MOCASIM WORKBOOK

JE van Zyl · HA Kretzmann

WRC Report No. 1389/1/04

Water Research Commission

MOCASIM WORKBOOK

Report to the
WATER RESEARCH COMMISSION

by

J E van Zyl and H A Kretzmann
Rand Afrikaans University

WRC Report No 1389/1/04
ISBN No 1-77005-212-7

SEPTEMBER 2004

Disclaimer

This report emanates from a project financed by the Water Research Commission (WRC) and is approved for publication. Approval does not signify that the contents necessarily reflect the views and policies of the WRC or the members of the project steering committee, nor does mention of trade names or commercial products constitute endorsement or recommendation for use.

Printed by Silowa Printers: 012 804 1164

Three outputs emanated from this study:

1. This report, entitled "Mocasim Workbook";

2. The associated software package, Mocasim version 2.0. Mocasim II is available for download from the RAU Water Research Group's website at http://general.rau.ac.za/civil/wrg/mocasim.htm, or from the WRC website at http://www.wrc.org.za/wrcsoftware/mocasim.htm. It is recommended that the report and software be used together.

3. An object-oriented software library, called Ooten, for manipulating and simulating hydraulic networks using the public domain software package Epanet. Ooten is available for download from the RAU Water Group's website at http://general.rau.ac.za/civil/wrg/ooten.htm, or from the WRC website at http://www.wrc.org.za/wrcsoftware/ooten.htm.

EXECUTIVE SUMMARY

Background

Current design standards for bulk water supply systems do not allow much design flexibility. Firstly, they generally do not allow the designer to differentiate meaningfully between urban and rural systems, and secondly they do not allow the designer the freedom to assume different levels of reliability.

For these reasons, a methodology was deemed necessary to allow the designer to couple reliability with system storage capacity. Whereas similar methods are commonplace in many other fields of civil engineering, e.g. hydrology, no such tools are generally available to designers of bulk water supply systems.

The RAU Water Research Group developed simple software for the probabilistic analysis of water networks in a previous WRC project (Report no 985/1/02: *Development of a stochastic technique for the optimisation of pipe and reservoir systems*). The project also created a software package called Mocasim version 1, with which simple, linear supply systems could be analysed.

The biggest drawback of the first version of the Mocasim software is that it uses a simple mass balance model for calculating the flows in distribution systems, and does not incorporate the full hydraulic behaviour of these systems. While this approach works well for simple, linear supply systems, it is not adequate for the complicated hydraulic behaviour associated with most water distribution systems.

A need was therefore identified to include system hydraulics in the stochastic analysis methodology. This will allow complex bulk water distribution and reticulation systems to be analysed and thus extend the reach of the stochastic analysis methodology substantially. The main software package developed in this study is called ***Mocasim II.***

The aims of this research project were:

- To continue the WRC-supported drive to establish probabilistic analysis as a powerful adjunct to the usual, well established deterministic analysis used by water engineers to design and improve water supply systems
- To provide a powerful, easy-to-use design tool to support this initiative in design practice.

Implementation

In implementing the project, a number of technical obstacles had to be overcome to achieve the aim of hydraulically linked stochastic analysis of water distribution systems. Stochastic analysis require very long simulation runs, the ability to implement stochastic demands, fires and pipe failure events, and the ability to handle pressure-dependent demands and isolated network sections.

Hydraulic calculations were done using the public domain hydraulic modelling package ***Epanet***. Epanet can perform both snapshot (steady-state) and extended-period hydraulic analyses of incompressible flow in pipe networks. It was necessary to modify the Epanet hydraulic solver source code to overcome some of the technical obstacles of stochastic simulations. The following changes were implemented:

- Isolated network sections, due to empty reservoirs or failed pipes, are identified and managed to avoid convergence problems in the hydraulic network solver.
- Pressure-dependent demands are introduced when the capacity of the network is inadequate to supply the full network demand.
- To handle pressure-dependent demands, the behaviour of the Epanet emitter elements were adjusted to fix the emitter demand above a certain critical pressure, and to avoid negative emitter demands.

To provide an efficient interface between Mocasim II and the Epanet hydraulic engine, a programmer's toolkit called ***Ooten*** was developed. Ooten (Object Oriented Toolkit for Epanet) is an object-oriented shell for the Epanet source code. It gives programmers easy access to the Epanet source code functions through a number of objects and methods.

The Mocasim II software uses Ooten to handle hydraulic simulations of water distribution systems. Long simulation runs are implemented by repeatedly running a daily simulation in Epanet. Continuity is ensured by updating the starting tank levels to reflect the ending levels of the previous day.

Stochastic demands and fires are calculated in advance for a day and implemented in using standard Epanet demand patterns. Pipe failure events are implemented by closing pipes using controls.

Epanet has a well defined and friendly user interface. Since the source code for the user interface is also available as public domain software, a decision was taken to implement Mocasim II inside the Epanet user interface. This required amending the user interface to handle the additional input data required by Mocasim II, linking the user interface to the Mocasim simulation routines and adding display features to display the Mocasim results.

The first chapter of this document gives an overview of the project, its aims and accomplishments. The remaining chapters form a workbook for the Mocasim II software package. A number of examples are included to illustrate the application and benefits of stochastic modelling of water distribution systems.

ACKNOWLEDGMENTS

The research in this report emanated from a project funded by the Water Research Commission and entitled :

"K5/1389 – THE INTEGRATION OF SOFTWARE PACKAGES FOR THE PROBABILISTIC ANALYSIS OF COMPLEX WATER SUPPLY SYSTEMS"

The Advisory Committee responsible for this project, consisted of the following persons :

Mr JN Bhagwan	Water Research Commission (Chairman)
Mr S Gillham	Umgeni Water
Mr R Thompson	Rand Water
Dr R McKenzie	WRP Consulting Engineers
Mr A Cross	Johannesburg Water
Mr A Grobler	Goba Maholi Keevy Steyn
Mr J Ndiritu	Wits
Mr M van Dijk	University of Pretoria

The financing of the project by the Water Research Commission and the contribution of the members of the Advisory Committee are acknowledged.

This project was possible due to the co-operation of many individuals and institutions. The authors therefore wish to record their gratitude to the following:

Johannesburg Metropolitan Council	Water consumption data
Magalies Water	Water consumption and network data
City Council of Windhoek	Water consumption and network data

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE – PROJECT OVERVIEW

1.1 Introduction

A technique for the stochastic analysis of water distribution systems was developed under a previous Water Research Commission Project, published as WRC report no 985/1/02: *Development of a stochastic technique for the optimisation of pipe and reservoir systems*. While the current project builds upon this work, the theory and methodology essentially remain the same. The main advance offered by the current project is that the simple linear hydraulic model used in the previous project is replaced by a non-linear hydraulic simulation model, capable of simulating large and complex water supply systems.

At a project progress meeting held on 30 July 2003, it was decided that the project report should be published in the form of a workbook for the software package, rather than simply a reworking of the previous project report.

To provide the necessary background and context, the first chapter of this document gives an overview of the project, its aims and accomplishments. The remaining chapters form the workbook for the Mocasim II software package.

1.2 Background and Motivation

The demography and water resources of South Africa are such that the vast majority of its inhabitants live a substantial distance away from the closest water source. This necessitates a vast, expensive and often complex network of bulk water supply systems (consisting mainly of pipes and storage tanks) to transport water from its source to the immediate neighbourhood of the consumers. With the current national emphasis on bringing water to the previous unserved communities (often those which are furthest away from existing sources), huge capital investments are being made towards new additions to the bulk supply systems. Ensuring that the systems are optimally designed to provide acceptable service at an affordable cost, is of obvious importance.

Current design standards for bulk water supply systems do not allow much design flexibility. Firstly, they generally do not allow the designer to differentiate meaningfully between urban and rural systems, and secondly they do not allow the designer the

freedom to assume different levels of reliability. A small rural community, which had been limping along for many years with their own rudimentary water supply system, may opt for a cheaper system, even at lower reliability, since they have an alternative during the short periods of non-supply. Likewise, a large industry which can ill afford to shut down due to a water shortage, may opt for higher reliability, even if at higher cost.

For these reasons, a methodology was deemed necessary to allow the designer to couple reliability with system storage capacity. Whereas similar methods are commonplace in many other fields of civil engineering, e.g. hydrology, no such tools are generally available to designers of bulk water supply systems.

The RAU Water Research Group has gone a long way in developing methods for the probabilistic analysis of water networks for determining their reliability. Recently, the progress made over almost ten years of research was integrated in a WRC report no 985/1/02: *Development of a stochastic technique for the optimisation of pipe and reservoir systems*. This project demonstrated the added power and insight brought to water systems analysis by the probabilistic analysis approach. The project also created a software package called Mocasim version 1, with which simple, linear supply systems can be analysed. In summary, this project clearly established probabilistic analysis as a viable, necessary approach for the design of better, more efficient systems.

The biggest drawback of the first version of the Mocasim software is that it uses a simple mass balance model for calculating the flows in distribution systems, and does not incorporate the full hydraulic behaviour of these systems. While this approach works well for simple, linear supply systems, it is not adequate for the complicated hydraulic behaviour associated with most water distribution systems.

A need was therefore identified to include system hydraulics in the stochastic analysis methodology. This will allow complex bulk water distribution and reticulation systems to be analysed and thus extend the reach of the stochastic analysis methodology substantially.

1.3 Project Aims

The original research proposal for this project set out the following aims:

- To continue the WRC-supported drive to establish probabilistic analysis as a powerful adjunct to the usual, well established deterministic analysis used by water engineers to design and improve water supply systems
- To provide a powerful, easy-to-use design tool to support this initiative in design practice.

1.4 Implementation

In implementing the project, a number of technical obstacles had to be overcome to achieve the aim of hydraulically linked stochastic analysis of water distribution systems. Stochastic analysis require very long simulation runs, the ability to implement stochastic demands, fires and pipe failure events, and the ability to handle pressure-dependent demands and isolated network sections.

1.5 Epanet

Hydraulic calculations were done using the hydraulic modelling package Epanet. Epanet was developed by the United States Environmental Protection Agency to help water utilities to better understand the movement and transformations undergone by water in water distribution systems (Rossman 2000).

Epanet can perform both snapshot (steady-state) and extended-period hydraulic analyses of incompressible flow in pipe networks. It can handle various hydraulic components including reservoirs, tanks, pipes, pumps and control valves. Epanet can also model water quality by simulating the behaviour of chemical constituents in the water distribution system with time.

The Epanet hydraulic and water quality engine has become the standard for the modelling of water distribution systems and is used by several commercial software packages. Epanet also has its own user interface and is available as a stand-alone software package.

Epanet is as public domain software and can be downloaded from http://www.epa.gov/ORD/NRMRL/wswrd/epanet.html.

1.6 Ooten

To provide an efficient interface between Mocasim II and the Epanet hydraulic engine, a programmers toolkit called Ooten was developed. Ooten (Object Oriented Toolkit for Epanet) is an object-oriented shell for the Epanet source code. It gives programmers easy access to the Epanet source code functions through a number of objects and methods.

OOTEN was developed in standard ANSI C++ code and can be incorporated in any compatible programming code. It was developed for this project, but is generic in nature and can be used for any software project in which hydraulic calculations needs to be performed. For this reason Ooten was developed as a stand-alone code library with a comprehensive help function. More detail on Ooten is provided in Appendix C.

The Mocasim II software uses Ooten to handle hydraulic simulations of water distribution systems. Long simulation runs are implemented by repeatedly running a daily simulation in Epanet. Continuity is ensured by updating the starting tank levels to reflect the ending levels of the previous day.

Stochastic demands and fires are calculated in advance for a day and implemented in using standard Epanet demand patterns. Pipe failure events are implemented by closing pipes using controls.

Ooten is available for download from the RAU Water Research Group's website at http://general.rau.ac.za/civil/wrg/ooten.htm, or from the WRC website at http://www.wrc.org.za/wrcsoftware/ooten.htm.

1.7 Epanet Hydraulic Engine

It is inevitable in stochastic analysis of water distribution systems that reservoirs will run dry, sections of the network will be isolated and nodal pressures will be low or negative during a stochastic simulation run. A number of changes were required to the Epanet

hydraulic engine (i.e. the Epanet source code calculating the pressures and flows in a system) to handle these eventualities.

To handle network sections that are isolated due to empty reservoirs or failed pipes, the isolated sections are identified and all demands and pressures in these sections are set to zero.

In some cases, a failing pipe will not isolate a section, but will restrict the flow in such a way that the full demand cannot be met. Epanet uses a demand-driven model and will satisfy all demands, even when this causes negative pressures in the system. This is clearly not a realistic solution, since demands will reduce if the pressure in the system becomes low.

To handle this problem, the Epanet source code was changed to model demand as pressure dependent under certain conditions. When the nodal pressure is above a certain critical value, demands are assumed to be fixed and normal demand-driven analysis is performed. However when nodal pressures are below the critical pressure, demand is varied with the nodal pressure according to the equation:

$$q = ch^{\alpha}$$

with q the demand
h the pressure head at the node
c a constant coefficient
α a constant exponent.

Negative demands are not allowed, even when the nodal pressure is below zero.

To implement pressure dependent demands in the Epanet source code, demands at junctions with pressures below the critical pressure were handled by modelling them as special emitters. In the Epanet code, emitters are modelled as a fictional pipe connecting the node to a fictional reservoir whose elevation equals that of the node. This definition allows emitters to be included in the system equations without altering the sparse matrix structures used in the code.

Two changes were required to the Epanet emitters code to allow pressure-dependent demand models to be handled realistically. First, the Epanet code had to be amended to allow the demand to be fixed above a certain critical pressure. This was achieved by calculating the emitter matrix coefficient using the critical pressure rather than the true pressure at the node if the pressure at the node is above the critical pressure. To avoid fluctuation between the pressure dependent and fixed demand, a small tolerance is used when determining whether a nodal pressure is below or above the critical pressure.

The second change that had to be made to the Epanet emitters code was to ensure that emitters cannot have negative demands. The Epanet emitter function is implemented in such a way that negative pressure at a node will cause a negative demand (i.e. an inflow into the node). This problem was addressed by closing the virtual pipes belonging to emitters at nodes with negative pressures. Again a small tolerance is used to decide whether a pressure is negative or positive.

1.8 Epanet User Interface

Epanet has a well defined and friendly user interface. Since the source code for the user interface is also available as public domain software, a decision was taken to implement Mocasim II inside the Epanet user interface. This required amending the user interface to handle the additional input data required by Mocasim II, linking the user interface to the Mocasim simulation routines and adding display features to display the Mocasim results.

1.9 Structure of this Document

Chapter 1 gives an overview of the project and some of the technical obstacles that had to be overcome.

Chapter 2 gives a brief introduction to the Mocasim II software and workbook .

Chapter 3 explains the basic concepts used in Mocasim II and gives a theoretical framework of the methodology used. It also provides data obtained from various studies to assist the user in selecting realistic simulation parameters.

Chapter 4 provides instructions on how to install Mocasim II and contains a tutorial on how to use the software. It is assumed that users have a basic knowledge of modelling water distribution systems using Epanet.

Chapter 5 gives an overview of the Mocasim II methodology and environment and serves as a basic reference for Mocasim II.

Chapter 6 consists of a number of examples and case studies, chosen to demonstrate the capabilities of Mocasim II as well as providing the user with examples to assist in the use of the software.

Appendix A discusses the Mocasim II text input format.

Appendix B discusses the Ooten programmers toolkit developed as part of this project.

Appendix C summarizes the procedures inherent to the parameter extraction from consumption data, which was used for extracting the required model parameters from actual data sets.

Appendix D provides some input and output files for the systems discussed in Chapter 6

CHAPTER TWO – INTRODUCTION TO THE WORKBOOK

2.1 Introduction

Mocasim II is a software package combining stochastic analysis with hydraulic analysis. It allows various types of analysis to be performed, including capacity versus reliability analysis for service reservoirs. Mocasim II is based on a methodology that uses Monte Carlo analysis.

Mocasim II is designed to run inside a modified version of the Epanet user interface. Epanet is a computer program that performs extended period simulation of hydraulic and water quality behaviour within pressurized pipe networks. Epanet can be downloaded as public domain software from http://www.epa.gov/ORD/NRMRL. An Epanet users manual can also be downloaded from this site. It is assumed throughout this workbook that the user has a fair understanding of hydraulic modelling of water distribution systems and Epanet.

This workbook should be used in conjunction with Mocasim II's help facility. The objective of this workbook is to give an overview of the theoretical framework and methodology on which Mocasim is based, and provide the user with examples to assist him/her in using Mocasim II.

2.2 Method

The Water Research Group at the Rand Afrikaans University developed the methodology used in Mocasim II as part of an ongoing research programme. The technique was developed by Nel (1993) and improved by Haarhoff and Van Zyl (2002). The initial method used a simple mass balance model for calculating the flows in simple linear distribution networks and was used to quantify the relationship between the capacity and reliability of service reservoirs.

The method was developed further, by incorporating the Epanet hydraulic engine in the stochastic analysis method. The Epanet hydraulic engine had to be modified to handle pressure-dependent demands and isolated network sections, for example when

simulating pipe failures. The new method allows stochastic analysis to be applied to large and complex water distribution systems.

Mocasim II uses the Monte Carlo method to analyse the stochastic behaviour of a water distribution supply system. Monte Carlo simulation involves the repeated calculation of the system performance, each time with a different combination of input parameters, consisting of both deterministic and probabilistic components. The probabilistic components are randomly selected from probability distribution functions that describe the variability of each input parameter. The input parameters are those stochastic variables that affect the system, including consumer demand, fire-fighting demand and pipe failure behaviour. After a large number of iterations, typically using one hour time steps over a simulation period of hundreds or thousands of years, the performance of the system is reported statistically.

2.3 Capabilities of Mocasim II

In addition to the hydraulic and water quality capabilities due to the integration with the Epanet hydraulic engine, Mocasim's capabilities include the following:

- Capacity versus reliability analysis for service reservoirs.
- Failure behaviour analysis of service reservoirs including the number of failures, average failure duration and maximum failure duration.
- Calculation of statistics and frequency distributions of flow rate, unit headloss and status for each pipe in the system.
- Calculation of statistics and frequency distributions of demand, demand deficit, head and pressure for each node in the system.

2.4 Steps in using Mocasim II

To model a water distribution system stochastically using Mocasim II, the following steps are typically performed. The sections referenced refer to the quick start tutorial in Chapter 4.

- Model an Epanet network representation of the distribution system (Sections 4.3.to 4.5).
- Edit the stochastic analysis properties of the system (Sections 4.6 to 4.9)
- Set the stochastic analysis options (Section 4.10)
- Run the stochastic simulation (Section 4.11)
- View the results of the analysis (Section 4.12)

CHAPTER THREE – THEORETICAL BACKGROUND

3.1 Introduction

An adequate water supply is essential for community health and quality of life. Water supply systems should provide water reliably at adequate volumes, flowrates, pressures and quality. The reliability of supply is essential, but difficult to measure and design for in reticulation systems. One reason for the difficulty is that there exists no commonly accepted definition for reliability of a water supply system.

In this report, as in the previous one, reliability is defined in terms of the reliability of the municipal service reservoir. A service reservoir fails when it runs empty. The reliability of the reservoir, or lack thereof, can be defined in terms of the number, duration or maximum length of reservoir failures.

If a service reservoir fails, the users served by the reservoir will experience an interruption in service, and obviously also a failure in the system. However, a much less severe event, such as the system pressure below the specified minimum pressure, may also be regarded as a failure, as may a reduction in water quality. Mocasim II allows the performance of systems to be determined stochastically. The percentage of time that the pressure at a certain node is below the required minimum pressure, may for instance be calculated and used as a measure of reliability.

The reliability of a service reservoir is determined by its capacity and both the flow into and out of the reservoir. If the outflow volume exceeds the inflow volume by the reservoir's storage volume, a failure will occur. If the inflow capacity is less than the average demand, and infinitely large reservoir is required. On the other hand, if the inflow capacity is greater than the design peak flow, no balancing storage is required (although emergency and fire storage may still be necessary). Design guidelines attempt to balance these factors by specifying the required reservoir storage and inflow capacities.

Table 3.1 lists some guidelines used in South Africa. In all cases, the size of the reservoir is specified in terms of the annual average daily demand (AADD).

Table 3.1 Typical South African guidelines for sizing reservoirs and feeder pipes (Nel & Haarhoff, 1996)

Authority	Nature of supply	Feeder capacity	Service reservoir capacity
Department of Water Affairs	gravity feed	1.5 x AADD	24h of AADD
	pumped main	1.5 x AADD	48h of AADD
Co-operation & Development	gravity feed	1.5 x AADD	24h of AADD
	pumped main	1.5 x AADD	48h of AADD
National Building Institute	one source	1.5 x AADD	48h of AADD
	two sources	1.5 x AADD	36h of AADD
National Housing	gravity feed	1.5 x AADD	20h of AADD
	pumped main	1.5 x AADD	30h of AADD
	two sources	1.5 x AADD	66% of capacity with one source

The biggest problem with the guidelines listed in Table 3.1 is that they do not take the supply area's unique properties into account. They may be adequate for a typical system, but provide too low reliability for a system with a very long supply line or an abnormally high seasonal peak demand. If the municipal storage reservoir is close to the source and is supplied by a number of parallel pipes (as sometimes happens with municipal systems supplied directly from a bulk supplier's reservoirs) the guidelines may specify an overly conservative and expensive system.

Another limitation is that the relationship between the feeder pipe and the service reservoir is fixed, preventing the designer from exploring the most economic combination of feeder pipe and service reservoir. The service reservoirs probability of failure is also not considered when making use of such rigid guidelines (Nel & Haarhoff, 1996).
Figure 3.1 depicts the relationship between the reliability of a bulk supply system, the capacity of the service reservoir(s) and the size of the supply pipeline. The size of the service reservoir depends directly on the capacity of the bulk water supply pipeline.

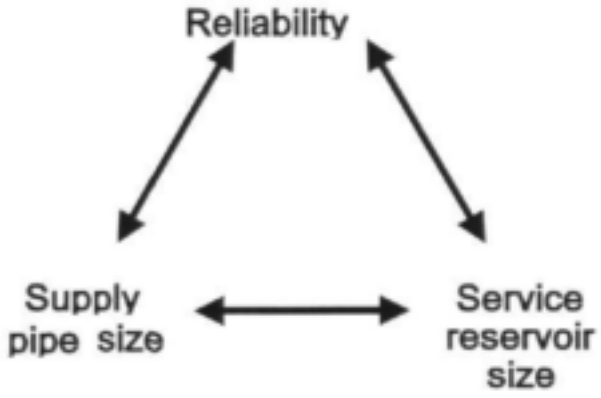

Figure 3.1 Interrelationship amongst bulk water supply variables

Figure 3.2 shows a section from a typical water demand pattern. If the supply rate is Q_{max}, no balancing reservoir is needed, because the peak demand can be supplied directly from the supply pipeline. If the supply rate is less than Q_{ave}, then a reservoir of infinite volume is required, because the long-term demand exceeds the supply capacity. Naturally, it follows that larger reservoirs would require smaller pipelines, and vice versa.

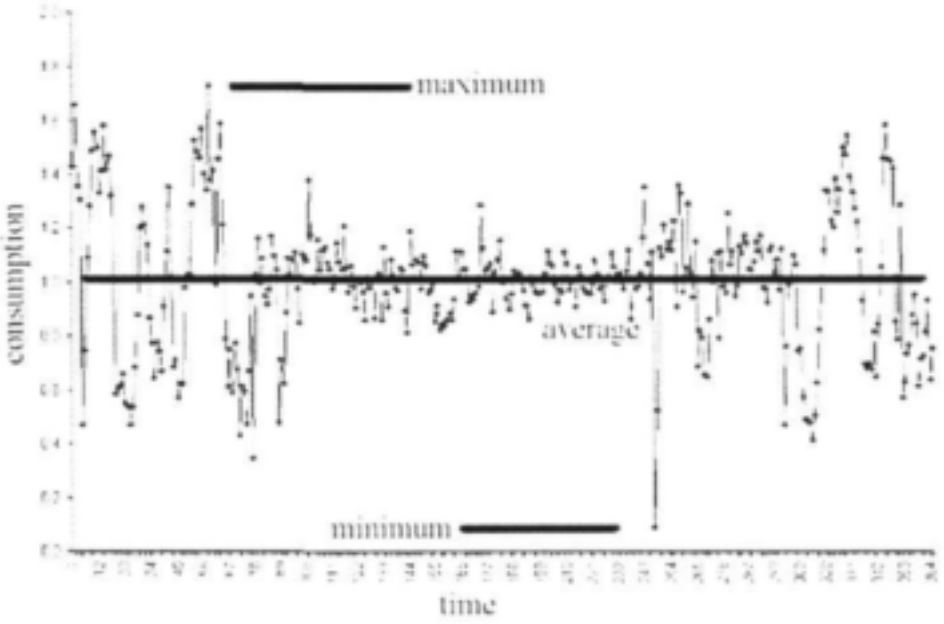

Figure 3.2 Typical water demand pattern

As shown in Figure 3.1, the reliability of supply plays a major role. Lewis (1996) described reliability as follows: "In the broadest sense, reliability is associated with dependability, with successful operation, and with the absence of breakdowns or failures.

It is necessary for engineering analysis however, to define reliability quantitatively as a probability. Thus reliability is defined as the probability that a system will perform its intended function for a specified period of time under a given set of conditions. A product or system is said to fail when it ceases to perform its intended function."

A reservoir that runs dry would constitute a failure event in the bulk water supply system and the part of the reticulation system served by it. The reliability of a bulk supply system can thus be described in terms of the failure behaviour of its reservoir, for instance in terms of the annual number of failure events, the total annual fail time, or the maximum failure duration.

Larger reservoirs would fail less often and would thus provide a higher level of reliability. However, the higher reliability comes at a cost of increased capital expenditure and the potential for water quality problems due to longer retention times in the reservoir. In some cases it might be better to improve the reliability of the system by increasing the capacity of the supply pipelines, changing the pipe configuration, or reducing the time for finding and repairing pipe bursts (Van Zyl & Haarhoff, 1999).

3.2 Mocasim II Methodology

The methodology on which Mocasim II is based uses stochastic analysis, which is useful for evaluating system reliability in a wide range of fields, including water source analysis, electrical power distribution and communications networks (Yang et al, 1996). It is frequently used in the analysis of complex systems where risk and uncertainty play important roles.

A stochastic model is one whose outputs are predictable only in a statistical sense. With a stochastic model, repeated use of a given set of model inputs produces outputs that are not the same but follow certain statistical patterns (Haan, 1977).

