http://www.sciencedirect.com

[6] M. Ermel, T. Muller, J. Schuler, M. Schweigel, K. Begain, "*Performance of GSM networks with general packet radio services*", Performance Evaluation, Vol.48, no.1-4, p.285-310, May 2002.

[7] L. Hung-Huan, J.L.C. Wu,H. Wan-Chih, "*Delay analysis of integrated voice and data service for GPRS*", IEEE Communications Letters, vol.6, no.8, p.319-21., Aug. 2002

[8] ETSI Website – http://www.etsi.org

[9] GSM World Website, "*Today's GSM Platform*", Retrieved November 2003, http://www.gsmworld.com/technology/gsm.shtml

[10] Ericsson Articles, This is GSM, GPRS, and EDGE, 20 January 2004, http://www.ericsson.com/products/main/GGE_hpaoi.shtml

[11] Vodacom Website, Milestones section, Retrieved November 2003, http://www.vodacom.co.za/about/corporate_profile/vodacom_milestones.asp

[12] Brahim Ghribi, Luigi Logrippo, "*Understanding GPRS: the GSM packet radio service*", Computer Networks, Volume 34 (2000), Issue 5, November 2000, p763-779

[13] Vasiliki Emmanouil, "*GPRS Services and measurement of traffic performance*", MSc Thesis in Telecommunications, University College London, August 2003

[14] CISCO Internetworking Technology Handbook, Chapter 49 - Quality of Service, February 2003, http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/qos.html

[15] Chuck Semeria, "*Supporting Differentiated Service Classes: Queue Scheduling Disciplines – White Paper*", Juniper Networks, January 2002, http://www.juniper.net/solutions/literature/white_papers/200020.pdf - With special permission to use the entire document.

[16] CISCO – "*Understanding and Configuring MDRR and WRED on the Cisco 12000 Series Internet Router*", January 2004, http://www.cisco.com/warp/public/63/toc_18841.html

[17] Chuck Semeria, "*Supporting Differentiated Service Classes: Active Queue Memory Management – White Paper*", Juniper Networks, February 2002, http://www.juniper.net/solutions/literature/white_papers/200020.pdf

[18] Cisco IOS Release 12.0: "*Quality of Service Solutions Configuration Guide - Part 3: Congestion Avoidance*", October 2002, http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/qos_c/qcpart3/qcconavd.htm

[19] H. Graja, P. Perry, J. Murphey, "*Development of a Data Source Model for a GPRS Network Simulator*", Performance Engineering Laboratory, School for Electronic Engineering, Dublin City University, 2002.

[20] Rajiv Chakaravorty, Ian Pratt, "*WWW Performance over GPRS*", In Proceedings of IEEE International conference in Mobile and Wireless Communications Networks (IEEE MWCN 2002), September 9-11, Stockholm, Sweden

[21] C. Johansson, L. de Verdier, F. Khan, "*Performance of Different Scheduling Strategies in a Packet Radio System*", IEEE 1998 International Conference on Universal Personal Communications. Conference Proceedings, Vol. 1, p267-271, 1998

[22] W. de Sousa, F. Cavalcanti, Y. Silva, T. Maciel, "*System-Level Performance Evaluation of Different Scheduling Strategies in EGPRS*", GTEL-UFC, Wireless Telecom Research Group, Federal University of Ceara, Fortaleza, Brazil, 2002

[23] T. Irnich, P. Stuckmann, "*Analytical Performance Evaluation of Internet Access over GPRS and its Comparison with Simulation Results*", IEEE, Aachen University of Technology, Germany, 2002

[24] D. Staehle, K. Leibnitz, K. Tsipotis, "*QoS of Internet Access with GPRS*", University of Wurzburg, Germany, 2001, ISBN: 1-58113-378-2/01/07

[25] C. Courcoubetis, V. Siris, "*Procedures and tools for analysis of network traffic measures*", Performance Evaluation, Vol.48, p.5-23, 2002.

[26] ArcStream: "*Field Force Opportunities – White Paper*", May 2001, Retrieved From: http://www.arcstream.com/uploadfile/whitepapers/arcstream_wp_fieldservice.pdf

[27] S. Vardeman, "*Statistics for Engineering Problem Solving*", PWS Publishing Company, ISBN: 0-534-92871-4, 1994

[28] Tim Irnich and Peter Stuckmann, "*Fluid-flow modelling of internet traffic in GSM/GPRS networks*", Computer Communications, Volume 26, Issue 15, 22 September 2003, Pages 1756-1763

[29] Waltemar de Sousa, Francisco Cavalcanti, Yuri Silva, Tarcisio Maciel, "*System – Level Performance Evaluation of Different Scheduling Strategies in EGPRS*", Department of Telecommunication Engineering, Federal University of Ceara, Brazil, July 2001.

[30] Waltemar de Sousa, Francisco Cavalcanti, Yuri Silva, Tarcisio Maciel, "Capacity and QoS Enhancement for Data Transmission in EGPRS through Packet Scheduling and Smart Antennas

[31] R. Narayanan, P. Chan, M. Johansson, F. Zimmermann, and Lj. Trajković, "*Enhanced General Packet Radio Service OPNET Model*", School of Engineering Science, Simon Fraser University, Vancouver, BC, Canada, OPNETWORK Conference 2004, Retrieved From: http://www.ensc.sfu.ca/~ljilja/papers/opnetwork04_renju.pdf

[32] Lodewyk Swanepoel, "*Network Simulation for the Extraction of IP Network Statistics*", M.Eng Thesis, Potchefstroom University for Christian Higher Education, South Africa, November 2003

[33] The Network Simulator - ns-2 Official Website, September 2003, Retrieved From: http://www.isi.edu/nsnam/ns/

[34] OPNET Modeller Product Description Page, October 2003, Retrieved From: http://www.opnet.com/products/modeler/

[35] OPNET Modeller University Program, October 2003, Retrieved From: http://www.opnet.com/services/university/home.html

[36] OPNETWORK 2004 Conference, Press Release, October 2004, Retrieved From: http://www.opnet.com/news/press/opnetwork04_pr.html

[37] OPNET Modeller 10.5 Documentation, "*Standard Models User Guide – Applications Model User Guide – Custom Application*", March 2004