In stochastic analysis it is often necessary to assign a probability of occurrence to different possible values of a variable. We may, for instance, know from experience that the minimum time required to repair a burst on a particular pipe is 3 hours, the maximum 36 hours and the median 12 hours. By fitting a suitable function to these values, a probability distribution function may be defined which can then be used in a stochastic

analysis to simulate the pipe's failure duration behaviour (Haarhoff & Van Zyl, 2002). The probability distribution functions used with this software are summarized later in this chapter including a table listing their properties.

The stochastic method used in this methodology is the Monte Carlo simulation method. Monte Carlo methods have been used for a long time, but only in the past several decades has the technique gained the status of a full-fledged numerical method capable of addressing the most complex applications (CSEP, 2001).

Monte Carlo simulation entails the repeated calculation of the system performance, each time with a different combination of input parameters. The input parameters are randomly selected from probability distribution functions that describe the variability of each input parameter. Each calculated system performance is compared against some performance criterion: if the performance criterion is satisfied, that event is recorded as successful. If not, the event is recorded as a failure. After a large number of iterations, the performance of the system is reported statistically (Haarhoff & Van Zyl, 2002).

The components of a Monte Carlo simulation method may include the following:

- Probability distribution functions: the physical (or mathematical) system must be described by a set of probability distribution functions.
- Random number generator: a source of random numbers uniformly distributed on the unit interval must be available.
- Sampling rule: a prescription for sampling from the specified probability distribution functions, assuming the availability of random numbers on the unit interval, must be given.
- Scoring (or tallying): the outcomes must be accumulated into overall tallies or scores for the quantities of interest.
- Error estimation: an estimate of the statistical error (variance) as a function of the number of trials and other quantities must be determined.
- Variance reduction techniques: methods for reducing the variance in the estimated solution to reduce the computational time for Monte Carlo simulation
- Parallelization and vectorization: algorithms to allow Monte Carol methods to be implemented efficiently on advanced computer architectures (CSEP, 2001).

The input parameters are those stochastic variables that affect the system: consumer demand, fire-fighting demand and pipe failure events. Once estimates of the probability distributions of these components are available, Monte Carlo simulation is used to generate stochastic combinations of these variables for any required number of iterations, each iteration covering a time step of arbitrary length. The performance of the system is then reported statistically.

An important benefit of stochastic analysis is that it allows the inherent relationship between service reservoir capacity and reliability to be quantified. Service reservoir capacities are normally determined through deterministic design guidelines, such as 48 hours of average annual demand. The design guidelines implicitly specify a level of reliability for a typical service reservoir. Design guidelines are normally not fixed but require engineering judgment when applying them to water supply systems, such as exceptionally important service reservoirs or systems with very long or short supply lines.

The main functions of service reservoirs are to provide reserves for demand balancing, emergencies (mainly service reservoir supply interruptions) and fire fighting. By modelling all these processes stochastically, the number of failures, average failure duration and other performance parameters may be calculated.

3.3 Statistical Background

This section provides some basic definitions of statistical properties and descriptions of the distribution functions used in Mocasim II.

A frequency distribution is a table that divides a set of data into a suitable number of classes (categories) showing also the number of items belonging to each class. The most common form of graphical presentation of a frequency distribution is the histogram. The histogram of a frequency distribution is constructed of adjacent rectangles; the heights of the rectangles represent the class frequencies and the bases of the rectangle extend between successive class boundaries. An example of a histogram is shown in Figure 3.3.

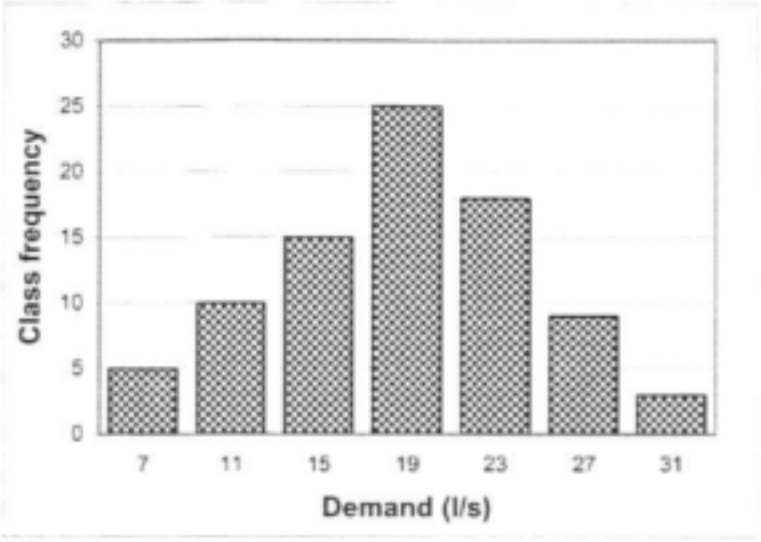

Figure 3.3 Example of a demand histogram

Inspection of a histogram often brings out features that are not immediately apparent from the data themselves. Numerical measures to describe a data set will now be introduced.

Consider the data set $x_1, x_2, x_3, \ldots, x_i, \ldots, x_n$ for a general sampling of n measurements. Here x_i is the i^{th} observation in the list so x_1 represents the value of the first measurement, x_2 represents the value of the second measurement, and so on.

The mean of the data set is defined by the formula:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

It is a descriptive measure of the centre of a set of data. Sometimes it is preferable to use the median as a descriptive measure of the centre, or location, of a set of data. This is true, particularly, if it is desired to minimize the calculations or if it is desired to eliminate the effect of extreme (very large or very small) values.

The median of n observations $x_1, x_2, \ldots, x_n$ can be defined loosely as the "middlemost" value once the data are arranged according to size. More precisely, if the observations are arranged according to sized and n is an odd number, the median is the value of the

One of the most important characteristics of almost any set of data is that the values are not all alike. The extent to which they are unalike, or vary among themselves, is of basic importance in statistics. Measures such as the mean and median describe one important aspect of a set of data – their "middle" or their "average" – but they tell us nothing about this other basic characteristic. We observe that the dispersion of a set of data is small if the values are closely bunched about their mean, and that it is large if the values are scattered widely about their mean. Therefore if the set of numbers $x_1, x_2, ..., x_n$ has mean $\bar{x}$, the differences $x_1 - \bar{x}$, $x_2 - \bar{x}$, ..., $x_n - \bar{x}$ are called the deviations from the mean. The sum of the deviations is always zero.

The sample variance, s^2, is the average of the squared deviations from the mean $\bar{x}$, and is defined by the formula

$$s^2 = \sum_{i=1}^{n} \frac{(x_i - \bar{x})^2}{n-1}$$

The reason for dividing by n–1 instead of n is that there are only n –1 independent deviations $x_i - \bar{x}$. Because their sum is always zero, the value of any particular one is always equal to the negative of the sum of the other n–1 deviations. If many of the deviations are large in magnitude, either positive or negative, their squares will be large and s^2 will be large. When all the deviations are small, s^2 will be small.

The standard deviation of n observations $x_1, x_2, ..., x_n$ is defined as the square root of their variance,

$$s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}}$$

The standard deviation is by far the most generally useful measure of variation. Its advantage over the variance is that it is given the same units as the observations.

A random variable is defined as a real-valued function for which the domain is a sample space. In other words, Let Y denote a variable to be measured in an experiment (where an experiment is the process of making an observation). Because the value of Y will

vary depending on the outcome of the experiment, it is called a random variable (Wackerly, Mendenhall, Scheaffer, 1996).

A random variable X is said to be continuous if it can take an infinite number of possible values associated with intervals of real numbers, and there is a function f(x), called the probability density function, such

that (Scheaffer & McClave, 1986):

- $f(x) \geq 0$, 0, for all x.
- $\int_{-\infty}^{\infty} f(x)dx = 1$
- $P(a \leq X \leq b) = \int_{a}^{b} f(x)dx$

A random variable X is said to be discrete if it can take on only a finite number or a countable infinity, of possible values x. In this case (Scheaffer & McClave, 1986):

- $P(X = x) = p(x) \geq 0$
- $\sum_{x} P(X = x) = 1$, where the sum is over all possible values of x

The distribution function F(b) for a random variable X is defined as

$$F(b) = P(X \leq b)$$

If X is discrete,

$$F(b) = \sum_{x=-\infty}^{b} p(x)$$

where p(x) is the probability function. If X is continuous,

$$F(b) = \int_{-\infty}^{b} f(x)dx$$

where f(x) is the probability density function.

Uniform Distribution

A uniform distribution is one for which the probability of occurrence is the same for all values of X. It is sometimes called a rectangular distribution. For example, if a fair die is thrown, the probability of obtaining any one of the six possible outcomes is 1/6. Since all outcomes are equally probable, the distribution is uniform. If a uniform distribution is divided into equally spaced intervals, there will be an equal number of members of the population in each interval as shown in Figure 3.4 (Lane, 2002).

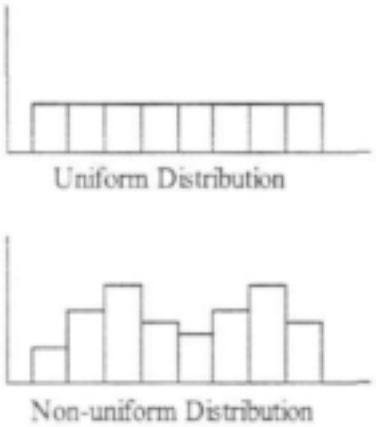

Figure 3.4 Uniform versus non-uniform distribution (Lane, 2002)

The uniform distribution, with the parameters α and β, has probability density function

$$f(x) = \begin{cases} \dfrac{1}{\beta - \alpha} & for\ \alpha < x < \beta \\ 0 & elsewhere \end{cases}$$

whose graph is shown in Figure 3.5.

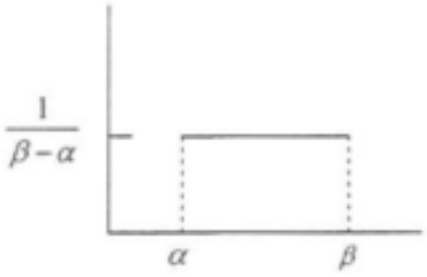

Figure 3.5 Continuous uniform distribution (Miller & Freund, 2000)

Note that all values of x from α to β are "equally likely" in the sense that the probability that x lies in an interval of width Δx entirely contained in the interval from α to β is equal to $\frac{\Delta x}{(\beta-\alpha)}$, regardless of the exact location of the interval.

Normal Distribution

Normal distributions are a family of distributions that have the same general shape. They are symmetric with scores more concentrated in the middle than in the tails. Normal distributions are sometimes described as bell shaped. Examples of normal distributions are shown in the Figure 3.6 below. Notice that they differ in how spread out they are. The area under each curve is the same. The height of a normal distribution can be specified mathematically in terms of two parameters: the mean and standard deviation (Lane, 2002).

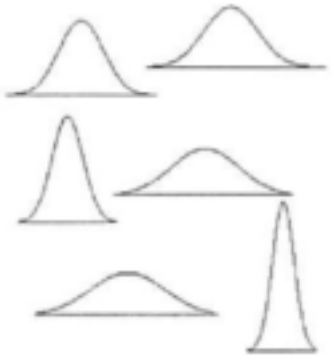

Figure 3.6 Normal distributions (Lane, 2002)

The equation of the normal probability density, whose graph is shown in Figure 3.7, is

$$f(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \quad -\infty < x < \infty$$

where the parameters μ and σ are its mean and standard deviation.

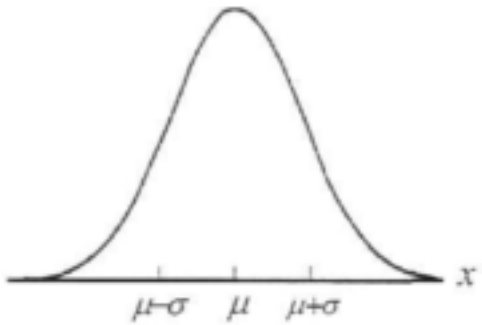

Figure 3.7 Graph of normal probability density (Miller & Freund, 2000)

Log normal distribution

The log normal distribution occurs in practice whenever we encounter a random variable, which is such that its logarithm has a normal distribution. Its probability density is

$$f(x) = \begin{cases} \frac{1}{\sqrt{2\pi\beta}} x^{-1} e^{\frac{-(\ln x-\alpha)^2}{2\beta^2}} & for\ x > 0, \beta > 0 \\ 0 & elsewhere \end{cases}$$

where ln x is the natural logarithm of x. A graph of the lognormal distribution with α = 0 and β = 1 is shown in Figure 3.8.

Figure 3.8 Lognormal distribution
(Miller and Freund, 2000)

It can be seen from the figure that this distribution is positively skewed or has a long right-hand tail.

The formal definition and properties of the probability distributions discussed above are summarised in Table 3.2.

Table 3.2 Properties of probability distributions

	Uniform	Normal	Lognormal
f(x) =	$\frac{1}{\beta-\alpha}$	$\frac{1}{\sqrt{2\pi}}e^{\frac{-x^2}{2}}$	$\frac{1}{\sqrt{2\pi\beta}}x^{\frac{-(\ln x-\alpha)^2}{2\beta^2}}$
x =	$\alpha<x<\beta$	$-\infty<x<\infty$	$x>0,\beta>0$
for p	$0\le p\le 1$	$0\le p\le 1$	$0\le p\le 1$
Mean (μ)	$\frac{\alpha+\beta}{2}$	0	$e^{\alpha+\frac{\beta^2}{2}}$
Variance (σ^2)	$\frac{(\beta-\alpha)^2}{12}$	1	$e^{2\alpha+\beta^2}(e^{\beta^2}-1)$

Cumulative Distribution Function

The cumulative distribution function is the probability that the variable takes a value less than or equal to x. This is

$$F(x)=P_r[X\le x]=\alpha$$

For a continuous distribution, this can be expressed mathematically as:

$$F(x) = \int_{-\infty}^{x} f(\mu) d\mu$$

For a discrete distribution, the cumulative distribution function can be expressed as

$$F(x) = \sum_{i=0}^{x} f(i)$$

The difference between a cumulative distribution function and a distribution function is that a distribution function is a density which represents the proportion that lies between specified values, and a cumulative distribution function is a curve whose value is the proportion with values less than or equal to the value on the horizontal axis (Dallal, 2000). Figure 3.9 shows an example of a normal distribution function and Figure 3.10 shows the corresponding cumulative distribution function.

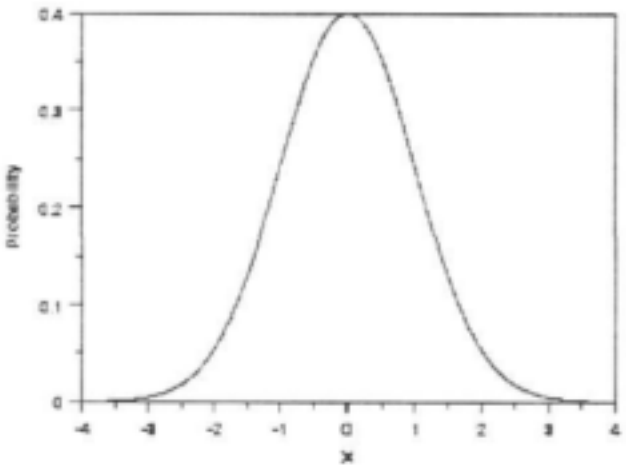

Figure 3.9 Normal distribution function

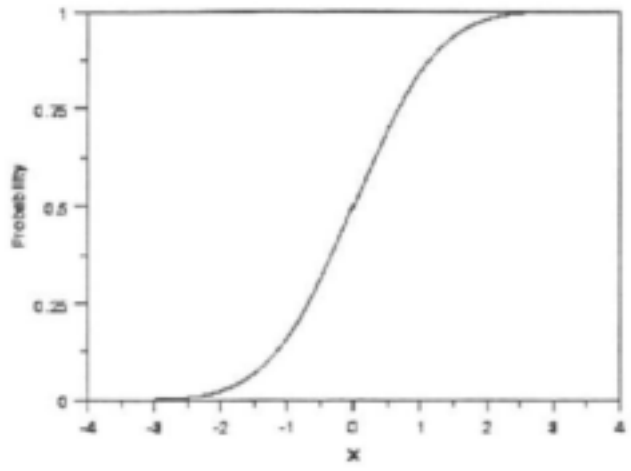

Figure 3.10 Normal cumulative distribution function

3.4 Stochastic Input Parameters

In this section we identify the parameters that contribute to system failure and identify the stochastic nature of these parameters, used in the Mocasim II software.

Service reservoirs usually have to provide more than one or all of the functions listed below:

- **Emergency storage**

  Volume must be provided to guarantee the supply to the consumers, even when the supply to the tank is partially or completely discontinued. Such events may be scheduled for maintenance purposes, which is not a stochastic variable. More important for probabilistic analysis are the unscheduled events due to pipe, power or source failures. The volume required for these unscheduled events is stochastic, as neither the time of occurrence nor the duration of the interruption can be predicted.

- **Fire storage**

  It is essential to have an adequate water supply available for fire fighting. This is mostly supplied through the water reticulation system, and is thus also drawn from the service reservoir. Most guidelines will specify an additional volume of water for which allowance must be made in the service reservoir. This approach is based on the conservative assumption that the fire demand will coincide with a period of

maximum consumer demand. The required fire storage is stochastic, as neither the time of occurrence nor the actual volume required can be predicted.

- **Demand storage**

  Consumers draw water from a service reservoir at a variable rate, whereas the supply line usually delivers water at a constant rate. The storage tank has to absorb the difference between inflow and outflow rates. The flow from the supply line into the reservoir is easily determined and usually well controlled. The consumer demand, on the other hand, is determined by the cumulative effect of a multitude of stochastic variables and is therefore itself a stochastic variable.

- **Operational requirements**

  There could be additional requirements for storage tank volume, such as freeboard (dependent on the sophistication of level sensing and control equipment), bottom storage (dependent on potential of air or sediment entrainment at the outlet), or a pump control band (required for automatic switching of pumps if water is being pumped to or from the reservoir). These components are all deterministic, i.e. they can be calculated once and simply added to the volume required for the stochastic components described above. (Haarhoff & Van Zyl, 2002)

3.4.1 Model for Emergency Storage

The volume required for emergency storage is equivalent to the total water demand for the duration of the supply interruption. To determine the emergency storage volume we need to estimate two factors; the first is the frequency of supply interruptions and the second is the duration of the interruption. The supply to a service reservoir is interrupted when the raw water source or water treatment plant fails, pumping installations fail (e.g. due to power cuts), or the feeder pipe system fails (e.g. pipe that breaks). These reasons for supply interruptions are the most common, however only pipe failures are considered here, as they are the most common cause of prolonged supply interruptions in practice.

3.4.1.1 Frequency of Supply Interruptions

To model the frequency of pipe breakages, it is necessary to know the length of the pipe and the average pipe breakage rate. From these two values the average number of breaks per year can be calculated. The frequency of supply interruptions is modelled using the time between fail events i.e. the number of days until the next fail event. For example, in a 1 km pipe, if the average fail rate is 2 failures per year, the average time between failures will be 182.5 days.

Table 3.3 shows the results concerning pipe failure frequency for different diameter categories from a study by A. Van der Mey. Figure 3.11 is a graphical representation of this data. Three years of data from the Johannesburg municipal area was studied, which consists of 3000 km of underground pipes. The focus of this study was on underground steel pipes.

Table 3.3 Failure/km/year relationships for each pipe diameter category for steel pipes

Diameter Category	100 - 199	200 - 299	300 - 449	450 - 599	> 600
Total bursts	2067	133	79	22	27
Total length (km)	1180	194	214	63	94
Burst/km per year	0.58	0.23	0.12	0.12	0.1

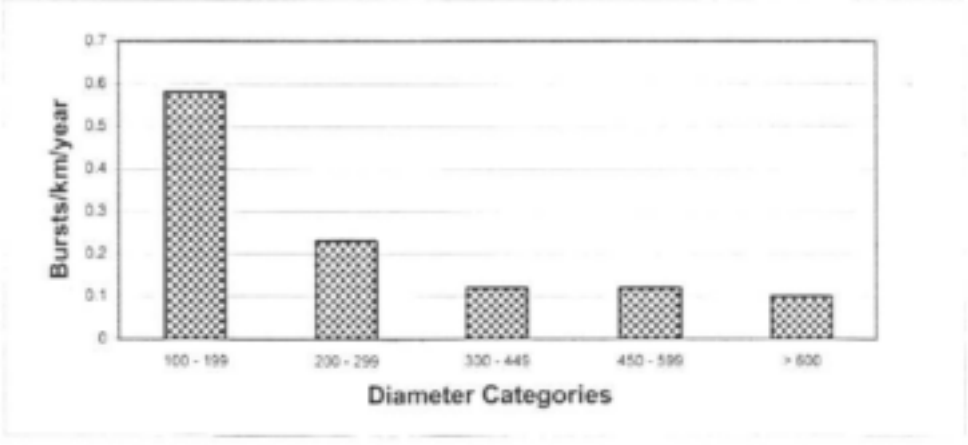

Figure 3.11 Graphical representation of failure/km/year relationships for each pipe diameter category for steel pipes.

Figure 3.12 depicts the number of failures per kilometre in three different water supply systems. These results were obtained during a study by Pelletier (2003). A modelling strategy was applied to these three systems, which were characterized by their brief recorded pipe break histories. Table 3.4 describes the characteristics of the three networks.

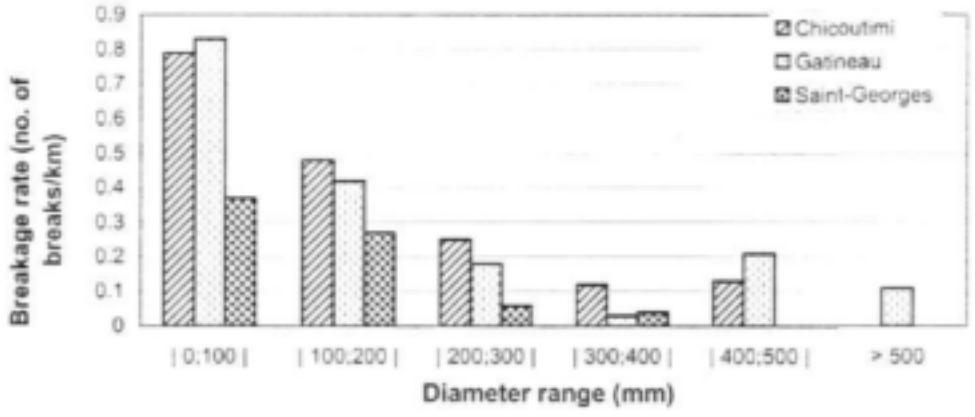

Figure 3.12 Breakage rates in different diameter categories for three municipalities in 1996 (Pelletier et al, 2003)

Table 3.4 Characteristics of Three Municipalities in France (Pelletier et al, 2003)

Characteristics	Chicoutimi	Gatineau	Saint-Georges
Approximate population	64 000	93 000	20 000
Pipe network length (km)	352	407	155
Ratio of pipe length (m) per inhabitant	5.5	4.4	7.8
Number of pipe segments in database	2 096	1 554	1 806
Average length of pipe segment in database (m)	168	262	86
Number of pipe breaks in database	1 719	1 426	279
Year of installation of first pipes	1981	1945	1949
Number of years of recorded pipe breaks	21	16	10
Number of pipe breaks per 100 km in 1996	46	36	19

Table 3.5 below summarises the pipe failure rates obtained from the study mentioned above, as well as other sources.

Table 3.5 Pipe failure rates from different sources

Study	Location	Material	Diameter mm	Area	Rate #/km/yr
Kettler et al (1985)	Winnipeg	Cast Iron	150	Urban	1.06
Kettler et al (1985)	Manhattan	Cast Iron	150	Urban	0.34
Kettler et al (1985)	Philadelphia	Cast Iron	150	Urban	0.32
Kettler et al (1985)	Manhattan	Cast Iron	300	Urban	0.11
Kettler et al (1985)	Winnipeg	Cast Iron	300	Urban	0.07
Kettler et al (1985)	Philadelphia	Cast Iron	300	Urban	0.05
Haarhoff et al (1999)	Magalies Water	Fibre Cement	150-300	Rural	0.074
GLS (1995)	Stellenbosch	Fibre Cement	50	Urban	1.63
GLS (1995)	Stellenbosch	Fibre Cement	75-100	Urban	1.43
GLS (1995)	Stellenbosch	Fibre Cement	150	Urban	0.48
GLS (1995)	Stellenbosch	Fibre Cement	200	Urban	0.1
GLS (1995)	Stellenbosch	Fibre Cement	250	Urban	0.05
Haarhoff et al (1999)	Magalies Water	PVC	90-160	Rural	0.041
Van der Mey (1995)	Johannesburg	Steel	100-199	Urban	0.58
Drusani (1995)	Northern Italy	Steel	100	Mixed	0.32
Van der Mey (1995)	Johannesburg	Steel	200-299	Urban	0.23
Drusani (1995)	Northern Italy	Steel	150-250	Mixed	0.16
Van der Mey (1995)	Johannesburg	Steel	300-449	Urban	0.12
Van der Mey (1995)	Johannesburg	Steel	450-599	Urban	0.12
Van der Mey (1995)	Johannesburg	Steel	600+	Urban	0.1
Nel (1993)	Rand Water (70's)	Steel	600+	Mixed	0.054
Nel (1993)	Rand Water (80's)	Steel	600+	Mixed	0.042
Drusani (1995)	Northern Italy	Steel	300-600	Mixed	0.04
Pelletier et al (2003)	Chicoutimi	Cast Iron	100-500	Urban	0.71
Pelletier et al (2003)	Gatineau	Cast Iron	100-500	Urban	0.69
Pelletier et al (2003)	Saint-Georges	Cast Iron	100-500	Urban	0.26
Pelletier et al (2003)	Chicoutimi	Ductile Iron	100-500	Urban	0.3
Pelletier et al (2003)	Gatineau	Ductile Iron	100-500	Urban	0.22
Pelletier et al (2003)	Saint-Georges	Ductile Iron	100-500	Urban	0.16
Pelletier et al (2003)	Chicoutimi	PVC	100-500	Urban	0.2
Pelletier et al (2003)	Gatineau	PVC	100-500	Urban	0.35

3.4.1.2 Duration of Supply Interruptions

The total interruption time associated with a pipe break is made up of a number of components, including:

- Reporting time from the instant of break until the maintenance centre is notified,
- Mobilization time to assemble crew, spares, equipment and vehicles,
- Travel time from maintenance centre to point of break,
- Preparation time, i.e. shutting valves, draining pipeline, and excavation to the level of the pipe break,
- Repair time, and
- Commissioning time, i.e. disinfection, air bleed, and controlled restoration of full pipeline pressure.

It is notable that the total interruption time for rural and urban will be dominated by completely different factors. In urban situations, the reporting and travel times will be minimal, as any break will be immediately visible and the maintenance centre is usually situated close by within the urban area. One could, however, expect that the excavation and repair time in a congested area, shared by numerous other services, will have to be slow and careful. For rural areas, the total interruption time is dominated by reporting and travel times due to remote location, absence of the people, and poor roads and communications. Excavation and repair time, however, will be shorter due to unrestricted room and mostly smaller pipeline diameters.

To model the duration of these interruptions in a probabilistic way, it is necessary to have a) a statistical distribution that adequately fits the observed or anticipated interruption, b) the mean interruption time, and c) the coefficient of variation describing the variability of the interruption time about the mean.

It is necessary to estimate the variability of the interruption time, as all breaks do not take exactly the same time to repair.

The literature, based on local studies, indicated that the lognormal distribution is appropriate when the interruption times are relatively short (as in urban areas), and that the normal distribution is appropriate when interruption times are longer (as in rural areas).

Two cases have been analysed by the RAU Water Research Group during previous investigations. In the first case, total interruption time was estimated for the CBD of the City of Johannesburg, based on the analysis of the maintenance records over a period of 36 months. The job cards (which indicated the time of start and end of each job) were assumed to represent the total interruption time. The reporting and mobilization times were therefore ignored for reasons mentioned earlier. In the second case, the total interruption time was estimated for the extensive rural system supplying a number of small villages along the northern edge of the Pilansberg in the North-West Province. It will be called the Mabeskraal system, as Mabeskraal is the largest village at the far end of the system. Here the total interruption time for all the breaks over a period of 19 months was estimated by plotting each break and calculating the travel time based on an average travelling speed. Fixed times were assumed for reporting, mobilization, preparation, repair and commissioning. Despite the shortcomings that are evident in both cases, the results obtained are considered to be valuable pointers until more exhaustive surveys and estimates are made. The results are summarized in Table 3.6.

Table 3.6 Duration of supply interruptions

Location	**Mabeskraal**	**Johannesburg**	**Johannesburg**	**Johannesburg**
System type	rural	Urban	urban	Urban
Diameter	90 - 350 mm	100 - 199 mm	200 - 299 mm	450 - 449 mm
Material	Mixed	Steel	Steel	Steel
Average duration	16.7 h	3.5 h	5.1 h	6.6 h
Distribution	Normal	Log-normal	Log-normal	Log-normal
Coefficient of variation	24%	14%	14%	14%

3.4.2 Model for Fire Storage

3.4.2.1 Fire Duration and Fire Flow Rate

To get a true probabilistic estimate, the fire frequency, duration and flowrate have to be described in statistical terms. In a study by Van Zyl (1993), the fire flow records of Johannesburg were analysed for 12 consecutive years. From this extensive database, the "large" fire events (those using more than 5000 litres of water) were selected and subjected to frequency analysis. The results from this analysis are summarized in Figure 3.13 and Figure 3.14. From data such as this, the mean, the appropriate statistical distribution and the standard deviation can be obtained.

Such data is rarely available in South Africa, and their retrieval and analysis are extremely tedious. Moreover, as large fires are rather infrequent, most of the sources will yield too few data points to allow meaningful statistical analysis. A much simpler, more direct approach is to get a deterministic estimate from the relevant fire codes. These codes usually specify both the maximum fire flow rate and the fire duration as a function of the land use category, population density, or other parameters. By multiplying these values, and estimate can be obtained. Such values were reflected in Table 3.7.

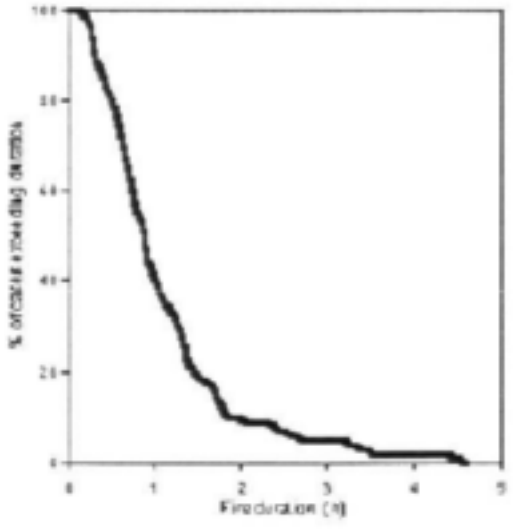

Figure 3.13 Johannesburg fire duration (Van Zyl, 1993)

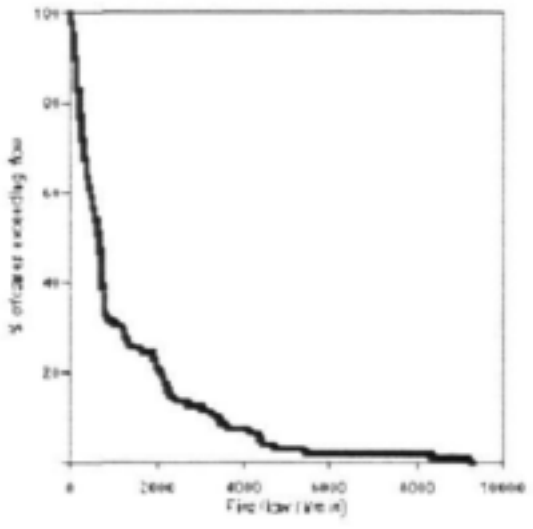

Figure 3.14 Johannesburg fire flow (Van Zyl, 1993)

Table 3.7 Comparison of Fire Codes (Van Zyl, 1993)

Parameter	Germany	Netherlands	USA	South Africa
Fire flow (l/min)				
high risk	3200	6000	17700	12000
moderate risk	1600	3000	11800	6000
low risk	800	1500	3800	900
Pressure (m)	15	20	14	15
high risk	15	20	14	15
moderate risk	15	20	14	7
low risk				
Fire duration (h)	2	2	4	6
high risk	2	2	3	4
moderate risk	2	2	2	2
low risk				
Code	DVGW-W405	KIWA #50 (1977)	AWWA M31 (1989)	SABS 090 (1972)

3.4.2.2 Probability of a Fire Occurring

It is difficult to estimate the probability of a fire occurring in the supply area of a water supply system, even if historical records are available. Most fires require very little or no fire water from the water supply system, and these smaller fires are obviously of little consequence for the design of the supply system. What is really needed, is the probability of a large fire occurring; a fire that will exert a substantial demand from the supply system. This is especially important if the fire water volume is estimated from the local fire code, as this value indicates an extreme fire event.

3.4.3 Model for Water Demand

An actual water demand pattern is the sum of two sets of factors:

- A number of cyclic factors, introduced by the seasons of the year or the days of the week.
- A random factor (termed white noise in this report), which is introduced by the cumulative effects of unaccounted-for variables such as temperature, rainfall, holidays, special events within the supply area, etc.

Note that the objective is to generate a series of hypothetical water demand volumes that, over the long term, will have the same statistical properties as an actually measured data set over a long term.

3.4.3.1 Random Component

The estimation of a daily volume of water used starts out by selecting a random value X_{random} from a normal distribution function with mean μ and standard deviation σ:

$$X_{random} \sim P(\mu, \sigma)$$

3.4.3.2 Serial Correlation

Serial correlation (also called autocorrelation) occurs when residuals from adjacent measurements in a time series are not independent of one another (that is, if the i^{th} residual is high, it is likely that the $(i+1)^{th}$ residual will also be high, and likewise low residuals tend to follow other low residuals).

Daily water demand data normally exhibit a strong degree of serial correlation. For example, if yesterday had a higher-than-average demand, today is also likely to have a higher-than-average demand. Behaviour of this kind is statistically described with a serial correlation coefficient Φ.

The correlation coefficient measures the similarity of variation of two residuals. Correlation coefficients range from +1 (which indicates a perfect linear relationship between two residuals), to 0 (which indicates absolutely no relationship), to –1 (for a perfect inverse linear relationship). A correlation coefficient between 0 and +1 means that the variables increase or decrease together, but not identically.

It is only necessary to take one preceding day into account (a lag-one model), as the analysis of numerous South African data sets have indicated that further serial correlation coefficients are not significant. The first correction (for serial correlation) is given by:

$$Xi = (1 - \Phi_1) * X_{random} + \Phi_1 * Xi_{-1}$$

3.4.3.3 Systematic Factors

The second correction is due to the systematic day-to-day variation found in a typical week. The average daily volume for a particular day is calculated by multiplying by a peak factor $PF_{Mon...Sun}$:

$$Xi = PF_{Mon...Sun} * Xi$$

The third correction is due to the systematic annual seasonal variation. Some prefer to describe the variation in terms of 12 months, others in terms of 52 weeks. The average daily volume for each of these time units, is calculated by multiplying by an appropriate peak factor $PF_{1...13}$:

$$Xi = PF_{1...13} * Xi$$

The fourth and final step is to dimensionalise the daily volume by multiplying with the Annual Average Daily Demand AADD:

$$Xi = AADD * PF_{1...13} * PF_{Mon...Sun} * Xi$$

The complete model it thus given by:

$$V_i = AADD^* = PF_{1...13} * PF_{Mon...Sun} * [(1 - \Phi_1) * X_{random} + \Phi_1 * Xi_{-1}]$$

If the simulation is done with a time step of one day, the above model is complete. If a time step of one hour is used, the approach is to generate a daily volume as indicated above, and then to disassemble the daily volume into 24 hourly volumes according to a fixed hourly pattern. The fixed hourly pattern is defined by $PF_{1...24}$. The hourly volumes are thus obtained by:

$$H_{1...24} = PF_{1...24} * [V_i/24]$$

CHAPTER FOUR – QUICK START TUTORIAL

4.1 Introduction

This chapter guides the user step-by-step through a simple Mocasim II simulation. It will assist the user to understand the modelling process and some of the capabilities of Mocasim II.

A basic understanding of hydraulic modelling of water distribution systems and working with Epanet is assumed. Epanet can be downloaded as public domain software from http://www.epa.gov/ORD/NRMRL. A user's guide is also available from this web site.

4.2 Installing and uninstalling Mocasim II

Mocasim II is designed to run inside a modified version of the Epanet user interface. It is distributed as a single file, mocasim2.exe, which contains a self-extracting set-up program. Mocasim II is available for download from the RAU Water Research Group's website at http://general.rau.ac.za/civil/wrg/mocasim.htm, or from the WRC website at http://www.wrc.org.za/wrcsoftware/mocasim.htm. Mocasim II runs under the Windows operating system. To install Mocasim II follow the steps below:

1. Select Run from the Windows Start menu.
2. Enter the full path name of the mocasim2.exe file e.g. C:\Mocasim\mocasim2.exe, or alternatively, click on the Browse button to locate it on your computer.
3. Click the OK button to begin the set-up process and follow the instructions.
4. Indicate the folder where the Mocasim files will be placed. The default folder is C:\Program files\Mocasim2.

Once the files are installed, Mocasim II will be listed in the Start / Programmes menu. To run Mocasim II simply select this item from the Windows menu.

If you wish to uninstall Mocasim II, follow the steps below:

1. Select Settings from the Windows Start menu.
2. Select Control Panel from the Settings menu.

3. Double-click on the Add/Remove Programs item.
4. Select Mocasim II from the list of programs that appears.
5. Click the Add/Remove button and follow the instructions.

4.3 Example Network Layout

In this tutorial we will analyse the simple distribution network shown in Figure 4.1.

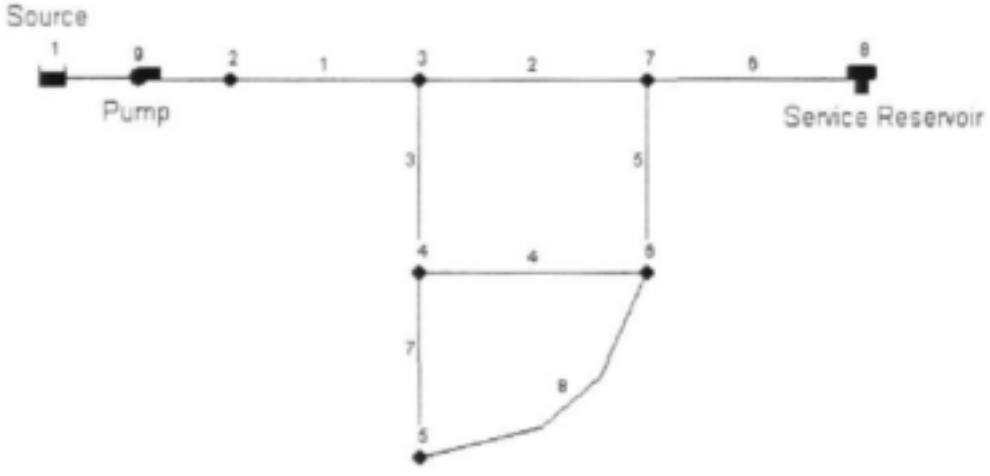

Figure 4.1 Example system layout

The example network consists of a source from which water is pumped into a two-loop network. There is also a pipe leading to a service reservoir. The nodes and link properties of the network are given in Tables 4.1 to 4.2 respectively.

Table 4.1 Example system node properties

Node	Elevation (m)	Demand (l/s)
1	200	0
2	200	0
3	203	10
4	200	10
5	187	13
6	200	10
7	200	0
8	239	0

Table 4.2 Example system link properties

Link	Length (m)	Diameter (mm)	C_H
1	900	350	100
2	1500	300	100
3	1500	200	100
4	1500	200	100
5	1500	200	100
6	2100	250	100
7	1500	150	100
8	2100	150	100

The pump can deliver 4 m of head at a flow of 47 l/s. The service reservoir has a diameter of 20 m, floor elevation of 239 m, initial water level of 1 m, and a maximum water level of 6 m.

4.4 *Building the hydraulic network*

Mocasim II is incorporated into the Epanet user interface and uses standard Epanet networks to simulate the system hydraulics. The first step in using Mocasim is to build the network model in Mocasim II using the standard Epanet functions.

The basic steps for building the network in Epanet (or Mocasim II) are listed below. For step-by-step instructions, consult Chapter 2 of the Epanet Users Manual or do the tutorial under the **Help** menu (up to and including the section Single Period Analysis").

It is important at this stage to understand how Mocasim II uses the Epanet simulation model. In stochastic analysis, a network simulation is repeated a large number of times with different stochastically selected input parameters. The hydraulic network built in this and the previous sections is used as the unit or base simulation in the process that is repeated.

The base network simulation is normally a single day (24 hour) extended-period simulation. Mocasim II is flexible in this regard and a longer or shorter simulation run can also be used. However, it is strongly advised that only experienced users make use of this facility, as sound judgement and experience are required to design the analysis and interpret the results.

The first step is to build the network as you would normally do in Epanet, using the mouse and the buttons contained on the Map Toolbar shown below in Figure 4.2.

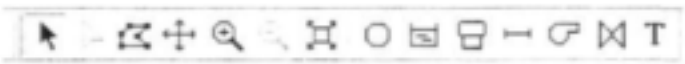

Figure 4.2 Map toolbar

The network layout and ID labels are shown in Figure 4.1.

Set the properties for each object in the network using the Property Editor shown in Figure 4.3. The properties for the nodes and links can be found in Tables 4.1 and 4.2 respectively. Don't forget to set the pump curve to deliver a head of 46 m at a flow rate of 40 l/s.

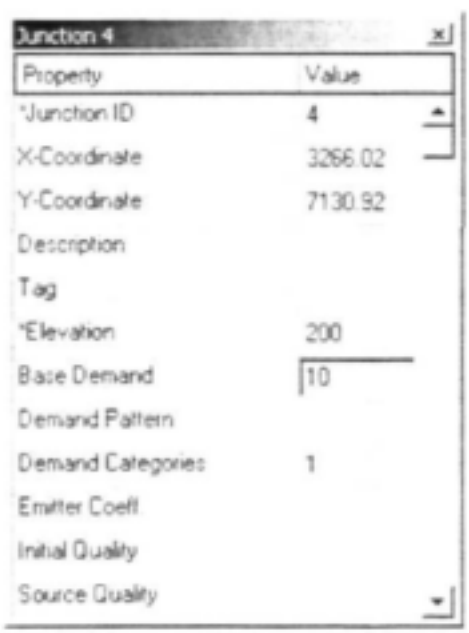

Figure 4.3 Property Editor

4.5 Analysis Options

The example system will be set up for a simulation period of 24 hours using one hour simulation time steps. To implement these options, open the Options-Times property editor from the Data Browser (See Figure 4.4) Set the Total Duration to 24 hours and the Hydraulic Time Step to 1 hour. Another important analysis option to set is the Pattern Time Step. This is the time step used for demand (and other) patterns in the daily

simulation run. It is also used by Mocasim II as basis for calculating network statistics. In other words, if a Pattern Time Step of two hours is used, Mocasim II will count the number of two hour periods that (say) the pressure a given node is within a certain range. Set the Pattern Time Step property to two hours.

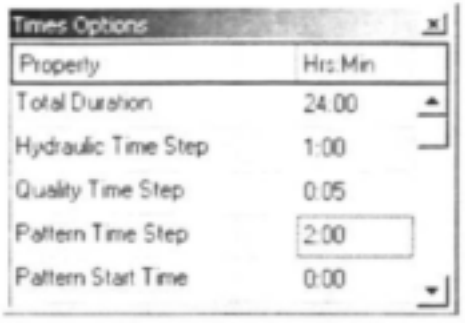

Figure 4.4 Times Options

4.6 Stochastic Demand Models

Now that the base model has been built, it is necessary to set the various stochastic parameters. Three types of stochastic behaviour can be modelled in Mocasim II, namely demands, pipe failures and fires. In the example network, stochastic demands and pipe failures will be implemented.

Stochastic demands are implemented by first building a stochastic demand model and then linking junction nodes to the model. Different nodes can use the same demand model, although a separate demand model may also be built for each demand node.

To create a new demand model, click the Add button in the Demand Models item in the Data Browser. The Demand Model properties are set in the dialog box shown in Figure 4.5. The demand model requires a unique name or ID. An important point to note is that a demand model is not allowed to have the same ID as another demand model or any pattern in the system. This is because demand models are implemented in Mocasim II by creating new patterns with the same IDs as the demand models. Set the name of the demand model to DemandM1.

Demand models in Mocasim II allow the user to set Seasonal, Day-of-Week and Time Step demand patterns. These demand pattern are used to vary the demand throughout the year, with the day of the week and throughout a day (diurnal pattern) respectively.

The Time Step pattern is not called the daily or diurnal pattern, since the user is allowed to do longer or shorter base simulation runs (this is not advised unless you are an experienced user). However, for the purposes of this example the Time Step pattern is the same as a daily demand pattern.

Enter the IDs of Epanet patterns in the three pattern edit boxes. Use the IDs 'Seasonal', 'DOW 'and 'Daily'. These patterns have not yet been defined, but this will be done later.

The next option to set is the Auto-correlation coefficient. This is an indication of how much the current day's demand is determined by the previous day's demand. Set the auto-correlation coefficient to 0.53. This is a high correlation coefficient and means that 53 % of the current day's demand is determined by the previous day's demand.

Finally the daily random variation in the demand has to be set. This is done by first selecting the random component's distribution. Distributions that can be used are uniform, normal and lognormal. If no random component is used, this option is set to none. The last two fields set the parameters of the random distribution. For most distributions this is represented by the average and standard deviation, but for uniform distributions, the minimum and maximum values are selected. Set the random component to a normal distribution with an average of 1 and a standard deviation of 30 %.

Figure 4.5 Stochastic Demand Model Property Editor

The patterns used in the demand model must now be created. These three patterns, Seasonal, DOW and Daily are created as normal Epanet patterns, using the pattern editor shown in Figure 4.6. Refer to the Epanet users manual or online help for further assistance on creating Paterns.

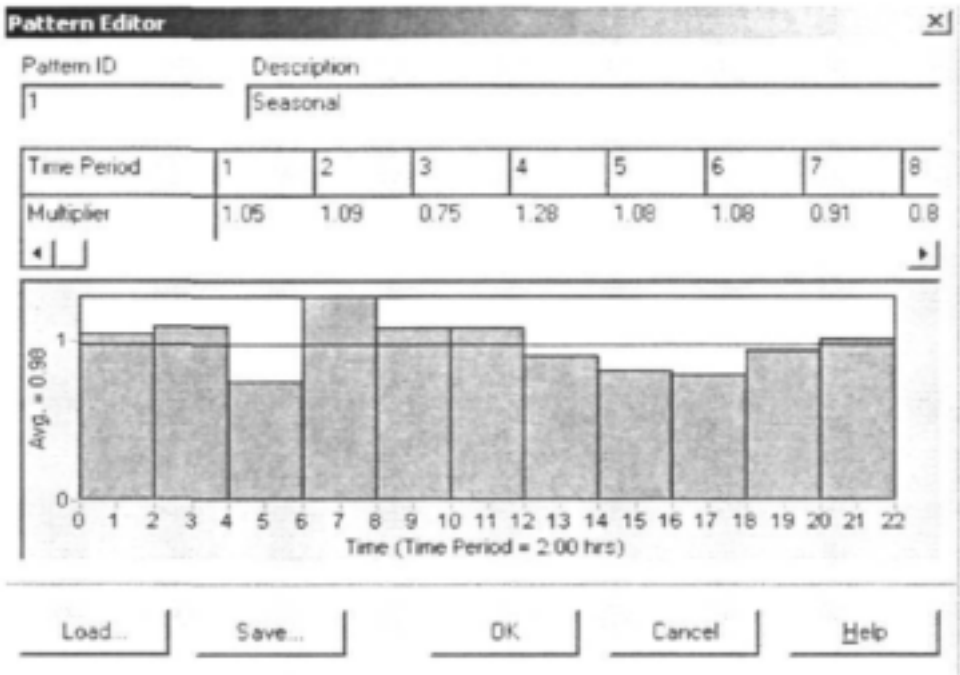

Figure 4.6 Pattern Editor

The values of the three patterns are given in Table 4.3.

Table 4.3 Demand model patterns

Seasonal	1.05	1.09	0.75	1.28	1.08	1.08	0.91	0.81	0.79	0.94	1.01	1.11
Daily	0.29	0.25	0.25	0.47	1.45	2.01	1.78	1.51	1.32	1.22	1.32	0.73
DOW	1.034	1.012	1.015	0.974	1.013	1.077	0.877					

Note that the Daily pattern has 12 factors, as the pattern time step was set to 2 hours and therefore the factor will change 12 times in 24 hours. The Seasonal pattern has 12 factors because, in this example we will use 12 seasons and the DOW pattern has one factor for each day of the week. If the size of the patterns do not correspond to the size specified by other Mocasim II options, an error message will be displayed and the stochastic simulation will be aborted.

4.7 Setting Stochastic Demands

Before the demand model can be implemented, the user must specify which demand nodes should implement the demand model. This is done in the Property Editor of the respective nodes. Double click on the node to which the demand model must be applied, and enter the demand models ID in the field called *Demand Model*. In this application we apply the demand model to all the nodes, therefore the demand model's ID must be set for all the nodes in the network. Figure 4.7 shows the Node property editor with the demand model set.

Figure 4.7 Node Property Editor

Note that, besides the demand model, a normal Epanet demand pattern can also be specified for the node. If a demand model and a demand pattern are specified, only the demand model will be implemented in stochastic analysis. In normal Epanet simulations, however, the demand pattern will take preference.

4.8 Fail Models

Pipe failures are implemented by first building a stochastic pipe fail model and then linking pipes to the model. Different pipes can use the same fail model, although a separate fail model may also be built for each pipe.

To create a new fail model, click the Add button in the Fail Models item in the Data Browser. The Fail Model properties are set in the dialog box shown in Figure 4.8. The fail model requires a unique name or ID. Set the fail model ID to FailM1.

Two distributions have to be set for the fail model, the first to specify the stochastic properties of the time between failures, and the second to specify the failure duration properties. Distributions that can be used are uniform, normal and lognormal. Set the fail spacing distribution to lognormal with an average spacing of 1587 days per kilometre of pipe and a standard deviation of 150 days. Set the failure duration distribution to lognormal with an average duration of 6 hours and a standard deviation of 0.84 hours.

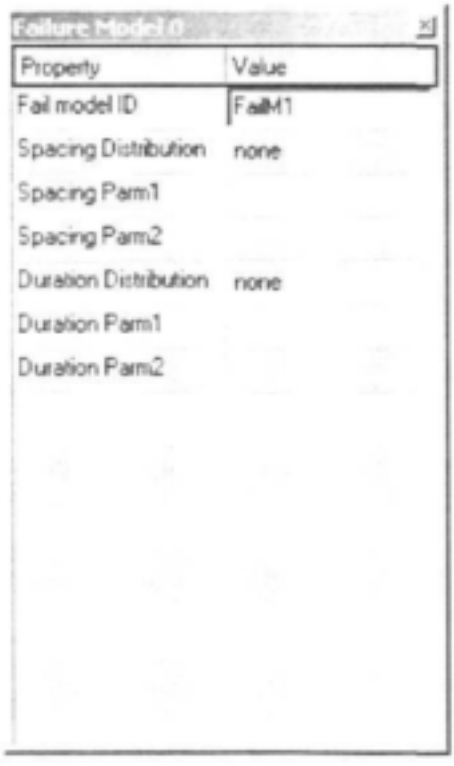

Figure 4.8 Fail Model Property Editor

4.9 Setting Pipe Failures

As for the demand model, the pipes to which the fail model should be applied must be specified. This is done in the Property Editor of the pipe. Double click on the pipe to which the fail model must be applied, and enter the fail model's ID in the field called Fail Model. In this application we apply the fire model to all the pipes, therefore the fail model ID must be set for all pipes in the network. Figure 4.9 shows the property editor of pipe 6 with the fail model set.

Figure 4.9 Property Editor of link 6

4.10 Stochastic Simulation Options

To set the stochastic analysis options, select Options-Stochastic from the Data Browser (See Figure 4.10). Simulation Duration represents the number of base simulations Mocasim II will perform. Since the example simulation was set up for a 24 hour simulation run, this is also equal to the number of days that the simulation will be run for. To perform a 100 year simulation run, set the simulation duration to 36500. Make sure that the Days per Year is set to 365, the Days-per-Week to 7 and the Number of Seasons to 12 to correspond to the size of the patterns set up in the demand model.

Mocasim II can perform a capacity - reliability analysis for one reservoir in the system. In this analysis, the performance of different reservoir capacities will be analysed to determine how they will perform. Specify the ID of the reservoir to analyse in the Reservoir field. Note that in Epanet a distinction is made between reservoirs, for which the hydraulic head is fixed, and tanks, which have varying heads. The reservoir specified here must be of the fixed-head type. Set this field to 8 (the ID of the only fixed-head reservoir in the system).

It is now necessary to specify the range of reservoir capacities to analyse. This is done by specifying a minimum and maximum capacities, and Add and Multiply values to determine the intermediate reservoir sizes. Mocasim II will start by analysing the minimum reservoir size. If it fails during a simulation run, the next reservoir capacity will be determined by multiplying its capacity by the Multiply value and adding the Add value. Enter a minimum size of 2 238 kl, a maximum of 90 000 kl, an add value of 0 and a multiply value of 1.3.

Finally, to ensure that statistics for events (pipe failures) are calculated, set the Event Statistics item to yes. Also note the name of the results file, which Mocasim will to write the results to.

Stochastic Options

Property	Value
Simulation Duration	365
Days per Year	365
Days per Week	7
Seasonal per Year	13
Critical Pressure	10
Reservoir	Source
Min Reservoir Size	2238
Max Reservoir Size	90000
Multiply By	1.3
Add	0
Write Log	No
Log file	molog.txt
Event Statistics	Yes
Results file	mores.txt
Random seed	0

Figure 4.10 Stochastic Options

4.11 Running a Simulation

All the relevant information necessary for the stochastic analysis has now been set. To run the simulation, select Project and Run Stochastic Analysis from the main menu. Mocasim will show the progress while the simulation is running.

4.12 Simulation Results

The simulation results can be viewed by selecting Reports and Stochastic Results from the main menu. A good way to view the results is by plotting the failure behaviour against the hours of AADD stored in the reservoir. Figures 4.11 and 4.12 shows the average number of failures per year and the Average empty time per year for different storage capacities. It can, for instance be seen that a storage capacity of 48 hours will have less than one failure per year and an average fail time of approximately 2 hours per year.

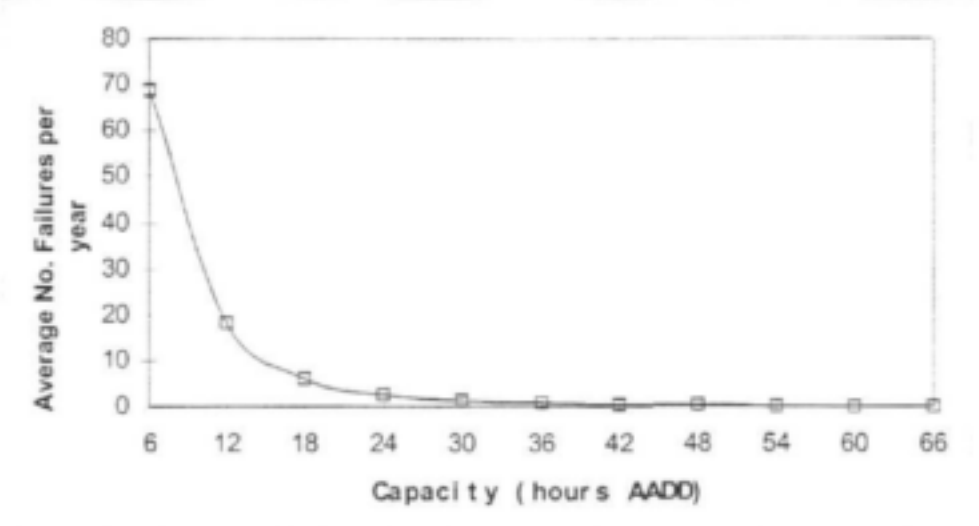

Figure 4.11 Average annual number of failures for service reservoir 1

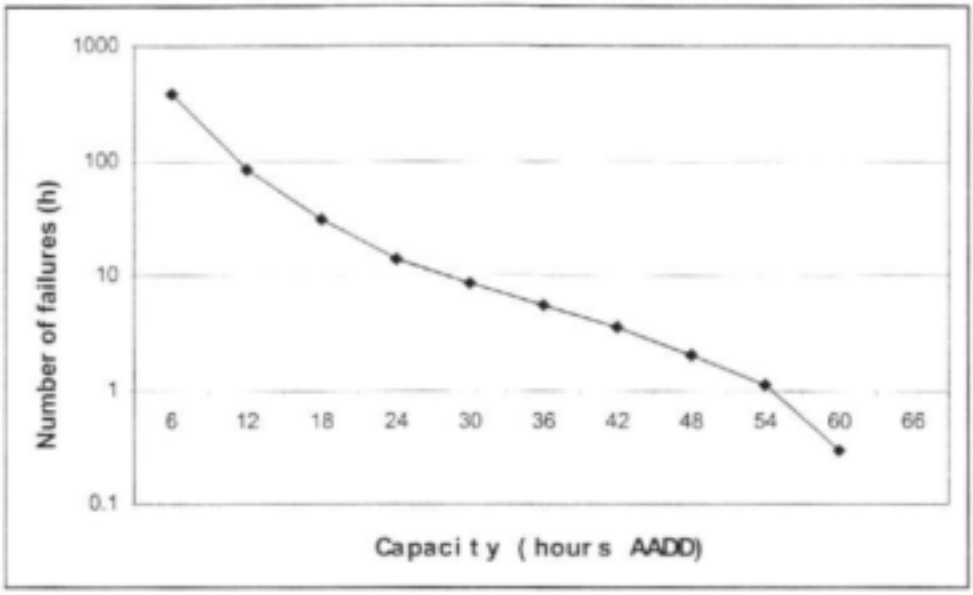

Figure 4.12 Average failure duration for service reservoir 1

CHAPTER FIVE – MOCASIM II METHODOLOGY AND ENVIRONMENT

5.1 Introduction

This section discusses the Mocasim II methodology and environment and can serve as a basic reference for the software.

5.2 Mocasim II methodology

Mocasim II performs stochastic analyses of water distribution systems. It uses the Epanet hydraulic engine to calculate flowrates and pressures in the system and is incorporated into the Epanet user interface. The look and feel of the Epanet user interface was retained as far as possible. All Epanet's functionality was retained unchanged and Mocasim can thus be used for both stochastic simulations and normal Epanet hydraulic simulations. Epanet input files can be opened in, and exported from Mocasim II.

Before a stochastic simulation can be performed, a network model must first be built in Epanet. This network is called the base network. Mocasim does stochastic analysis by repeatedly running the base network model. The model will typically represent a daily simulation run, although this is not a requirement for running Mocasim. Refer to the Epanet Users Manual and online help for assistance on setting up a network and running Epanet simulations.

After building the base model, the Mocasim input data have to be entered. This involves building demand, fail and fire models and linking these models to nodes or links, setting the statistical output parameters for nodes and links and setting the stochastic analysis options. These actions are discussed in more detail in the rest of the chapter.

Mocasim simulations are performed by selecting Project / Run Stochastic Simulation from the main menu. Mocasim performs a number of checks to ensure the integrity of the data before the simulation is run. Certain data errors are critical and will cause the simulation to be aborted. Other potential data errors only generate warnings to the user. Critical errors and warning messages are written to the output file.

Mocasim performs long simulation runs by repeatedly running the base network. Reservoir levels at the beginning of a base run is set to the reservoir levels at the end of the previous base run to ensure continuity.

Demands for a base simulation is calculated in advance from the demand models and implemented using Epanet Patterns. If pipe failures will occur in the next base simulation, Mocasim generates controls to close and open the pipe at the start and end of the failure respectively. If a fire will occur in the next base simulation, the demand pattern for the fire node is adjusted to reflect the fire demand. If the start or end of a fire does not coincide with the start of a new demand factor in the demand pattern (as is normally the case), the fire demand for that pattern step is averaged over the full step.

Mocasim repeats this procedure until the end of the stochastic analysis period is reached. Results are then written to the results file.

5.3 Saving and retrieving networks

Epanet uses two file formats for saving and retrieving networks. The first is a binary file denoted by a *.net* extension and the second is a text file denoted by a *.inp* extension. Mocasim uses the same system with different extensions: *.mcs* is used for binary files and *.inm* for text files.

Mocasim II allows the user to save files as *.net* files and export files to *.inp* Epanet files. Only the Epanet and not the Mocasim input data will be saved to these files.

It is possible to open Mocasim text (*.inm*) files directly in Epanet. A number of warning messages will be displayed when opening the file in Epanet, but the Epanet input data will be read correctly.

5.4 Data input in Mocasim

Mocasim requires various input parameters to be set before a successful stochastic simulation can be performed. These input parameters are discussed and explained in the sections below.

5.4.1 Demand models

Demand models describe both deterministic and probabilistic elements of demand. Demand models are created, deleted and edited from the Data Browser. The following input parameters can be set for Demand Models (Table 5.1):

Table 5.1 Input parameters for demand models

Parameter	Description
Demand Model ID	A unique identifier for the demand model. Can be up to 15 characters long. May not be the same as another demand model or Pattern.
Seasonal Pattern	ID of pattern specifying the seasonal variation in demand. The seasonal pattern must have the same length as the number of seasons specified in Stochastic Options.
Day-of-Week Pattern	ID of pattern specifying the day-of-week variation in demand. The Day-of-Week pattern must have the same length as the number of days in a week specified in Stochastic Options.
Time Step Pattern	ID of pattern specifying the variation in demand in a base simulation run. The length of a pattern step is defined by the Pattern Step variable in Times Options. The number of factor in the Time Step pattern times the Pattern Step length must be equal to the base simulation duration.
Auto correlation	The one-day auto-correlation (or serial correlation) coefficient. It represents the percentage of the average base simulation demand that is determined by the demand in the previous base simulation.
Random Distribution	Statistical distribution of the random component applied to the average base simulation demand. Available options are uniform, normal and lognormal.
Random Parm 1	The average of the random demand component. The default value is 1. For a uniform distribution, the minimum value is specified.
Random Parm2	The standard deviation of the random demand component. For a uniform distribution, the maximum value is specified.

5.4.2 Fail models

Fail models describe the time between failure events and the durations of the failure events. The following input parameters can be set for Fail Models (Table 5.2):

Table 5.2 Input parameters for fail models

Parameter	Description
Fail model ID	A unique identifier for the fail model. Can be up to 15 characters long.
Spacing Distribution	Statistical distribution of the spacing between failure events. Available distributions are uniform, normal and lognormal.
Spacing Parm1	The average spacing between failure events in days / 1000 length units (e.g. days per kilometre if SI units are used). For a uniform distribution, the minimum failure spacing is specified.
Spacing Parm2	The standard deviation of the spacing between failure events in days / 1000 length units (e.g. days per kilometre if SI units are used). For a uniform distribution, the maximum failure spacing is specified.
Duration Distribution	Statistical distribution of the duration of the failure events. Available distributions are uniform, normal and lognormal.
Duration Parm1	The average duration of failure events in hours. For a uniform distribution, the minimum failure duration is specified.
Duration Parm2	The standard deviation of the duration of failure events in hours. For a uniform distribution, the maximum failure duration is specified.

5.4.3 Fire models

Fire models describe the time between fire events, the durations of fire events and the fire demands of the events. The following input parameters can be set for Fire Models (Table 5.3):

Table 5.3 Input parameters for fire models

Parameter	Description
Fire Model ID	A unique identifier for the fire model. Can be up to 15 characters long.
Spacing Distribution	Statistical distribution of the spacing between fire events. Available distributions are uniform, normal and lognormal.
Spacing Parm1	The average spacing between fire events in days. For a uniform distribution, the minimum fire spacing is specified.
Spacing Parm2	The standard deviation of the spacing between fire events in days. For a uniform distribution, the maximum fire spacing is specified.
Duration Distribution	Statistical distribution of the duration of the fire events. Available distributions are uniform, normal and lognormal.
Duration Parm1	The average duration of fire events in hours. For a uniform distribution, the minimum fire duration is specified.
Duration Parm2	The standard deviation of the duration of fire events in hours. For a uniform distribution, the maximum fire duration is specified.
Demand Distribution	Statistical distribution of the fire demand. Available distributions are uniform, normal and lognormal.
Demand Parm1	The average fire demand in the current flow units. For a uniform distribution, the minimum fire demand is specified.
Demand Parm2	The standard deviation of the fire demand in the current flow units. For a uniform distribution, the maximum fire demand is specified.

5.4.4 Junctions

A number of variables were added to junctions. The following stochastic input parameters can be set for junctions (Table 5.4):

Table 5.4 Stochastic input parameters for junctions

Parameter	Description
Demand model	ID of the demand model used for the node. If a demand pattern is also specified for the node, the demand model will take preference in stochastic analyses and the demand pattern in Epanet simulation runs.
Fire model	ID of the fire model used for the node.
Log Parameters	Parameters for which statistical data should be calculated during a stochastic simulation run. Allowable values are pressure, head, demand and demand deficit.

5.4.5 Pipes

A number of variables were added to pipes. The following stochastic input parameters can be set for pipes (Table 5.5):

Table 5.5 Stochastic input parameters for pipes

Parameter	Description
Fail model	ID of the fail model used for the pipe. Since Mocasim uses controls to close and open pipes during failure events, an error will be generated if a user-defined control acts on the pipe.
Log Parameters	Parameters for which statistical data should be calculated during a simulation run. Allowable values are flow, unit headloss and status.

5.4.6 Stochastic options

Various options can be set for stochastic simulation runs. The following stochastic options can be set (Table 5.6):

Table 5.6 Stochastic options for simulation runs

Parameter	Description
Simulation Duration	Number of base simulations to run
Days per Year	Number of days per year. The default is 365.
Days per Week	Number of days per week. The default is 7.
Seasons per Year	Number of seasons per year.
Critical Pressure	The nodal pressure below which demands are considered to be pressuer-dependent.
Reservoir	ID of an Epanet reservoir (not tank) for which a reservoir reliability analysis should be performed.
Minimum Reservoir Size	Minimum reservoir capacity to analyse
Maximum Reservoir Size	Maximum reservoir capacity to analyse
Multiply by	Value to multiply the current largest reservoir by to obtain the next reservoir capacity to analyse. Must be greater than one.
Add	Value to add to current largest reservoir capacity to obtain the next reservoir capacity to analyse. Must be greater than zero.
Write Log	Write a log of events. Possible values are yes and no.
Log File	Log file name.
Event Statistics	Calculate statistics for all events. Possible values are yes and no.
Results File	Results file name.
Random Seed	The value with which the random number generator is initialized. This allows a simulation to be repeated exactly as the previous one.

5.5 Running stochastic simulations

A stochastic simulation is run by selecting Project / Run stochastic simulation from the main menu.

5.6 Viewing results

Two files are written for viewing the simulation results, i.e. a log file and a results file.

5.6.1 Log file

The log file is written if the Write Log option in stochastic options are set to yes. The log file maintains a record of the year, season, day of week, day of year, time, description, duration and flowrate (for fire events) of all events as they occur.

5.6.2 Results file

The results of the simulation are written to a text file that is divided up into sections. The sections in the results file are the following:

- Reliability analysis for the service reservoir
- Node statistics
- Link statistics
- Event statistics

Reliability Analysis

For different capacities of the service reservoir, the following properties are listed.

- The average number of failures in a year
- The average failure duration
- The standard deviation of the failure durations
- The minimum failure duration
- The maximum failure duration

Node Statistics

This section displays the parameter properties of the nodes that where set up under the Log Parameters options. For each log parameter, the following results are listed for the simulation period:

- Number of points, i.e. the number of "measurements" taken
- Average
- Standard deviation
- Minimum
- Maximum
- Bin values and number of points in each bin (if bins were defined)

Link Statistics

This section displays the parameter properties of the nodes that where set up under the Log Parameters options. For each log parameter, the following results are listed for the simulation period:

- Number of points, i.e. the number of "measurements" taken
- Average
- Standard deviation
- Minimum
- Maximum
- Bin values and number of points in each bin (if bins were defined)

Event Statistics

This section displays the statistics for all events that occurred in the system. Events currently include pipe failures and fires. Statistics will be written for the

- The number of days between events
- The duration of the event
- The fire flow in case of fire events.

For each of these, the statistical properties are listed, namely the number of points, average, standard deviation, minimum and maximum.

CHAPTER SIX – EXAMPLE APPLICATIONS

6.1 Introduction

This chapter consists of a number of examples and case studies, chosen to demonstrate the capabilities of Mocasim II as well as provide the user with example applications. Basic network descriptions are provided, and some input and output files are listed in Appendix D.

6.2 Simple Water Supply System

In this section a simple generic water supply system is analysed to show the basic capabilities of Mocasim II, and some of the results that can be obtained from a simulation run. In further sections, the simple system is analysed with changes made to certain modelling parameters to illustrate the effect these parameters have on the system reliability.

The simple system parameters were chosen to represent a typical water supply system, with values and design parameters chosen to be realistic. It is not advisable to generalise results since every water supply system is unique. However, the results obtained from this study provides a general idea of the behaviour of water supply systems and the effect different parameters have on them.

6.2.1 System description

The system is shown in Figure 6.1 and consists of a source feeding a service reservoir, which in turn is supplying a town. The town's annual average daily demand (AADD) is 100 l/s and the capacity of the pipe between the source and reservoir 150 l/s (as specified by the Red Book). Results are shown in terms of hours of AADD storage, and the actual values of the demand and supply capacity is thus not important – but rather the ratio of supply capacity to AADD.

The demand model applied to the town consists of three demand patterns: a seasonal, day-of-week and diurnal pattern with a time step of one hour.

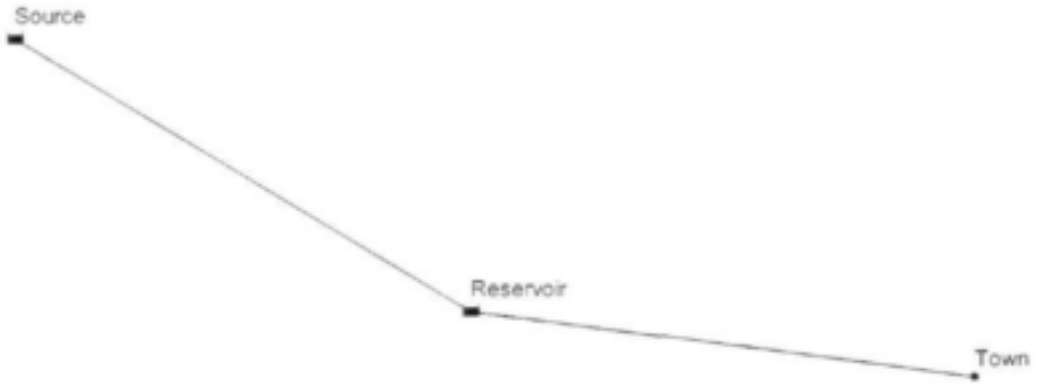

Figure 6.1 Simple System Layout

The feeder pipe's fail model was chosen to produce an average of five failures per year with an average duration of 15 hours. No failures were applied to the pipe between the Reservoir and Town.

The system was simulated for a period of 1000 years. The full description of the system and stochastic parameters may be found in the input file in Appendix D. The resluts file is also included in Appendix D.

6.2.2 Reservoir Reliability results

As mentioned before, the reliability of a service reservoir may be measured using a number of parameters. In this section, the different reservoir reliability parameters for the service reservoir are presented and discussed. The results are shown against hours of AADD storage capacity rather than actual storage capacity to make them more generic. These results will be the same for any system with similar layout, feeder pipe failure distribution and demand behaviour.

The first result that is discussed is the average number of failures per annum shown in Figure 6.2. To show the large range of failures between small and large reservoir capacities, the number of failures are plotted on a logarithmic scale. The figure shows

that a reservoir with capacity of six hours AADD will experience approximately 20 failures per annum, while a reservoir with a capacity of 57 hours AADD will only experience one failure every 100 years. In the capacity range of 24 to 48 hours of AADD, the average number of failures vary between one every year to one every 20 years.

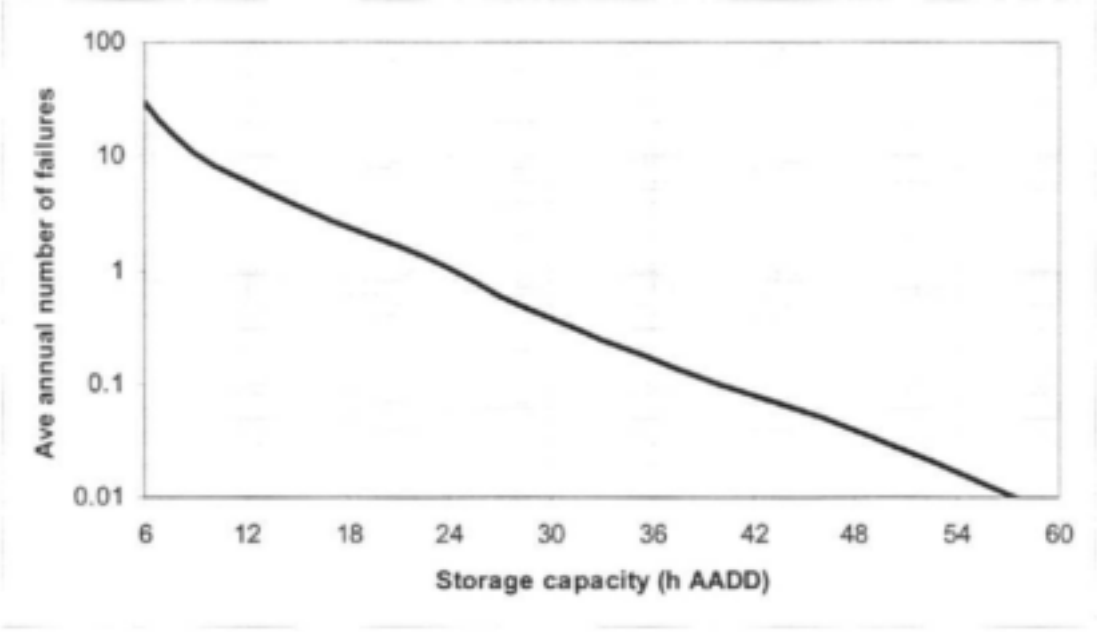

Figure 6.2 Average number of failures per annum for the simple system

Mocasim II also calculates data for the actual failure events. Two parameters are important: the average failure duration and the failure duration standard deviation. These results are shown in Figures 6.3 and 6.4 respectively. It is interesting to note that for reservoir capacities less than 54 hour storage, the reservoir capacity has little effect on the average failure duration, with all capacities experiencing average failure durations of approximately 5 hours. The reason for this behaviour is probably that the long failures experienced by small reservoir capacities are balanced out by many short failures, which brings their averages down.

This explanation is supported by the standard deviation results, which give an indication of the variability in the failure duration data. The standard deviation drops steadily with increasing reservoir capacity, indicating that the failure events are less variable.

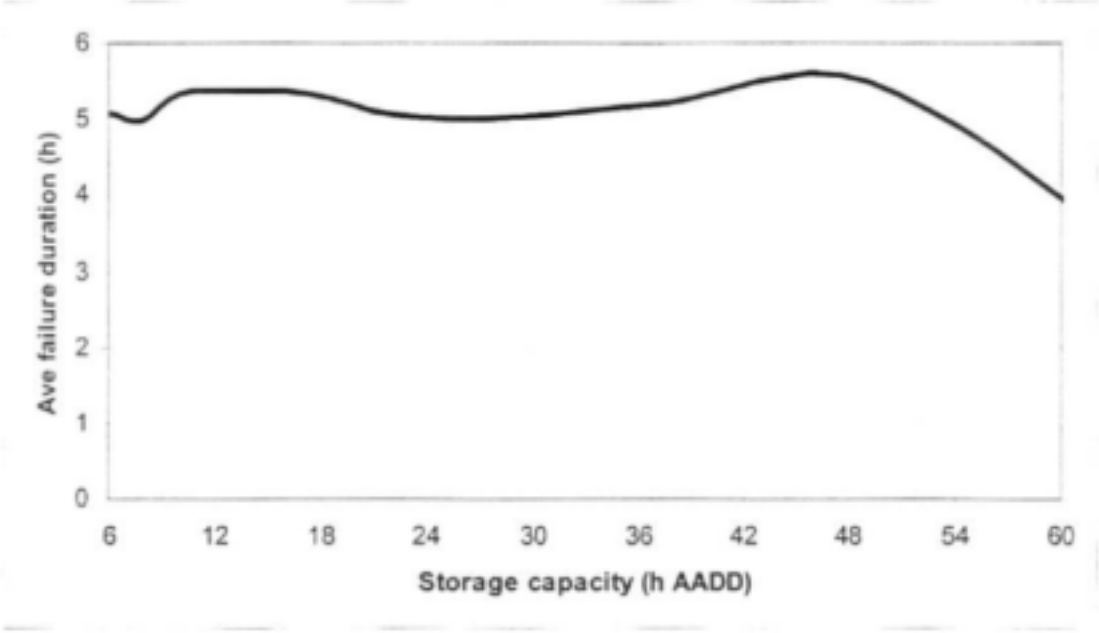

Figure 6.3 Average failure duration for the simple system

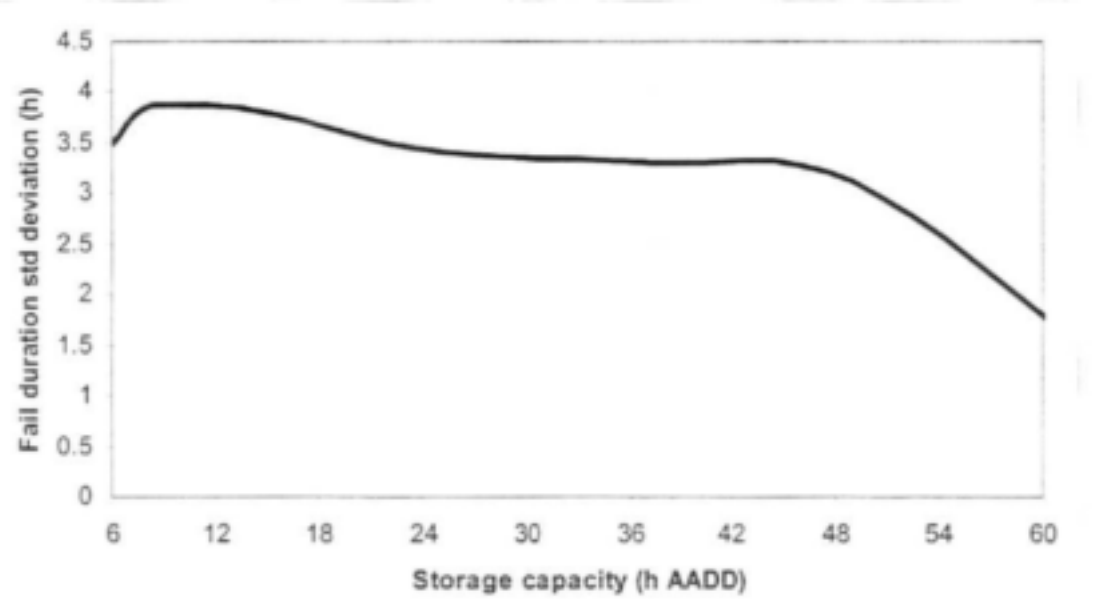

Figure 6.4 Failure duration standard deviation for the simple system

The failure frequency and duration results can be combined to show the average time per year a reservoir with given capacity will be empty. This is shown in Figure 6.5. It is clear that the total reservoir fail time is substantially reduced by increasing the reservoir capacity. For instance, the annual fail time reduces from 5 hours to 0.2 hours (12 minutes) when the reservoir capacity is increased from 24 to 48 hours of AADD.

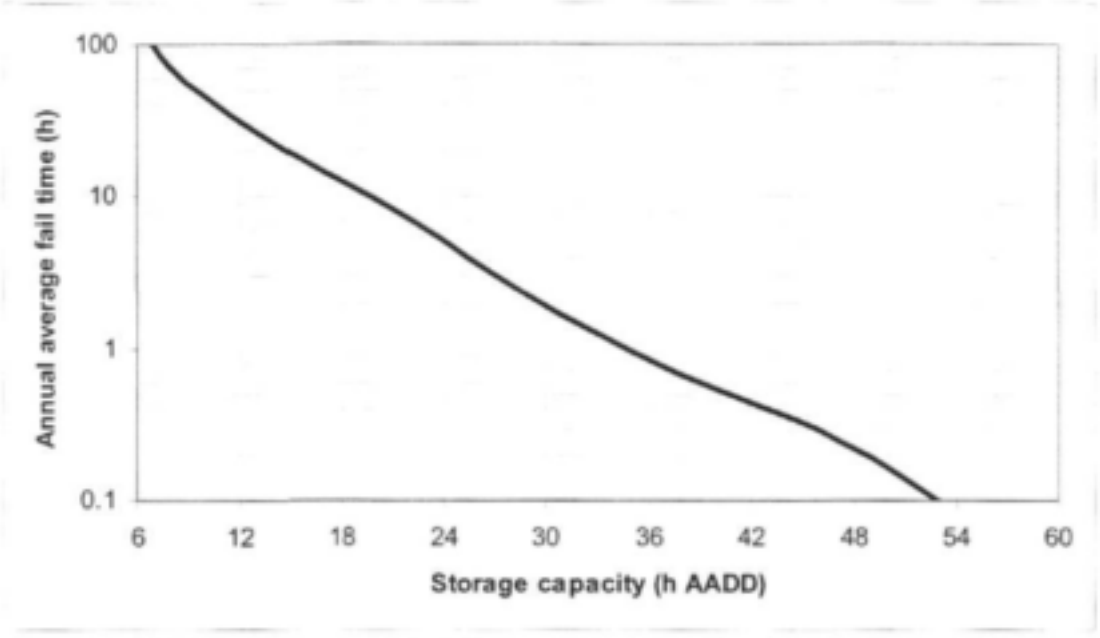

Figure 6.5 Average reservoir fail time per annum for the simple system

A final measure of importance is the maximum failure duration recorded. In some applications, users may for instance accept frequent short failures, but be very sensitive to the maximum duration of a failure event. The maximum failure duration data is shown in Figure 6.6. The figure shows that the maximum failure duration is also substantially affected by the reservoir capacity.

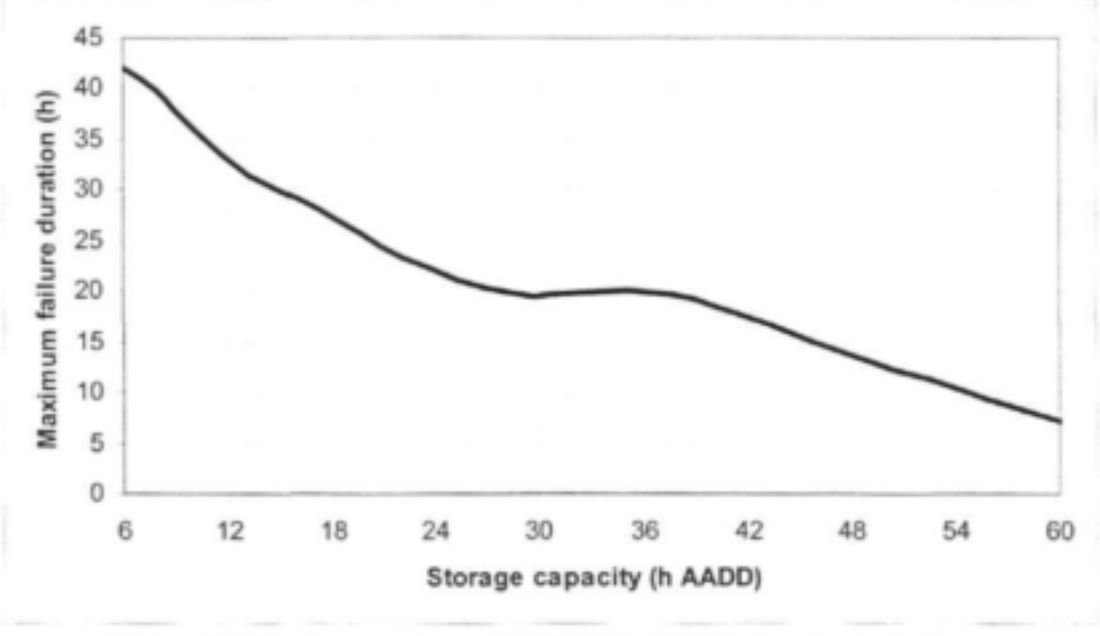

Figure 6.6 Maximum reservoir fail time for the simple system

6.2.3 System performance statistics

Incorporation of the hydraulic model into the stochastic analysis methodology has the advantage that the distribution of various parameters in the system can be sampled and reported statistically. For junctions, the demand, pressure, head and demand deficit can be reported and for pipes the flowrate, unit headloss and status.

The statistics to calculate can be set as options for each junction or pipe, as well as the minimum and maximum values and the number of intermediate values (bins) to log. The results file also gives the number of points and average, standard deviation, minimum and maximum values for all statistics options set. The distributions of demand and pressure were calculated for the Town and are shown in Figures 6.7 and 6.8.

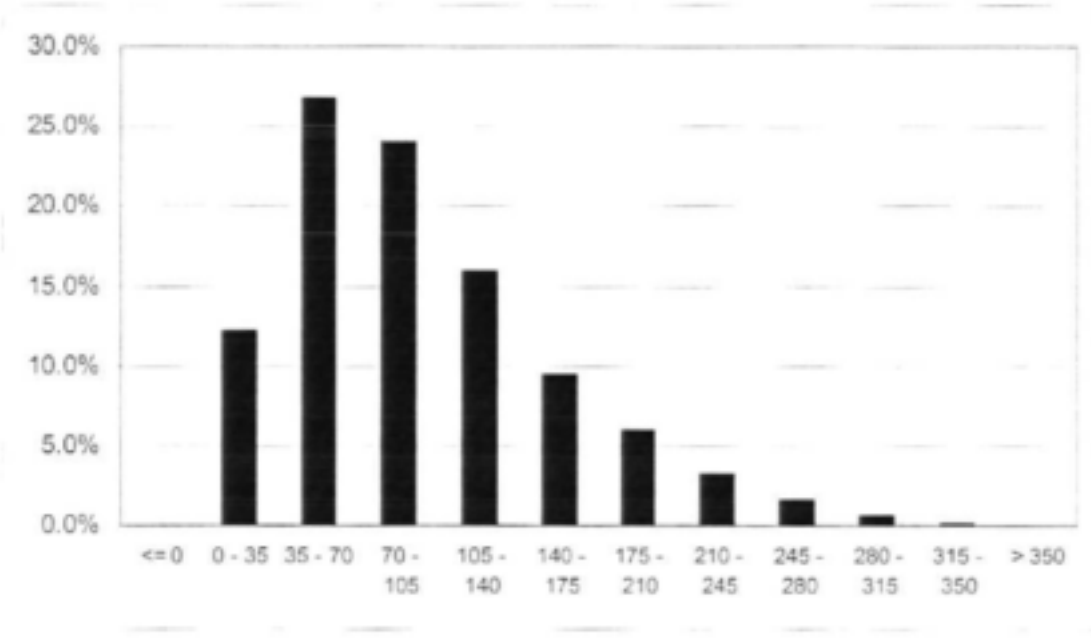

Figure 6.7 Demand distribution for node Town

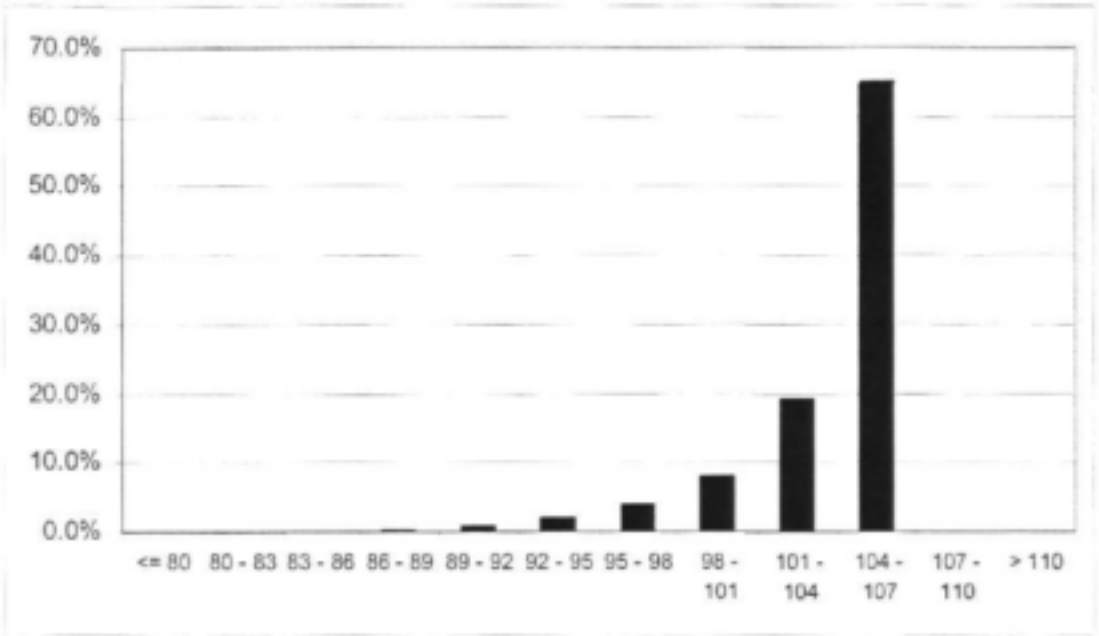

Figure 6.8 Pressure distribution for node Town

6.3 Simple System with Fire Demands

The simple system did not make any provision for fire water. In this section two separate fire models are imposed on the system to investigate the effect of the fires on its reliability. The first fire model represents the Red Book fire demand required for residential areas (Low-Risk Group 1) and the second fire model the Red Book requirements for high density industrial or commercial areas (High Risk). The requirement specified for Low-Risk Group 1 is a fire flow of 15 l/s for a duration of two hours. The requirements for the High Risk category is 200 l/s for 6 hours. Fire frequencies are not specified by the Red Book, but a fire frequency of one fire per day and one fire per month were assumed for the Low-Risk and High-Risk categories respectively. The input files are given in Appendix D.

The results for the number of failures, average failure duration and maximum failure duration are compared in Figures 6.9 to 6.11.

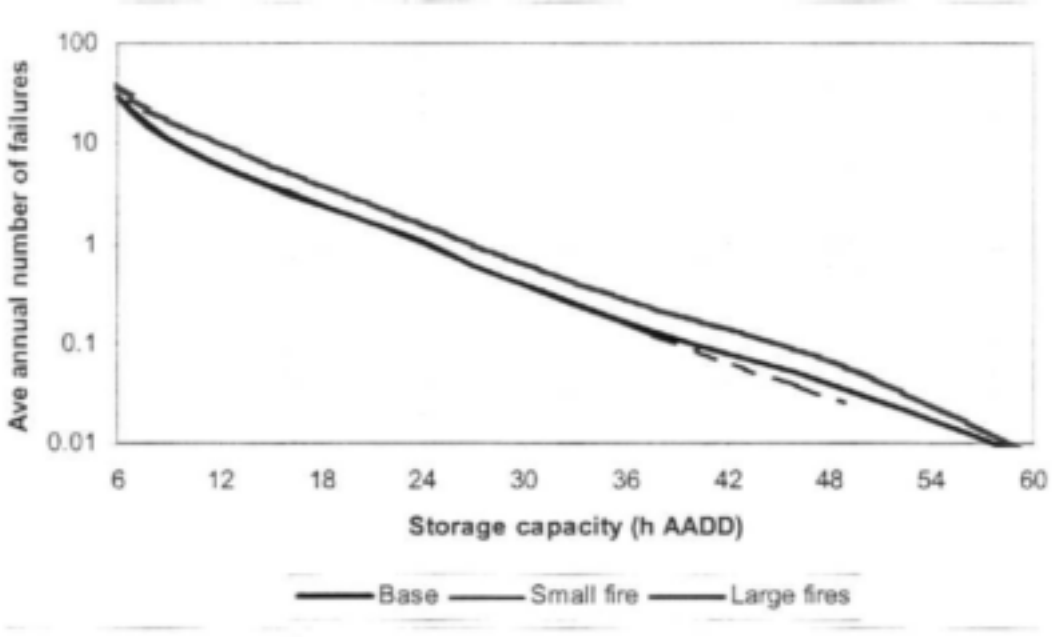

Figure 6.9 Average number of failures per annum for the simple system with different fire models

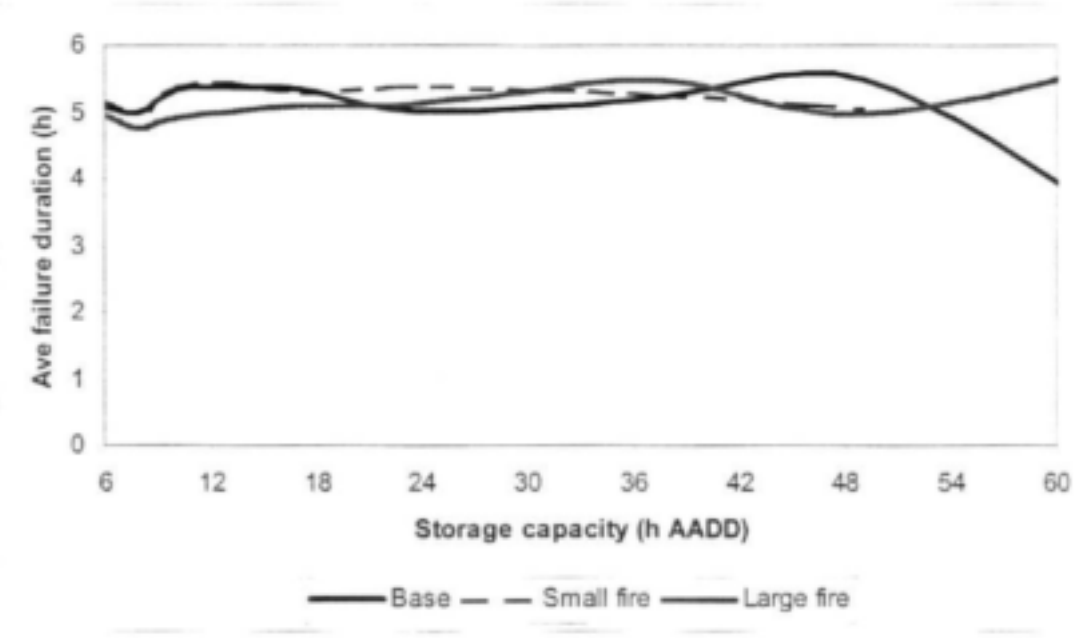

Figure 6.10 Average failure duration for the simple system with different fire models

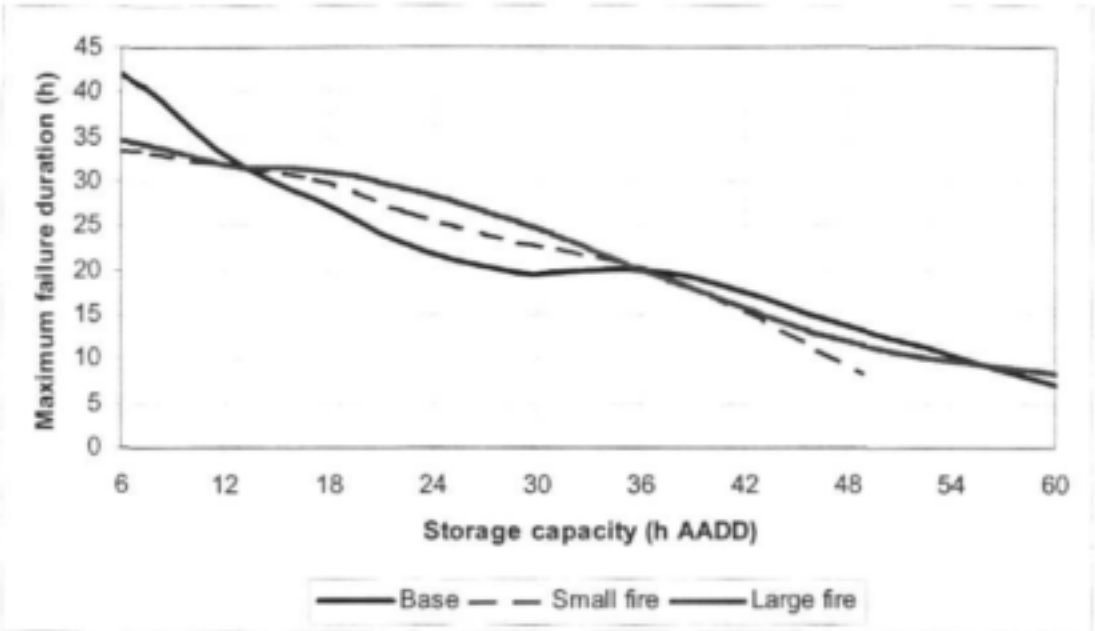

Figure 6.11 Maximum failure duration for the simple system with different fire models

The figures show that there is there is no significant difference in the number of failures for the systems with and without small fires, even if the fires occur at an average frequency of one fire per day. The large fires, however, made a significant difference to the number of failures. To get the same number of failures for the High Risk fire model than for the model without fires, the reservoir's capacity has to increase by three hours AADD, or 2.1 Ml. Although this is a large increase in capacity, it is still less than half of the fire storage capacity required by the Red Book for High Risk areas. Doubt can also be thrown on how realistic the fire flow and duration requirements are for High Risk areas. A study by Van Zyl and Haarhoff (1997) showed that the Red Book requirements are very high compared to both the requirements of other countries and actual fire demands used by fire services in Johannesburg. Using more realistic figures for the fire water requirements in High Risk areas will reduce the required fire storage capacity even further.

It is interesting to note that neither fire model made significant differences to the average or maximum fire durations recorded. Although there are differences, these differences are sometimes positive and other times negative and thus does not convey a clear improvement or reduction in reliability.

6.4 Simple System under low pressure conditions

One of the changes that had to be made to the Epanet source code is the introduction of pressure dependent demands for systems under low pressure conditions. It is known that demand is affected by pressure. This effect is considered negligible under normal pressure conditions. However, under low pressure conditions, the demand will be affected significantly by the system pressure.

Mocasim does normal demand-driven hydraulic simulations when pressures in the system are above a user specified critical pressure. However, when lower pressure occur, the demands at the low pressure nodes are modelled as pressure-dependent demands or emitters. The critical pressure is set as part of the Stochastic Options in Mocasim II.

Emitters demand is calculated using the function

$$q = ch^{\alpha}$$

with q the emitter demand
h the pressure head at the node
c a constant coefficient
α a constant exponent.

The nodal pressure, emitter flowrate and emitter coefficient are calculated by Mocasim II. However, the user can modify the shape of the pressure-demand curve by setting the emitter exponent. This is done under Hydraulics Options in Mocasim II.

In the simple system simulation studied earlier, the pressure in the node Town was always adequate and pressure dependent demands were never required. To illustrate the effect of pressure dependent demands, the elevation of node Town was lifted by 90 m. The critical pressure was set to 10 m and the emitter exponent to 0.5.

The annual average fail time of the system with and without low pressures are compared in Figure 6.12. The figure shows that the reservoir is more reliable under low pressure conditions than under normal pressure conditions. However, this result is misleading. The only reason why less fail time is reported for the low pressure system is because the low pressures effectively reduced the system demand. The users of the system are using less water not because they want to, but because a failure of the system to provide the required flowrates.

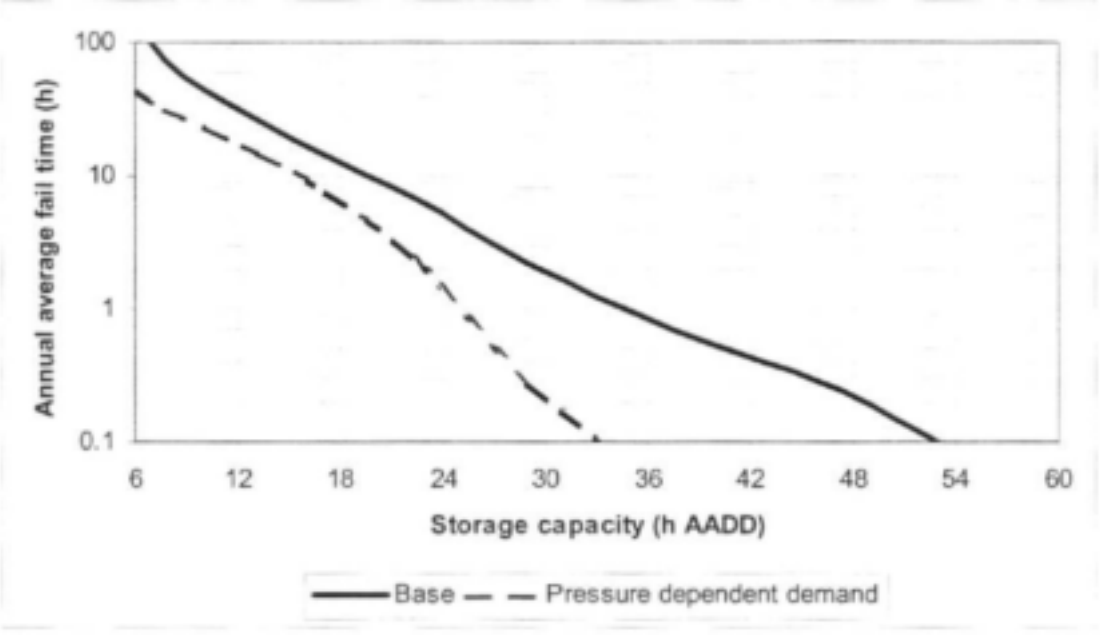

Figure 6.12 Annual average fail time for the simple system under normal and low pressures conditions

The reduced level of service is clear when considering the pressure and demand distributions for the system with low pressure (Figures 6.13 and 14) and comparing them to the distributions for the system with high pressures (Figures 6.7 and 8). The figures clearly show the reduced pressure and the reduction in demand, especially in at the higher end due to the reduced pressures in the system.

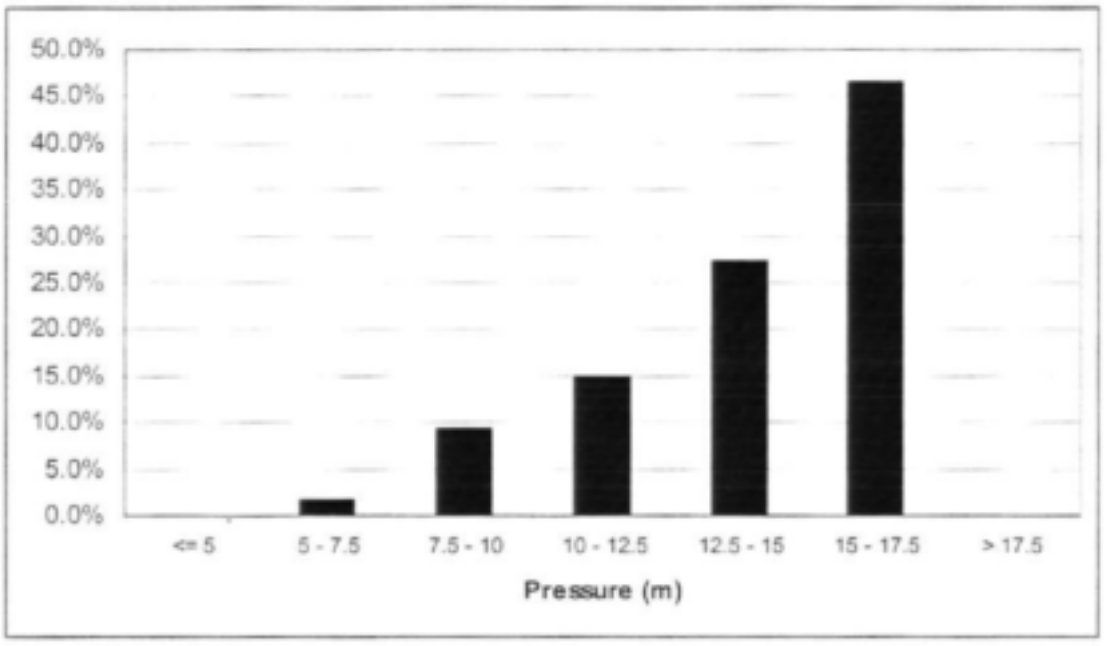

Figure 6.13 Pressure distribution for node Town under low pressure conditions

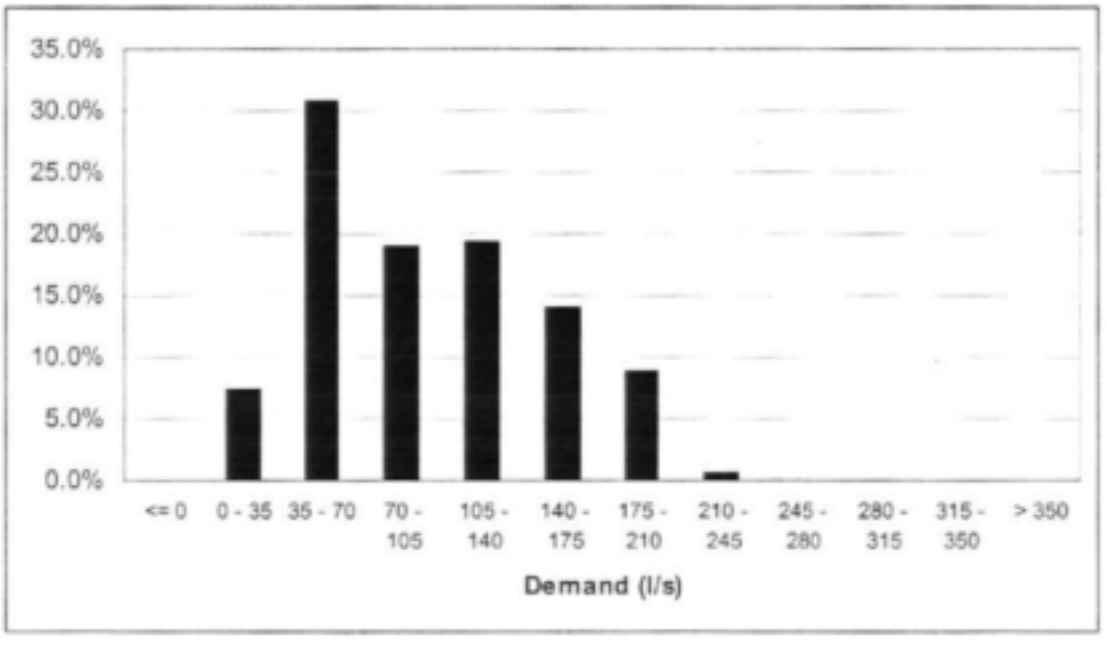

Figure 6.14 Demand distribution for node Town under low pressure conditions

Mocasim II also calculates the demand deficit, which is defined as the difference between the required demand and the actual demand. Under high-pressure conditions, the demand deficit is zero. However, under low-pressure conditions it becomes a significant factor as shown in Figure 6.15. The figure shows that only about 89 % of the required demand was supplied.

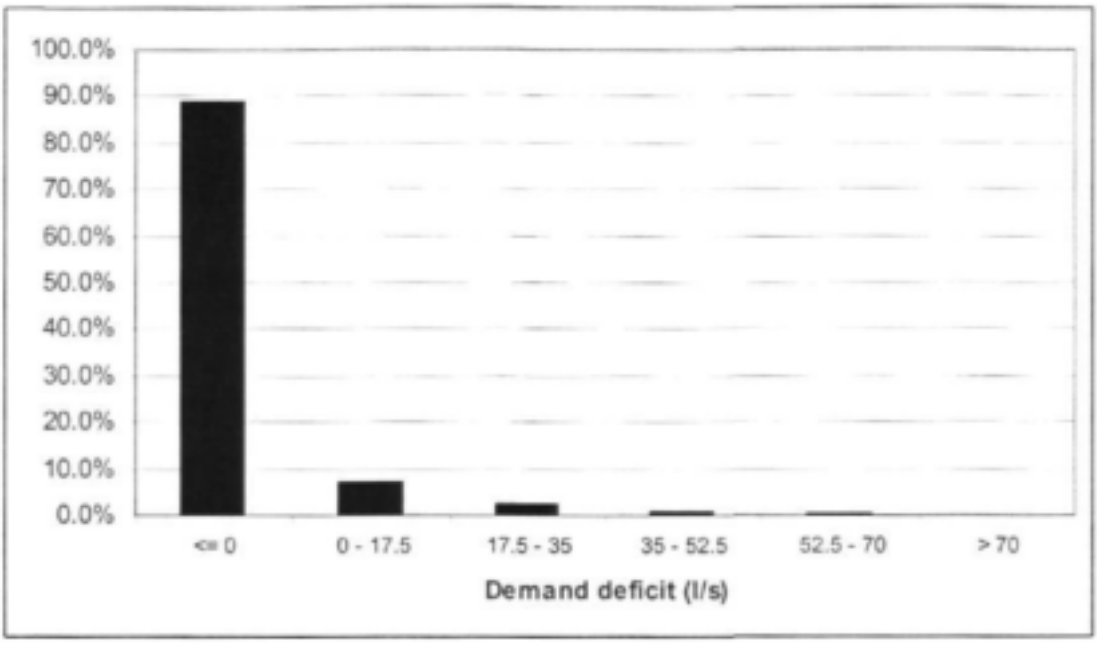

Figure 6.15 Demand distribution for node Town under low pressure conditions

6.5 Urban Network – Windhoek

An analysis was made of a part of the water supply system of the City of Windhoek. The system layout is shown in Figure 6.16.

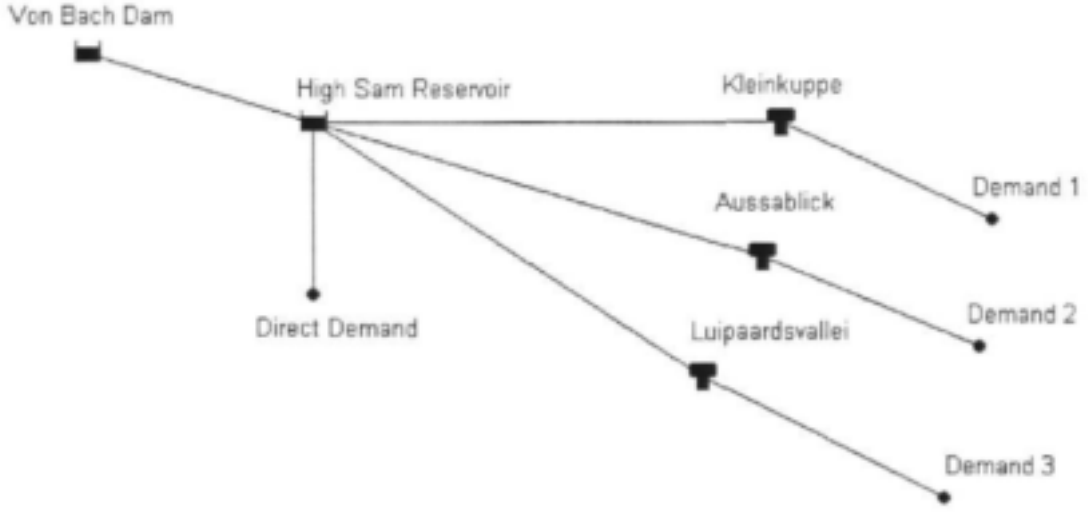

Figure 6.16 Schematic representation of a part of the Windhoek network

A central reservoir (High Sam) is supplied from a distant source (the Von Bach Dam near Okahandja) through a long pipeline. There are four separate off-takes from the High Sam reservoir:

- Direct consumer demand, with an AADD of 6800 m^3/d
- A pipeline feeding the secondary reservoir Aussablick, from which an AADD of 4800 m^3/d is supplied
- A pipeline feeding the secondary reservoir Luipaardsvallei, from which an AADD of 800 m^3/d is supplied
- A pipeline feeding the secondary reservoir Kleinekuppe, from which an AADD of 7800 m^3/d is supplied.

The AADD of the combined area is 20 200 m^3/d and the feeder pipe from the source to the primary reservoir has a capacity of 1.3 times the AADD of the entire area.

This network is an example of a primary reservoir having dual purposes, namely to balance out the fluctuations of its consumers, as well as being a transfer reservoir which must balance out the imbalances between source and end reservoirs.

The emphasis is on the entire four-reservoir system. The actual reservoir volumes were therefore treated as variables to see how they influence the reliability. Three cases were analysed:

- In the first case, the secondary reservoirs were fairly large, namely having volumes equal to 60h of the AADD.
- In the second case, the volumes of the secondary reservoirs were reduced to 48h of the AADD.
- The third case was similar to the second, with the exception that the direct demand from the primary reservoir was removed. In this case, the primary reservoir was thus a pure "transfer" reservoir.

The control was such that the secondary reservoir had first priority to the water in the primary reservoir. In other words, even if the primary reservoir was almost empty, water would be transferred to the secondary reservoirs if there were the slightest shortfall. The net effect of this control philosophy is that consumers of the primary reservoir would

suffer more interruptions than consumers from the secondary reservoirs. The results of the analyses are shown in Figures 6.17 and 18.

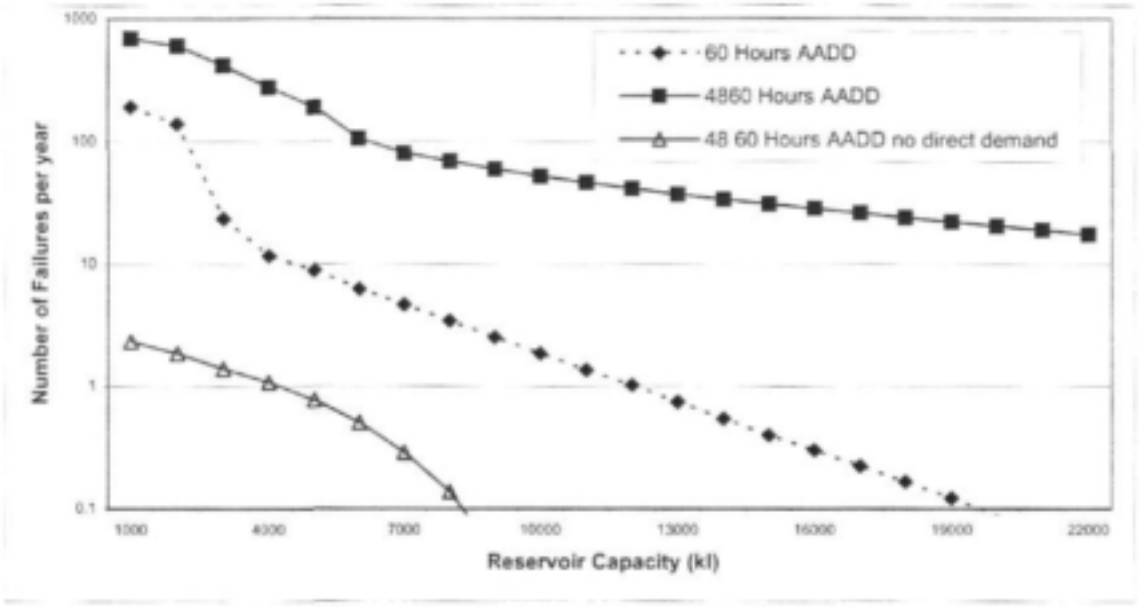

Figure 6.17 Average number of failures per year for the Windhoek reservoir system.

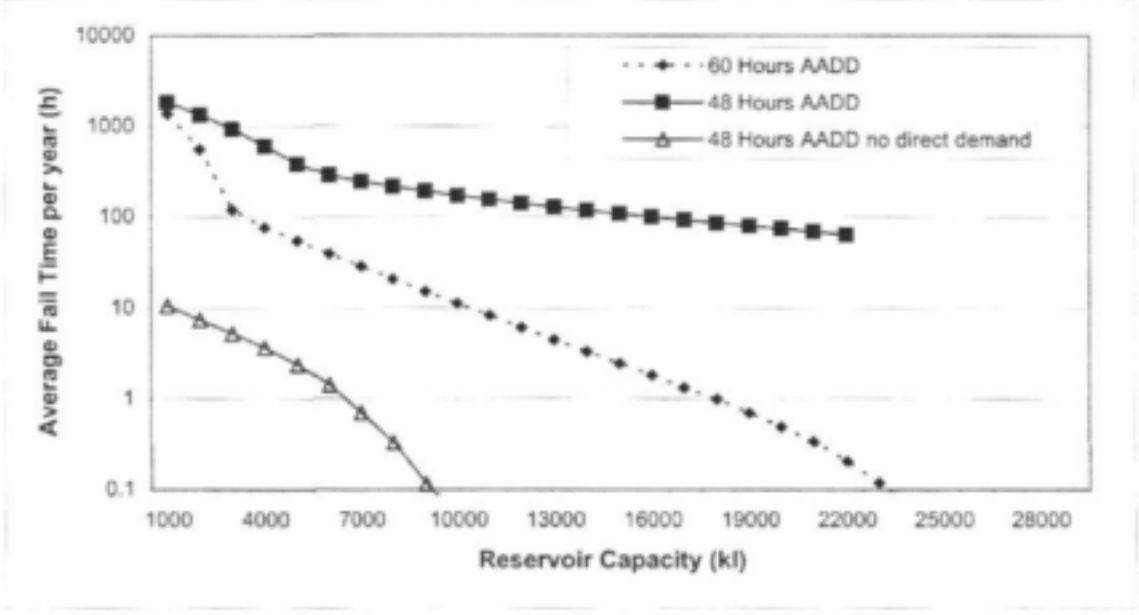

Figure 6.18 Annual average fail time for the Windhoek reservoir system

It can be seen from the figures that increasing the volume of the secondary reservoirs

It can be seen from the figures that increasing the volume of the secondary reservoirs from 48h AADD to 60h AADD has a positive effect on the reliability of the primary reservoir. To keep the total interruption time to about 1 hour per year, the primary reservoir should have a capacity of approximately 18 000 kl which is about 21 hours of the AADD for the entire area.

If the demand from the primary reservoir is ignored, the primary reservoir acts purely as a transfer reservoir. To keep the total interruption time to about 1 hour per year, the primary reservoir volume can be substantially reduded to 7000 kl, which is roughly 8 hours of the AADD for the entire area, even if the secondary reservoirs all have capacities of 48h AADD.

6.6 Evaluating System Performance

Some of the benefits of stochastic analysis are demonstrated using the more complex water supply network shown in Figure 6.19. The network consists of two sources, three reservoirs, 92 nodes and pipes with a total length of approximately 80km. The average demand from the network is 685 l/s. No isolated demand zones exist in the model, and most nodes can be supplied from more than one source or reservoir. This makes for a robust system where a satisfactory level of service can normally be maintained even when one of the reservoirs is out of operation.

The pipes in the system were subjected to failures following a log normal distribution with an average failure spacing of 1587 days/km (or a frequency off 0.23 failures/km/a) and a standard deviation of 0.032 failures per kilometre of pipe per year. Failure durations also followed a log normal distribution with an average of 6 hours and a standard deviation of 0.84 hours. System demands varied according to a diurnal demand pattern, an auto-correlation coefficient of 0.4 and a normally distributed random component. No fires were included in the model. The model was run under seasonal peak demand conditions and would thus give conservative results for the remainder of the year when the seasonal demand factor is reduced. The simulation was run with a time step of 1 hour for a duration of 100 years.

Initial simulation runs showed that both Tanks 1 and 2 under-performed and had unacceptably high numbers of failures. To improve their performances, two changes were made to the system. Firstly it was realized that the levels of Tank 1 used to control the operation of the reservoir at the River did not utilize the available capacity of Tank 1: it was switched off when the level of the reservoir reached 5.8 m, while the maximum water depth in the reservoir is 9.8 m. The pump control was adjusted to utilize the full reservoir capacity. Secondly, the volumes of both Tanks 1 and 2 were doubled by increasing their diameters.

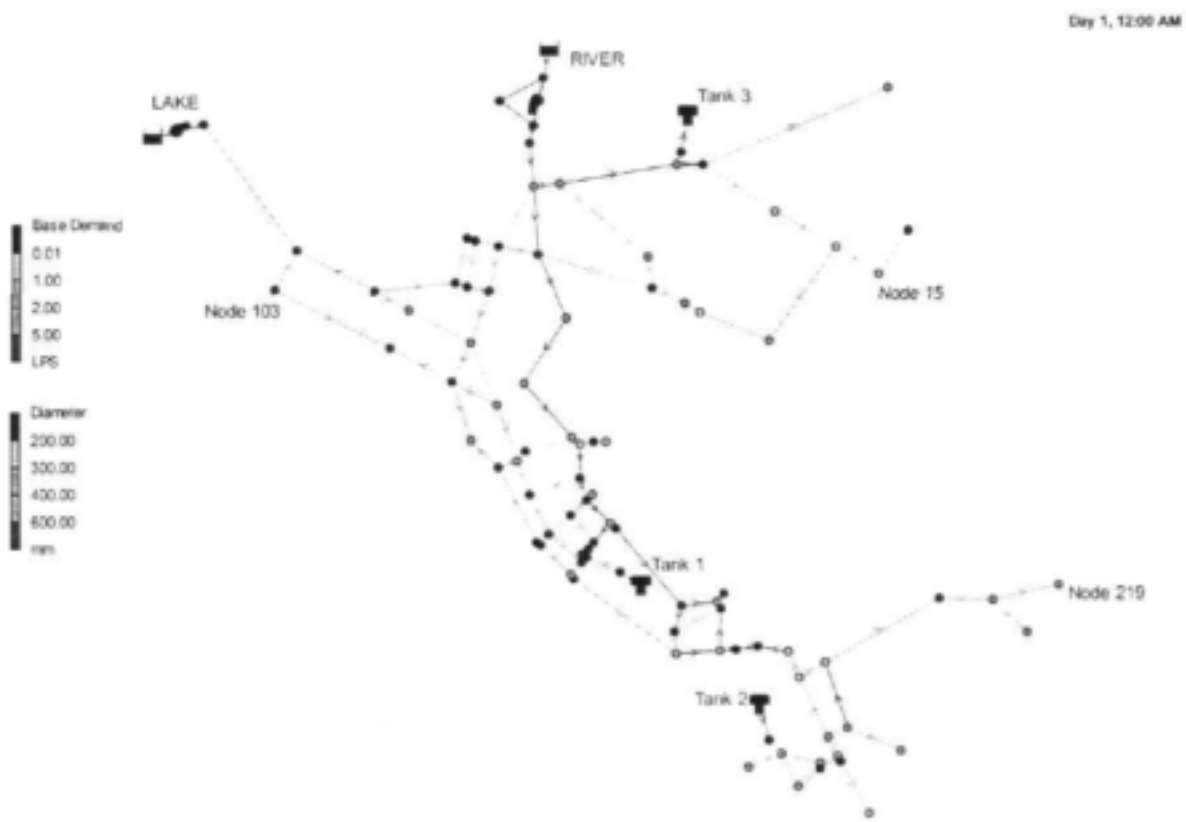

Figure 6.19 System layout

The results of the stochastic simulations before and after the changes are shown in Figures 6.20 to 23. Figures 6.20 and 21 show frequency distributions for the water levels in Tanks 1 and 2 before and after the changes.

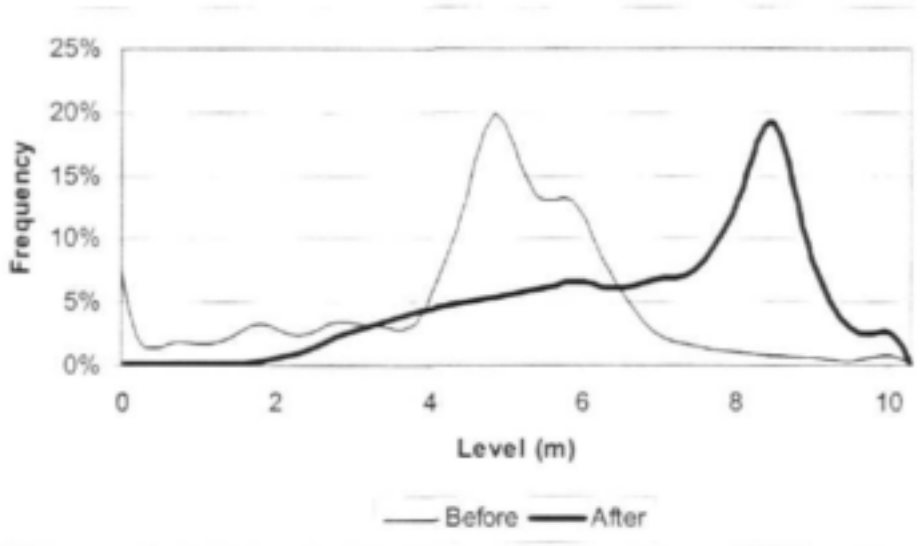

Figure 6.20 Frequency distribution of the water level in Reservoir 1

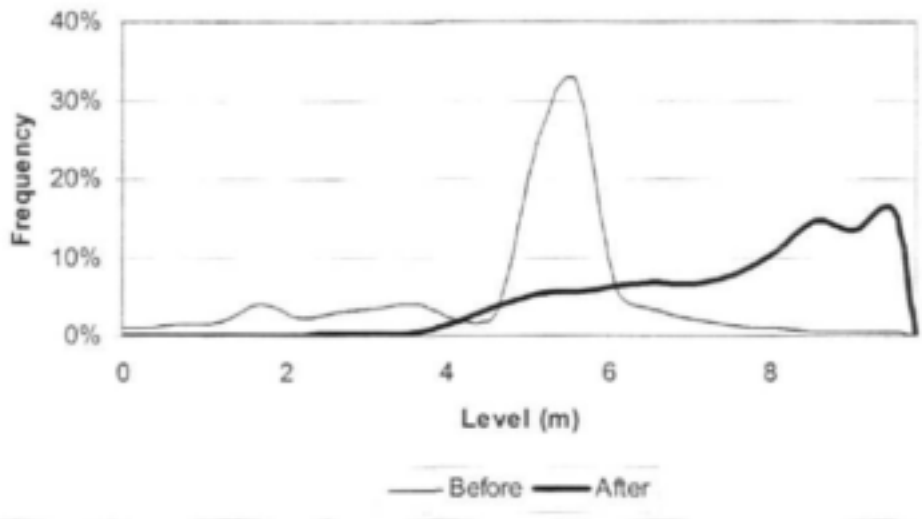

Figure 6.21 Frequency distribution of the water level in Tank 2

It is clear that the performance of both reservoirs were significantly improved by the changes. The average levels in the reservoirs increased, and the amount of time at or near the minimum levels were drastically reduced.

The effect of the changes on the pressures at two nodes, Nodes 103 and 219, are shown in Figures 6.22 and 23. In both cases the pressure was increased by the increases in reservoir levels. Node 103 shows an interesting curve with two distinct peaks. These peaks represent different operational conditions, i.e. gravity supply and pumped supply to the node.

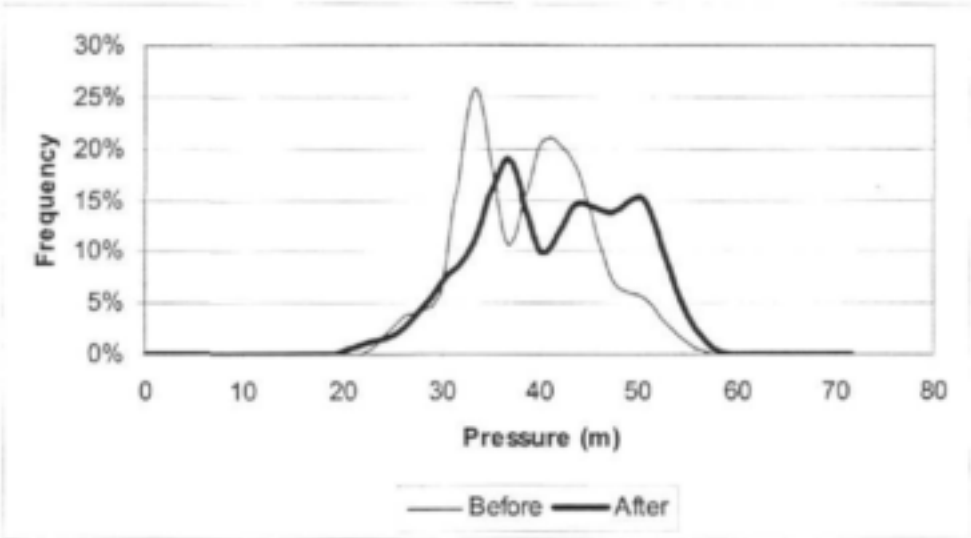

Figure 6.22 Frequency distribution of the pressure at Node 103

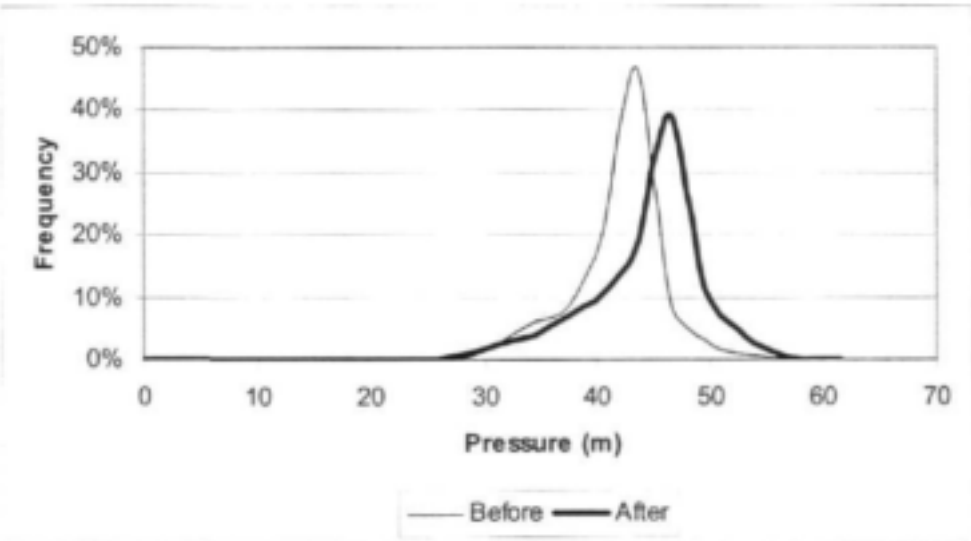

Figure 6.23 Frequency distribution of the pressure at Node 219

The pressure frequency distribution curves allow the user to evaluate the level of service to the users at the node. If, for instance, the minimum pressure requirement at a node is 24 m, it can be determined from the figures that this pressure was not supplied to Node 103 approximately 1 % of the time. While improving the average pressure at the node, it is interesting to note that the changes to the system had little effect on the compliance to the minimum pressure – in fact it had a small negative effect.

Planned and emergency maintenance make it impossible for any water supplier to adhere to minimum level of service requirements 100 % of the time. Stochastic analysis provides a tool to estimate the level of compliance for a given system layout and operational conditions. Stochastic analysis can thus be a valuable aid in designing better systems and showing compliance to level of service standards.

In the example above it can be seen that the capacity of a reservoir has a substantial effect on its reliability (i.e. the probability of it not running empty). Stochastic analysis can also be used to quantify this relationship for a given reservoir in the system. Figure 6.24 shows the relationship between the reservoir capacity and the average number of failures per annum for Tank 2. It is clear that the number of failures initially reduces drastically with increasing reservoir size and then levels out to provide less benefit for the same increase in capacity. The acceptable level of reliability for the system is not always easy to determine, but stochastic analysis provides a tool for calculating the relationship based on which a rational decision can be made, taking both risk and capital cost into account.

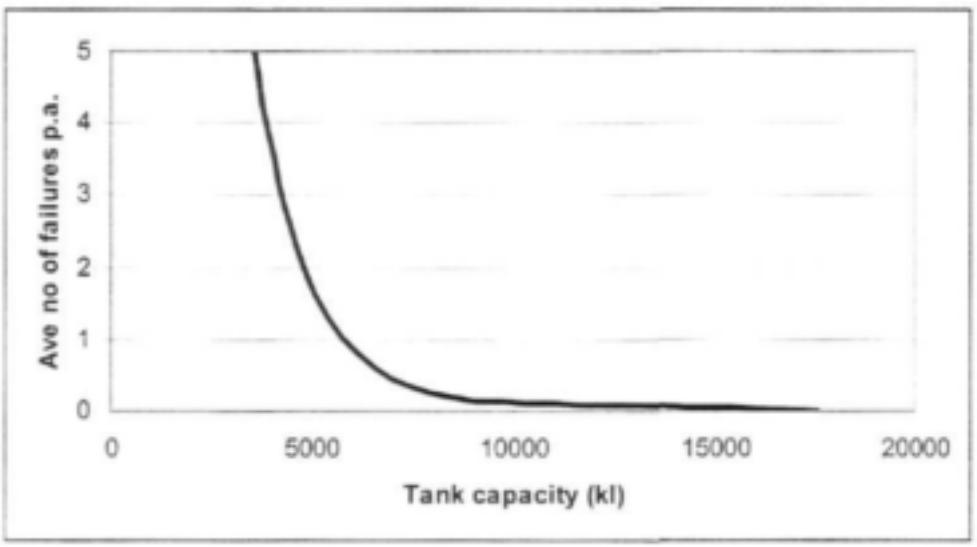

Figure 6.24 Average number of failures per annum vs. the capacity of Tank 2

APPENDIX A – MOCASIM INPUT FILE FORMAT

A.1 Introduction

In the same way as Epanet, Mocasim II can also be run from an input file, and the results written to a text file. The input file is organised into sections, where each section begins with a keyword enclosed in brackets. Example input files for various systems discussed in the workbook appear in Appendix D.

A.2 Input File Headings

The Mocasim II input file consists of two parts. The first part is the normal Epanet input file for the system being modelled and ends with the heading [END]. The Mocasim II input parameters form the second part of the input file and ends with the heading [MSEND].

The input file is organised into various sections using standard headings. The headings are listed in Table A.1.

Table A.1 Input file headings

Network Components	System Operation	Water Quality	Options and Reporting	Network Map/Tags	Stochastic Properties
[TITLE]	[CURVES]	[QUALITY]	[OPTIONS]	[COORDINATES]	[MSDEMANDMODELS]
[JUNCTIONS]	[PATTERNS]	[REACTIONS]	[TIMES]	[VERTICES]	[MSFAILMODELS]
[RESERVOIRS]	[ENERGY]	[SOURCES]	[REPORT]	[LABELS]	[MSFIREMODELS]
[TANKS]	[STATUS]	[MIXING]		[BACKDROP]	[MSDEMANDS]
[PIPES]	[CONTROLS]			[TAGS]	[MSFAILURES]
[PUMPS]	[RULES]				[MSFIRES]
[VALVES]	[DEMANDS]				[MSSTATISTICS]
[EMITTERS]					[MSOPTIONS]

The first five columns in Table A.1 are the normal Epanet keywords, and the last column lists the Mocasim keywords. The order in which the sections are listed is not important, except that the Epanet and Mocasim parts should remain separated.

Each section can contain one or more lines of data. Blank lines can appear anywhere in the file and the semicolon (;) is used to indicate that the rest of the line should be ignored. This is useful for adding comments to the input file.

A maximum of 255 characters can appear on a line. The ID labels used to identify nodes, links, curves and patterns can be any combination of up to 15 characters and numbers.

In the rest of this section, the contents and formats of each Mocasim II section is described. For more information on the Epanet keywords, the user is referred to the Epanet users manual.

A.3 Mocasim input file sections

[MSDEMANDMODELS]

Purpose:

Defines a set of properties to describe the demand at a node.

Format:

One line for each demand model containing:

- Demand model ID label: provides unique identification for each of the demand models
- Seasonal pattern: pattern defining the variation in demand over a number of seasons
- Day-of-the-week pattern: pattern defining the variation in demand over week
- Time-step pattern: pattern defining the variation in demand over whatever time-step is specified during a day, usually one hour
- Auto correlation coefficient: factor according to which each days demand is affected by the previous days demand
- Random component distribution: defines a distribution according to which the random element in the demand can be determined
- Random component average: the average value of the random component distribution
- Random component standard deviation: the standard deviation of the random component distribution

[MSFAILMODELS]

Purpose:

Defines a set of properties to describe the failure behaviour of a link.

Format:

- Fail model ID label: provides unique identification for each of the fail models
- Frequency distribution: defines a distribution according to which the frequency of failures can be determined
- Average of frequency distribution: the average value of the frequency distribution
- Standard deviation of frequency distribution: the standard deviation of the frequency distribution
- Duration distribution: defines a distribution according to which the duration of a failure can be determined
- Average of duration distribution: the average value of the duration distribution
- Standard deviation of duration distribution: the standard deviation of the duration distribution

[MSFIREMODELS]

Purpose:

Defines a set of properties that describe the frequency and duration of fires as well as the required flow rate for fires.

Format:

- Fire model ID label: provides unique identification for each of the fire models
- Frequency distribution: defines a distribution according to which the frequency of fires can be determined
- Average of frequency distribution: the average value of the frequency distribution
- Standard deviation of frequency distribution: the standard deviation of the frequency distribution

- Duration distribution: defines a distribution according to which the duration of a fire can be determined
- Average of duration distribution: the average value of the duration distribution
- Standard deviation of duration distribution: the standard deviation of the duration distribution
- Fire flow: the required flow for fire-fighting

[MSDEMANDS]

Purpose:

Associates the demand models to the appropriate nodes.

Format:

Each node that requires a demand model must be specified with the appropriate demand model in the next column

[MSFAILURES]

Purpose:

Associates the failure models to the appropriate links.

Format:

Each link that requires a fail model must be specified with the appropriate fail model in the next column

[MSFIRES]

Purpose:

Associates the fire models to the appropriate nodes.

Format:

Each node that requires a fire model must be specified with the appropriate fire model in the next column

[MSOPTIONS]

Purpose:

Defines the simulation properties.

Format:

Eleven simulations parameters, each on their own line, in the following order:

- Duration: specifies the number of days the simulation must run for
- Days per year: specifies the number of days in a year
- Days per week: specifies the number of days in a week
- Seasons per year: specifies the number of seasons in a year
- Critical pressure: the lowest pressure allowed at a node
- Reservoir: the reservoir on which to do the reliability analysis
- Minimum size: the reservoirs minimum allowable capacity
- Maximum size: the reservoirs maximum allowable capacity
- Multiply: the factor by which the reservoir capacity must be multiplied if a failure occurs
- Add: the amount that must be add to the reservoirs capacity if a failure occurs

[MSSTATISTICS]

Purpose:

Specifies the statistics required by the user.

Format:

One line for each object with a single parameter. Each line must contain:

- Type: specifies the type of object the request is for e.g. node or link
- ID: specifies the ID label of the specific object
- Parameter: the required property of the object, e.g. demand, head, pressure, flow rate, status etc.
- Minimum: the minimum value from which to log results
- Maximum: the maximum value form which to log results
- No of bins: the number of categories required for the frequency distribution of the property

APPENDIX B – OBJECT ORIENTED TOOLKIT FOR EPANET (OOTEN)

B.1 Introduction

The Epanet hydraulic modelling package was developed by the United States Environmental Protection Agency to help water utilities to better understand the movement and transformations undergone by water in water distribution systems (Rossman 2000).

Epanet can perform both snapshot (steady-state) and extended-period hydraulic analyses of incompressible flow in pipe networks. It can handle various hydraulic components including reservoirs, tanks, pipes, pumps and control valves. Epanet can also model water quality by simulating the behaviour of chemical constituents in the water distribution system with time. The water quality capabilities of Epanet were extended to allow water age and source tracing analyses to be performed.

The Epanet hydraulic and water quality engine has become the standard for the modelling of water distribution systems and is used by several commercial software packages. Epanet also has its own user interface and is available as a stand-alone software package.

A very useful addition to Epanet was the release of the Epanet Programmers Toolkit. The Epanet Programmers Toolkit is a dynamic link library of functions that allows researchers and developers to customize Epanet's computational engine for their own specific needs. The Epanet Programmers Toolkit can be used to query and set element properties, run hydraulic and water quality simulations and obtain simulation results.

The Epanet Programmers Toolkit is used extensively in research on water distribution modelling as is evident from the numerous papers referring to it. The Toolkit is written in ANSI compatible C using an efficient procedural programming style.

Procedural programming is a style of computer programming that focuses on actions, which are programmed into functions and then into units. A disadvantage of procedural programming is that the code can be difficult to write and maintain, especially when programs become large.

A programming style that addresses this problem and that has gained much popularity is object-oriented programming. Object-oriented programming has a number of advantages over the more traditional programming styles, such as ease of developing, checking, expanding, sharing and maintaining programming code. Water distribution systems are particularly suitable for object-oriented description since they consist of different components (e.g. pipes, junctions and valves) that can be modelled efficiently using objects.

A need was identified for a toolkit that can make the Epanet source code functions available in an object-oriented way. As a result, the Object-Oriented Toolkit for Epanet (Ooten) was developed as an object-oriented alternative to the Epanet Programmers Toolkit. Ooten is available as a free download from http://general.rau.ac.za/civil/wrg/ooten.htm.

This appendix gives an overview of the benefits of object-oriented programming, discusses the Ooten class hierarchy and code, highlights the main differences between Ooten and the Epanet Programmers Toolkit and gives examples to illustrate the application of Ooten.

B.2 Object-oriented programming

In object-oriented programming, code is organized around objects rather than actions and data rather than logic. Historically, computer programs were written as logical procedures that take input data, process them, and produce output. The main programming challenge was considered to be writing the logic, not in defining the data.

Object-oriented programming takes the view that it is possible to break most problems up into smaller, easier to handle components called objects. Each object contains its own members (properties or variables) and methods (functions). Each object can be tested individually and can be used in different programs. The software code for each object is normally encapsulated in a class in C++ or some other object-oriented programming language.

Water distribution systems lend themselves well to object-oriented analysis. It is easy to identify objects, for example pipes, junctions and patterns. A pipe object would have

members such as diameter, length and roughness and methods to manipulate the pipe object, such as to set a new diameter, query the pipe length or calculate the headloss for a given flowrate.

There are many advantages to object-oriented programming, including the following:

- It makes it easier to maintain code. Since each class is self-contained, it is easier to maintain and modify without the risk of causing problems in the rest of the code.

- Since many objects share properties and functionality, it is possible to create a hierarchy of classes in which the members and methods of a base class are inherited by its derived classes. In good programming practice, a derived class is always a special case of the base class. For example, a pipe is a special kind of link, thus the pipe class is derived from the link class in the Ooten class hierarchy. This means that the pipe class automatically has access to most methods of the link class, such as methods for handling start and end nodes. It is also possible to create a new kind of pipe by deriving a new pipe class (say MyPipe) from the Ooten pipe class. The user can then add or modify the behaviour the new pipe class as required, but still have the other methods of the Ooten pipe class available to MyPipe.

- Classes are much easier to share between different programmers and applications than sections of code in the traditional programming style.

- Data hiding makes it possible for a class to hide certain members from other classes. This ensures that the developer can control how data is accessed and changed and avoids problems caused by different sections of code changing data in different ways.

- Polymorphism allows programmers to manipulate abstract collections containing a wide variety of objects, such as an array of links which contains pipes, pumps, valves, etc. It is then possible to call a method, for example as to calculate the headloss for the list, without knowing beforehand what the type of each element is. Derived link classes such as MyPipe in the previous example can be added to the array without having to alter the code that acts on the array.

B.3 Ooten

Object-Oriented Toolkit for Epanet (Ooten) provides many of the Epanet source code functions as methods of relevant objects. Ooten acts as a shell of the Epanet source code in the sense that very little data are duplicated. The project data remain in the Epanet source code, and Ooten calls the relevant Epanet source code functions to make the data and functions available in an object-oriented way.

All the Epanet Programmers Toolkit functions are incorporated into Ooten, but the functionality of Ooten has also been extended to include methods that are not available in the Epanet Programmers Toolkit. The main addition is the capability of Ooten to handle curves.

A generic object hierarchy was developed as basis for Ooten and is shown in Figure B.1. At the root of the class hierarchy is the ONObject class. This class is used to provide functionality used by most other classes in the class hierarchy, such as providing the link to the Epanet source code and certain error handling routines.
Classes modelling the main types of hydraulic objects, ONNode, ONLink, ONPattern, ONCurve and ONControl, are derived from ONObject via the ONElement class. Each of these classes handle functionality that is specific to these types of objects, and further specialised classes are derived from them when necessary.

The classes derived from ONNode are ONJunction and ONTank. Since a reservoir is a specialised type of tank (one with constant head), ONReservoir is derived from ONTank. The classes derived from ONLink are ONPipe, ONPump and ONValve. ONValve is specialised further to model the different types of valves handled by Epanet.

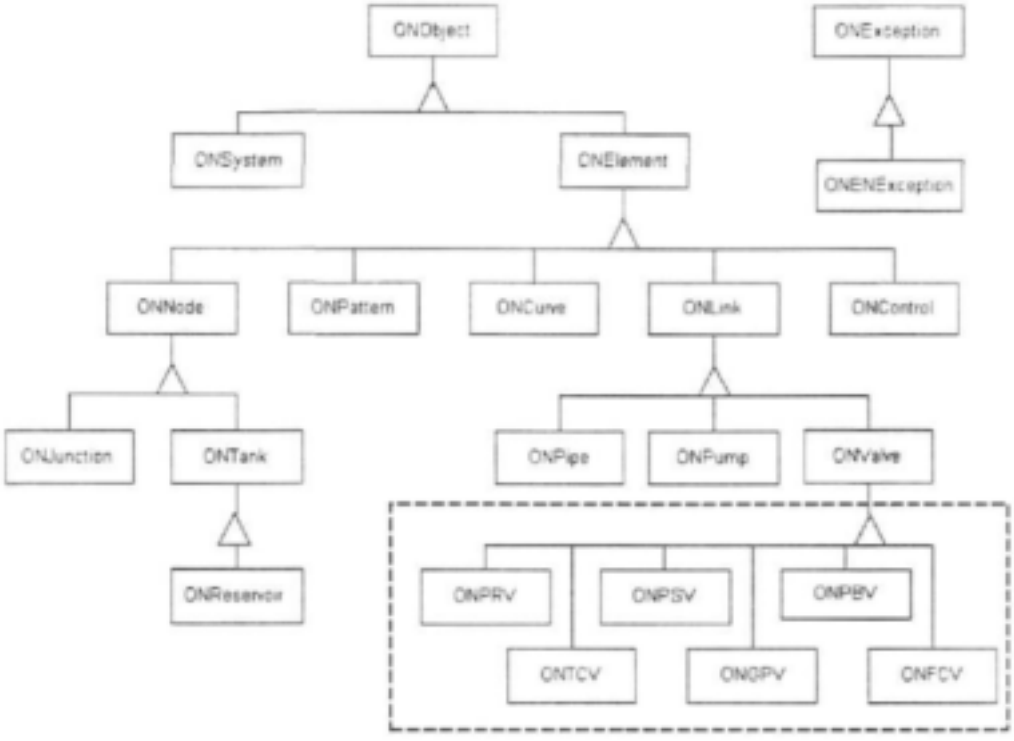

Figure B.1. The Ooten class hierarchy.

The ONSystem class is the main class in Ooten and is used to represent a water distribution system. To open an Epanet input file, an instance of ONSystem is created and the relevant Epanet input file is then opened using the open() method. This creates an object structure representing the network in Ooten and makes these objects available to the user for easy manipulation. Users can change or add to the functionality of objects by deriving their own classes from the Ooten classes.

Errors in Ooten are managed using exception handling. Exception handling provides a type-safe, integrated method for coping with predictable, but unusual conditions (call it errors) that arise while running a program. In C++ an exception is an object, often containing information about the error that occurred, that is created and passed to a part of the code that will then handle the problem.

Ooten checks for errors every time an Epanet source code function is called, and handles these errors by throwing exceptions. This frees the user from the task of error checking, allowing the user to concentrate on the main modelling tasks.

Two exception classes are used in Ooten. The root exception class is ONException, which contains two members: one to describe the error and another to describe the location in the code where the error was detected.

The second exception class is called ONENException and is used exclusively for errors returned by the Epanet source code. The error description is automatically queried by Ooten and stored in the exception object.

A comprehensive help program is provided with Ooten, giving extensive help on the capabilities and use of Ooten.

B.4 Differences between Ooten and the Epanet programmers toolkit

There are five main differences between Ooten and the Epanet Programmers Toolkit: the way functions are called, return value types, index starting values, the way errors are handled and naming of functions.

The most important difference lies in the style of programming used and subsequently in the way the toolkits are applied. The Epanet Programmers Toolkit consists of functions, while Ooten is made up of classes with associated methods. Table B.1 compares the way some common functions are implemented using the Epanet Programmers Toolkit and Ooten.

Table B.1. Some common functions implemented in the Epanet Programmers Toolkit and Ooten.

Function	Epanet implementation	Ooten implementation
Assign a pipe diameter to the variable *diam*	errCode = ENgetlinkvalue (linkIndex, EN_DIAMETER, diam);	diam = aPipe.diameter();
Set the elevation of a node to 100	errCode = ENsetnodevalue (nodeIndex, EN_ELEVATION, 100);	aNode.setElevation(100);
Assign the number of patterns in the system to the variable *num*	errCode = ENgetcount (EN_PATCOUNT, num);	num = aSystem. numberOfPatterns();
Run a hydraulic simulation	errCode = ENsolveH();	aSystem.solveHydraulics();

In some methods, Ooten returns different value types than the Epanet Programmers Toolkit. Ooten generally returns double instead of float value types. This is done since many programmers routinely use double instead of float, but does not improve the accuracy of the results.

Another difference in the return value types is where objects return references to other objects, for instance when a link is asked for its starting node. In the Epanet Programmers Toolkit, the Epanet index value of the starting node is returned. However, in Ooten, a pointer to the starting node object is returned.

In the Epanet Programmers Toolkit, index values generally start at 1, thus the index of the first pipe in an Epanet network would be 1. Ooten uses the standard C++ indexing style that starts all index values at 0. In most cases, the Ooten index value will be the Epanet index value minus 1.

In the Epanet Programmers Toolkit, most functions return an error code that should be checked by the user to determine whether or not an error has occurred. In Ooten, most error checking is done automatically. If an error is detected, an exception is thrown which the user can catch at different levels in the code. Exception handling is a simple, but elegant and powerful way of dealing with data and logical errors in software programs and is a part of standard C++.

The final difference between the Epanet Programmers Toolkit and Ooten lies in the fact that Ooten generally uses longer and more descriptive function names than Epanet.

B.5 Examples

Two examples are given to illustrate the application of Ooten. In the first example, a simple snapshot simulation is performed, while the second example illustrates how Ooten can be used to calculate a hydrant rating curve for a node.

Table B.2 lists the Epanet and Ooten programming code to perform a single snapshot simulation. The code is kept simple for illustration purposes. The main programming steps consist of opening an Epanet input file, opening and initialising the Epanet hydraulics system, running a snapshot simulation, closing the hydraulics system and finally handling any errors that might have occurred.

Table B.2. Performing a simple snapshot simulation using the Epanet Programmers Toolkit and Ooten.

Code using the Epanet Programmers Toolkit	Code using Ooten
// Define variables	// Define variables
int err;	try
long t;	{
	ONSystem mySystem;
// Open file "net1.inp"	// Open file "net1.inp"
err = ENopen("net1.inp", "net1.rpt", "");	mySystem.open("net1.inp", "net1.rpt", "");
// Open hydraulics	// Open hydraulics
if (err == 0)	mySystem.openHydraulics();
err = ENopenH();	
// Initialise hydraulics	// Initialise hydraulics
if (err == 0)	mySystem.initializeHydraulics(1.0);
err = ENinitH(1);	
// Run snapshot simulation	// Run snapshot simulation
if (err == 0)	mysystem.runHydraulicSnapshot();
err = ENrunH(&t);	
// Close hydraulics	// Close hydraulics
if (err == 0)	mySystem.closeHydraulics();
err = ENcloseH();	}
// Handle errors	// Handle errors
if (errcode > 0)	catch(ONException e)
{	{
cout << "Error detected",	cout << e.report();
}	}

The first difference that is evident between the programming codes using the Epanet Programmers Toolkit and Ooten is that in the code using Ooten all functions are called as methods of objects. In this example all the methods that are called are members of the class ONSystem. It is thus necessary to first create an instance of the ONSystem class, called mySystem in the example.

The second difference lies in the way errors are handled. In the code using the Epanet Programmers Toolkit, the user has to evaluate the error code each time a Toolkit function is called. In the example, functions are only called if the error code is currently equal to zero, i.e. no error has occurred. At the end of the example, the error code is checked and an error message written if required.

In Ooten, errors are handled using exceptions. The user is not concerned with errors at each step, but simply places the code in a try block. If an error occurs, the rest of the code in the try block is skipped and the error handled in the catch block. The catch block catches errors of types ONException and ONENException (since ONENException is derived from ONException). The report() method of ONException is called, which returns information on the type of error that occurred as well as the error location.

The second example illustrates how Ooten can be used to develop a hydrant rating curve used in fire flow studies. This curve shows the amount of flow available at a junction as a function of pressure. The curve is generated by running a number of snapshot analyses with the junction subjected to a different demand in each analysis.

The example calculates the hydrant rating curve of the example network Net1, which is distributed with the Epanet program. This example network was set up using US customary units. The hydrant rating curve for junction 22 is calculated in the example for demands of 0, 2000, 4000 and 6000 US gal/min.

A particularly powerful application of object-oriented programming, in which a derived class automatically inherits all the functionality of its base class, is illustrated in this example. This allows the user to expand or modify the behaviour of a given class with minimum effort, while retaining the functionality of the base class. In the example, a user-defined class, called MySystemClass is derived from the Ooten ONSystem class. The functionality of this class is then expanded by adding a method called hydrantRatingCurve() to calculate and return the pressures at a given junction for a list of user specified demands.

The example code is listed below:

```
#include <iostream>
#include "ONSystem.h"

// Derive a class from ONSystem.
class MySystemClass : public ONSystem
{
  public:
```

```
    // Add a method to calculate a hydrant rating curve. The method calculates and returns
the
    // pressures at  node junc that correspond to outflows in 'demands'.
    vector<double> hydrantRatingCurve(ONJunction* junc, const vector<double>
&demands)
    {
      // Calculate the hydrant rating curve
      vector<double> pressures(demands.size());
      // Test for null pointer
      if (!junc) return pressures;
      for (unsigned int i = 0; i < demands.size(); i++)
      {
        junc->setBaseDemand(demands[i]);
        initializeHydraulics(false, false);
        runHydraulicSnapshot();
        pressures[i] = junc->pressure();
      }
      return pressures;
    }
};

// Main function
int main()
{
  // Define the hydrant demands to test for in a vector called demands.
  vector<double> demands;
  demands.push_back(0);  demands.push_back(2000.0);
  demands.push_back(4000.0); demands.push_back(6000.0);
  vector<double> pressures;
  try
  {
    //create a new instance of MyClassSystem
    MySystemClass mySystem;
    // Open a new file in Ooten and the hydraulics solver
    mySystem.open("net1.inp", "net1.rpt", "");
    mySystem.openHydraulics();
    // Calculate the hydrant rating curve for junction with id "22"
    ONJunction* myJunc = dynamic_cast<ONJunction*>(mySystem.node("22"));
    // Check that conversion did not return the null pointer
```

```
    if (!myJunc) throw ONException("Junction " + myId + " not found", "main()");
    // Calculate the hydrant rating curve
    pressures = mySystem.hydrantRatingCurve(myJunc, demands);
    // Close the hydraulic system and Ooten
    mySystem.closeHydraulics();
    mySystem.close();
  }
  // Catch and handle any exceptions thrown in the code
  catch(ONException e)
  {
    mySystem.close();
    cout << e.report() << endl;
  }
  // Report results
  cout << "Hydrant rating curve for junction 22" << endl;
  for (unsigned int i = 0; i < pressures.size(); i++)
  {
    cout << "Pressure = " << pressures[i]
      << " for hydrant flow = " << demands[i] << endl;
  }
}
```

A few sections of the example code warrant further explanation. The method hydrantRatingCurve() of MySystemClass takes as argument a pointer to a junction. Since only junctions have demands, hydrant rating curves cannot be calculated for any other type of node.

To obtain a pointer to junction 22, the ONSystem class's node() method is called in the main function, using the node id as argument. This method returns a pointer to a node. However, the pointer has to be converted to point to a junction before the hydrantRatingCurve() method can be called. Dynamic casting is used to do the pointer conversion. If the conversion is not successful, for instance if the node is not a junction (e.g. a tank), the null pointer is returned. In such a case, the example program throws an exception, supplying information on the problem that occurred, and where the exception was thrown.

B.6 Conclusions

This appendix introduced an object-oriented programmers toolkit for Epanet, called Ooten, which was developed to provide certain Epanet source code functions in an object-oriented way. Ooten contains minimal data, but relies on the data structures defined in the Epanet source code. It can thus be viewed as an object-oriented "shell" for the Epanet source code.

Ooten currently provides all the functionality of the Epanet Programmers Toolkit, but also contains additional features such as methods for manipulating curves. Ooten is available as a free download from http://general.rau.ac.za/civil/wrg/ooten.htm.

B.7 References

Rossman, L.A. 2000. *EPANET 2 users manual EPA/600/R-00/057*. Cincinnati: US Environmental Protection Agency.

APPENDIX C – DATA ANALYSIS PROCEDURES

C.1 Introduction

Modern water meters and data logging equipment are rapidly becoming state-of-the-art equipment for most water supply authorities. These systems provide data at very high resolution (at 1-hour or 15-minute intervals, or even less) which open a window on water demand patterns which was not previously available. High data resolution, however, translates into vast volumes of data, which are not amenable to simple spreadsheet analysis. It was therefore necessary to develop software for the rapid and standard analysis of data retrieved from data loggers.

Numerous alternative data analysis procedures were experimented with by using a number of typical data sets, before the approach used in this project was adopted. The purpose of this annexure is twofold:

- to demonstrate how the methods are applied to a practical data set, and
- to serve as a preliminary proposal for a standard data analysis procedure which could also be useful to others working in this field.

The test data is a data set provided by Mr. Ben van Merwe, former Water Engineer of the City of Windhoek. Water flow rate was continuously recorded by pen onto monthly paper strips at the outlet of the High Sam Reservoir Complex, one of several serving the City of Windhoek. The strips were manually digitised at hourly intervals by Bernard Groenewald, a former final-year civil engineering student at the Rand Afrikaans University.

C.2 Data filtering and patching

Data sets will rarely be perfectly complete, or perfectly accurate. Typical errors are null values during power or other logging interruptions, a zero followed by a very high value when the volume used during one period is not transmitted properly and then inadvertently added to the volume used during the next period, *etc.* The first step is therefore to screen the data for imperfections. The simplest method is to plot the complete data set, sorted from high to low. Excessively high values, as well as zero or

very low values are then immediately and easily visible. By setting realistic upper and lower limits, the dubious points falling outside the acceptable range can be flagged.

The dubious points can either be completely stripped from the data set (leaving undefined gaps at those points), or they can be patched using some average substitute values. A stripped data set will have data discontinuities which require a significant complication in the method of analysis. For that reason, the general preference would be to rather patch a data set to enable simple and rapid analysis.

The choice between stripping and patching is, however, not only one of convenience. It is also determined by the type of analysis required. When extracting extreme peak factors for example (not covered in this project), patching would be in order, as an average substitute value could not possibly be an extreme value. When determining serial correlation, for example (as done in this project), it is essential to revert to stripping. When a data point is completely missing, the correlation with the preceding point simply cannot be calculated. The serial correlation for the entire set is thus only based on original values. If the set is to be patched with average values, the serial correlation for the entire set will be contaminated by correlation between real and patched values. For this project, stripping was therefore performed rather than patching.

C.3 Removal of annual trend

The raw daily volumes, expressed as peak factors, are shown in Figure C.1, after all the dubious data points had been stripped. (Consecutive data points are connected with a line - stripped points are thus present where there are breaks in the connecting line.) In this case, it is quite obvious that there is a consistent trend over the full year. To determine this trend, a linear regression line is fitted through all the data points, which is shown in bold in Figure C.1.

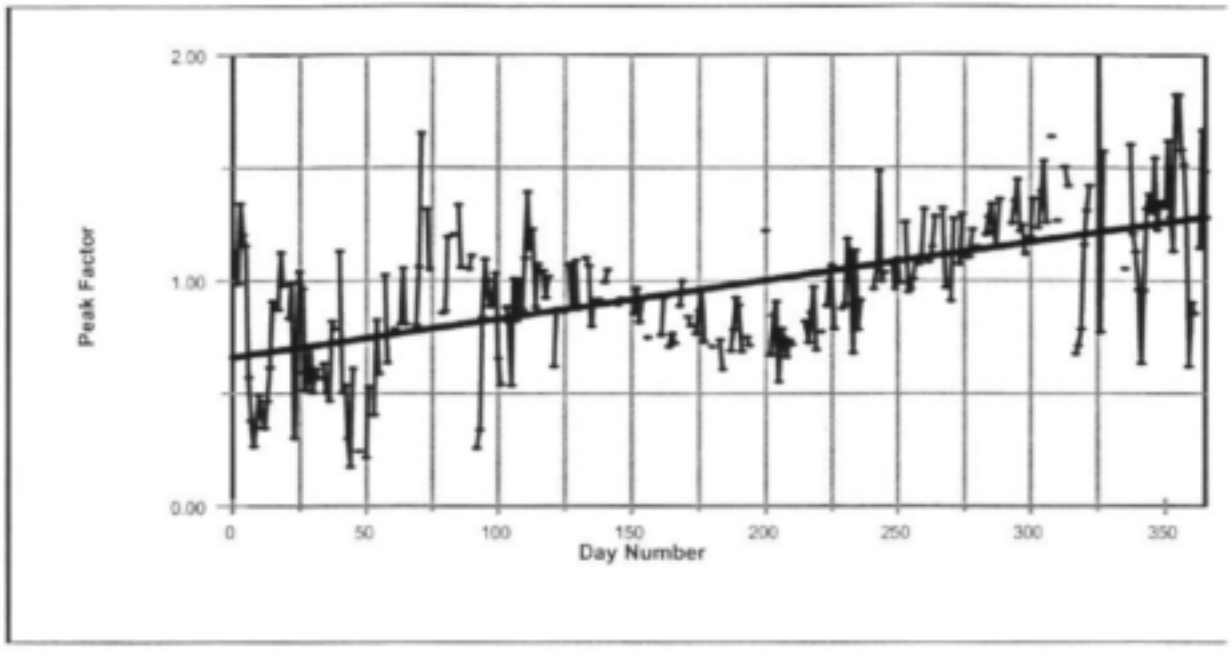

Figure C.1. Daily peak factors for sample data with linear trend line

The annual trend is removed by dividing each daily value by the corresponding value on the linear trend line. The resulting data points are shown in Figure C.2.

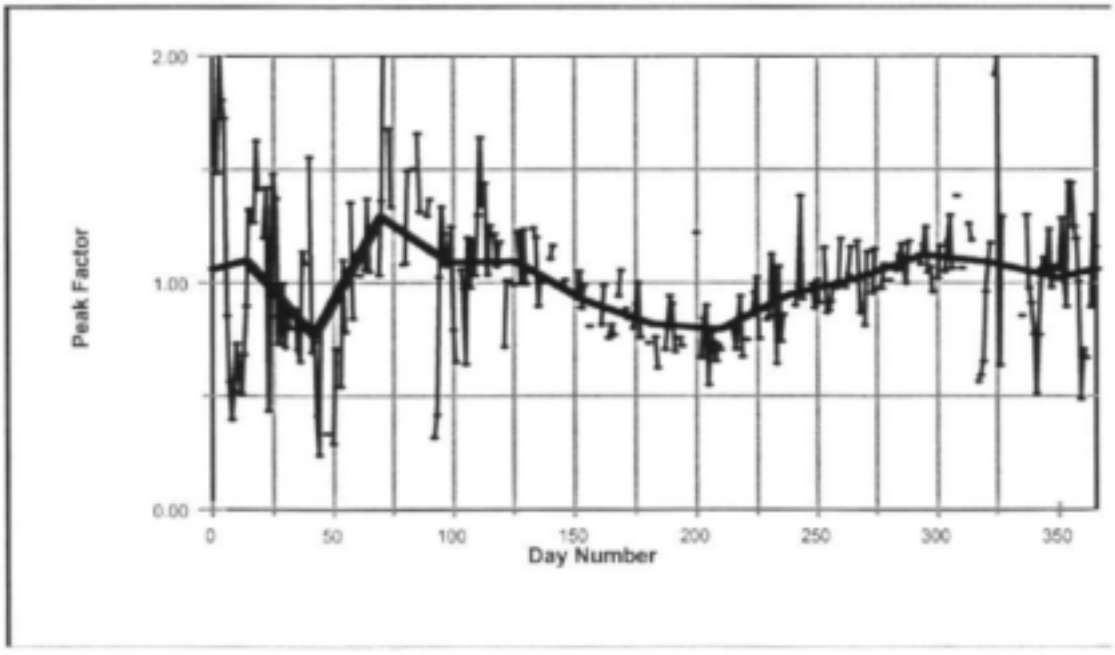

Figure C.2. Daily peak factors (after removal of linear trend) for sample data set with seasonal trend

C.4 Removal of seasonal trend

The seasonal trend is calculated by dividing the year into 13 blocks of 28 days each. (This accounts for 364 days - the remaining day is dropped for convenience with negligible error.) The average peak factor for every block is calculated and plotted in the centre of each block, with straight lines connecting these points. This results in the seasonal trend line, shown in bold in Figure C.2.

The seasonal trend is removed by dividing each daily value by the corresponding value on the seasonal trend line. The resulting data points are shown in Figure C.3.

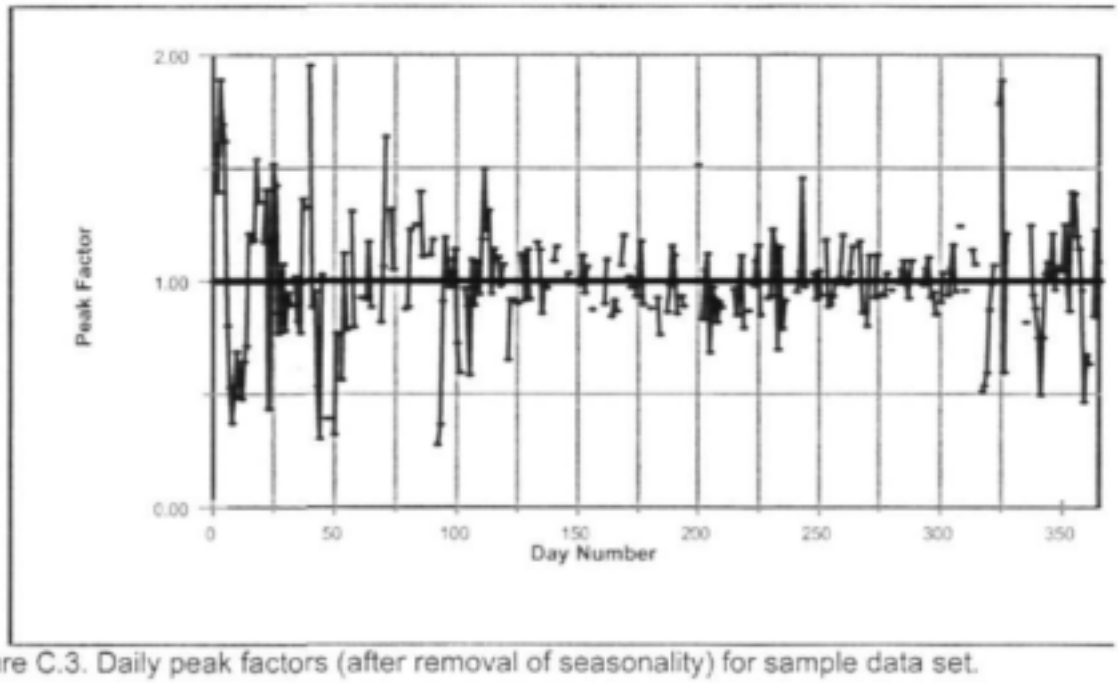

Figure C.3. Daily peak factors (after removal of seasonality) for sample data set.

C.5 Removal of weekly pattern

The average peak factors for each day of the week is calculated next. These average weekly factors are shown in Figure C.4. The weekly pattern is removed by dividing each daily value by the peak factor of the corresponding day of the week. The resulting data points are shown in Figure C.5.

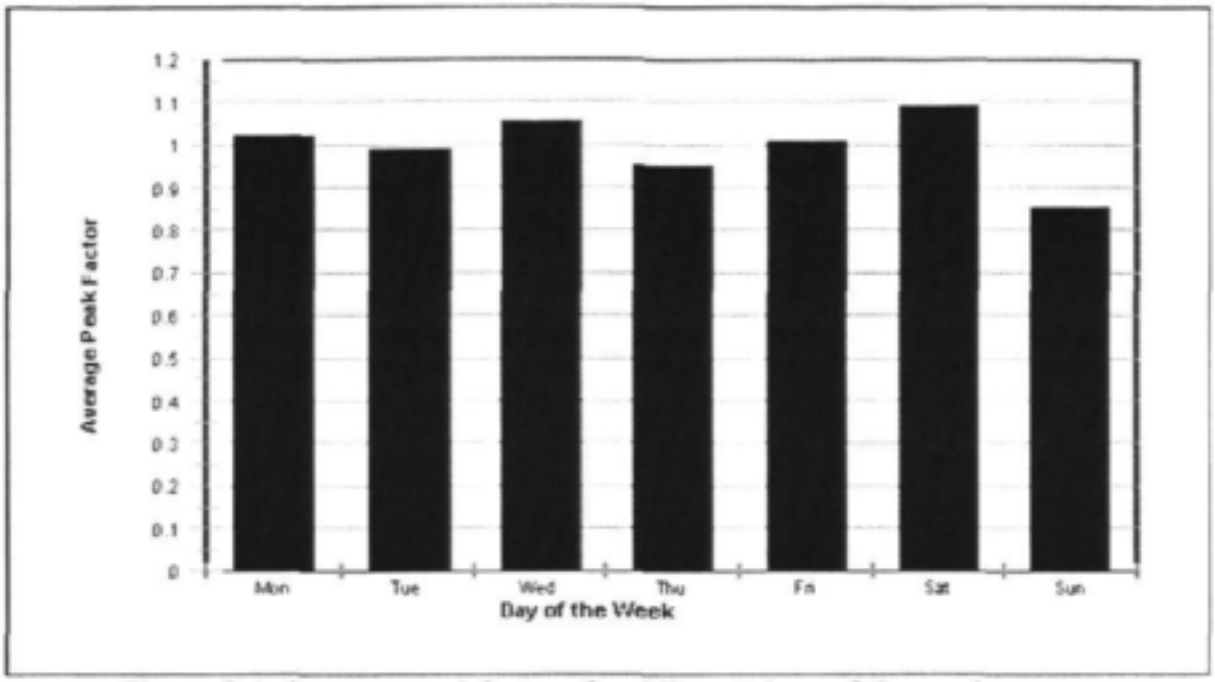

Figure C.4. Average peak factors for different days of the week.

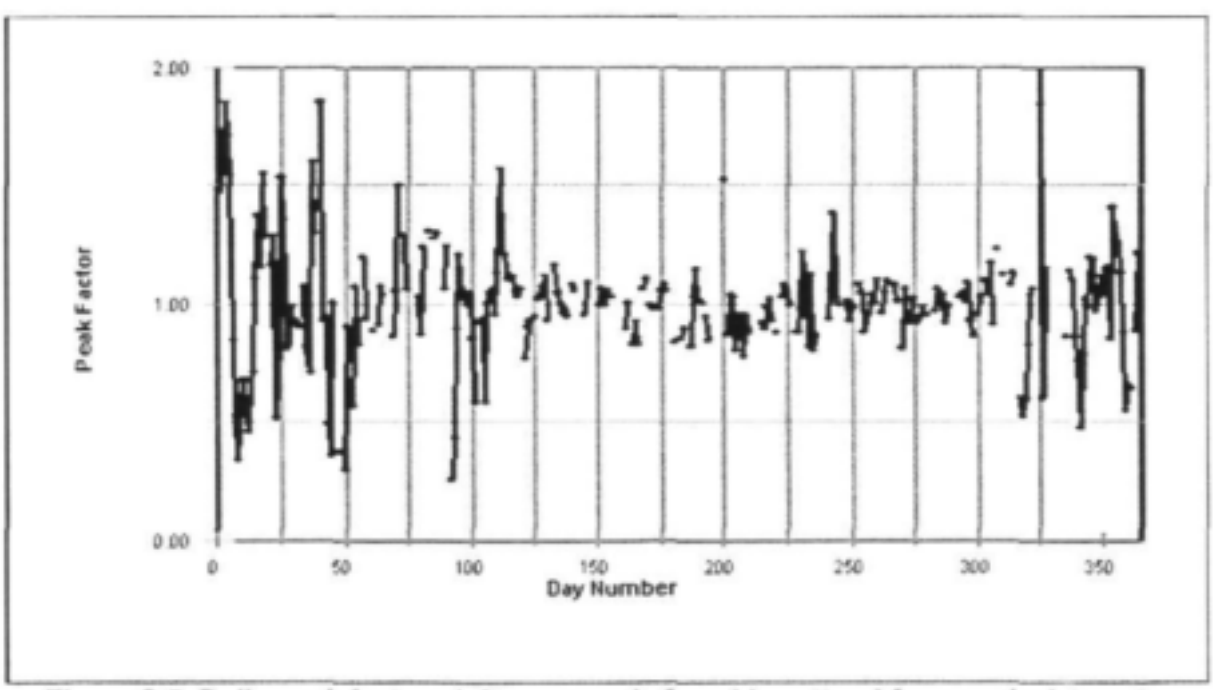

Figure C.5. Daily peak factors (after removal of weekly pattern) for sample data set.

C.6 Removal of serial correlation

The serial correlation is determined by plotting each value against the preceding value. (If there is no preceding value as a result of data stripping, nothing is done and the procedure moves forward by one step.) From this plot, a correlation coefficient is determined - the serial correlation coefficient.

The serial correlation can then be removed from each data point (see Chapter 3 in the report for the appropriate equations) to leave the remaining white noise. Although the year has a maximum of 365 days and can therefore theoretically yield 364 points of white noise, the result of data stripping in this case reduced the number to 197 points of white noise. (One solitary missing data point will lead to the loss of two points of white noise.)

C.7 Characterisation of white noise

The available points of white noise are then sorted into a number of "bins" in order to plot a histogram of the white noise. The histogram is shown in Figure C.6. This is a typical result found in all the numerous cases analysed during this project. The shape of the histogram strongly indicates a normal distribution.

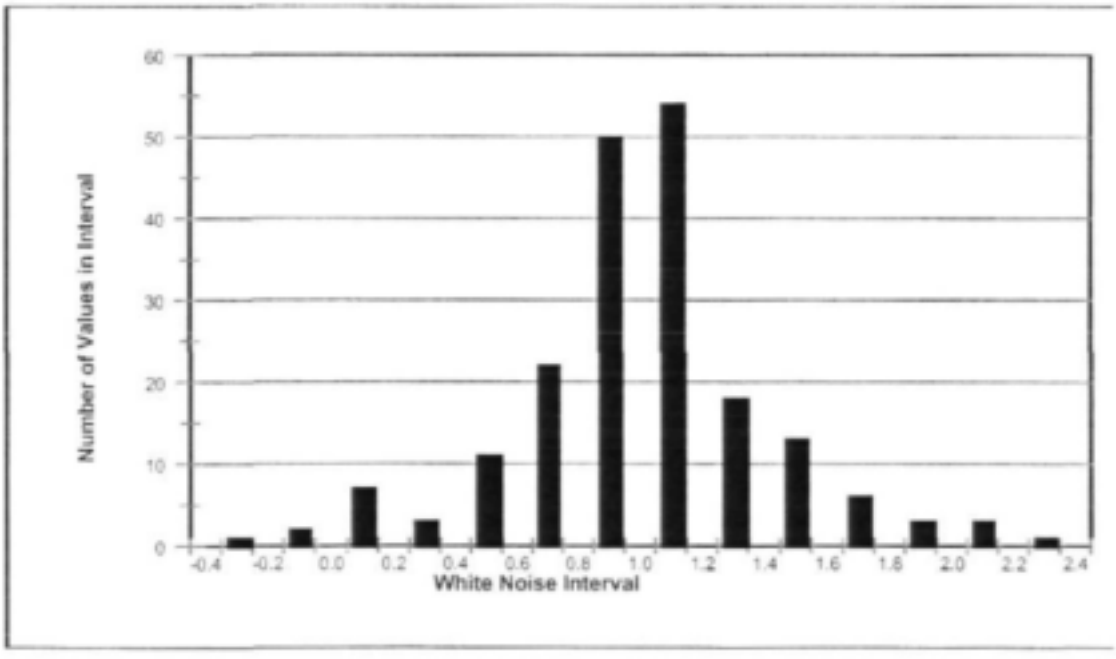

Figure C.6. Histogram of white noise values for sample data set.

As a final step, the hypothesis (that the white noise follws a normal distribution) is tested. In this case, the hypothesis was shown to be confirmed at a confidence level of 99,9%. The white noise is finally characterised by a normal distribution with mean of 1,0 and standard deviation of 0,46 which is graphically shown in Figure C.7.

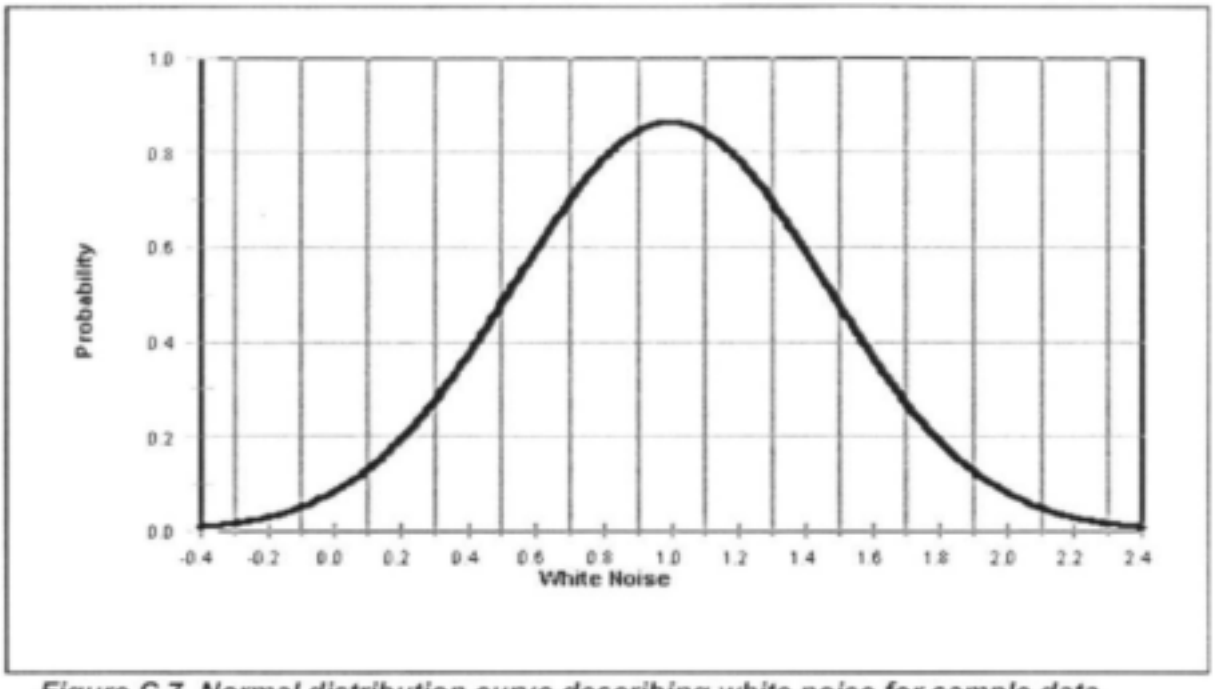

Figure C.7. Normal distribution curve describing white noise for sample data set.

C.8 Summary of parameters

The objective of the exercise described in this annexure is to reduce a typical data set to a number of numerical parameters which can be used for stochastical simulation. In the case of the data set used above, the equivalent parameters are given by:

Seasonal variation:

PF_1 = 1.0985
PF_2 = 0.7643
PF_3 = 1.2920
PF_4 = 1.0898
PF_5 = 1.0946

PF_6 = 0.9229
PF_7 = 0.8181
PF_8 = 0.7952
PF_9 = 0.9451
PF_{10} = 1.0157
PF_{11} = 1.1233
PF_{12} = 1.0920
PF_{13} = 1.0231

Weekly variation:

PF_{monday} = 1.0222
$PF_{tuesday}$ = 0.9908
$PF_{wednesday}$ = 1.0522
$PF_{thursday}$ = 0.9530
PF_{friday} = 1.0095
$PF_{saturday}$ = 1.0931
PF_{sunday} = 0.8536

Serial correlation coefficient:

Lag-one coefficient = 0.526

White noise:

Average = 1.00
Standard deviation = 0.46

Note Stochastic simulation is done for a stationary situation, in other words where there is not a long-term growth trend. For this reason the linear trend is first removed from the data. In other words, the linear trend is only removed to convert the data to a stationary situation - not to use it for stochastic modelling purposes.

APPENDIX D – MOCASIM INPUT FILES

D.1 Simple Water Suppy System Input File

```
[TITLE]
 Simple water supply system

[JUNCTIONS]
;ID                  Elev            Demand          Pattern
 Town                1100            100                     ;

[RESERVOIRS]
;ID                  Head            Pattern
 Source              1223                                    ;
 Reservoir           1207                                    ;

[PIPES]
;ID         Node1       Node2      Length  Diameter  Roughness  MinorLoss  Status
 TownSup    Reservoir   Town       1000    400       100        0          Open   ;
 Feeder     Source      Reservoir  2935    400       100        0          Open   ;


[VALVES]
;ID          Node1             Node2                  Diameter       Type   Setting
             MinorLoss

[PATTERNS]
;ID              Multipliers
;
 Seasonal        1.25          1.35          1.25          1.15          0.95
       0.85
 Seasonal        0.80          0.75          0.80          0.80          0.90
       1.15
 DOW             0.9           1.1           1.04          1.08          0.92
       1.04
 DOW             0.92
 Daily           0.42          0.4           0.41          0.43          0.46
       0.6
 Daily           0.99          1.4           1.72          1.8           1.75
       1.6
 Daily           1.4           1.25          1.2           1.22          1.25
       1.28
 Daily           1.2           0.95          0.71          0.6           0.51
       0.45

[TIMES]
 Duration             24:00
 Hydraulic Timestep   1:00
 Quality Timestep     0:05
 Pattern Timestep     1:00
 Pattern Start        0:00
 Report Timestep      1:00
 Report Start         0:00
 Start ClockTime      12 am
 Statistic            None

[REPORT]
 Status               No
 Summary              No
 Page                 0

[OPTIONS]
 Units                LPS
 Headloss             H-W
 Specific Gravity     1
 Viscosity            1
 Trials               40
 Accuracy             0.001
 Unbalanced           Continue 10
 Pattern              1
 Demand Multiplier    1.0
 Emitter Exponent     0.5
 Quality              None mg/L
```

```
 Diffusivity           1
 Tolerance             0.01

[COORDINATES]
;Node                  X-Coord              Y-Coord
 Town                  10470.02             7115.07
 Source                -1458.67             8525.12
 Reservoir             5243.73              7660.17

[BACKDROP]
 DIMENSIONS            0.00          0.00          10000.00          10000.00

 UNITS                 None
 FILE
 OFFSET                0.00                 0.00

[END]

[MSDEMANDMODELS]
;ID          Seasonal pat DOW pat  Daily pat Auto corr Random dist   Random parm1  Random
parm2
demandP1     Seasonal     DOW      Daily     0.4         normal        1             0.2

[MSFIREMODELS]
;ID    Sp dist   Sparm1  Sparm2  Dur dist  Dparm1  Dparm2  Demand dist   Drparm1  Drparm2


[MSFAILMODELS]
;ID          Spacing dist   Spac parm1 Spac parm2  Duration dist Dur parm1 Dur parm2
FailM1       uniform        0          428.51      lognormal     15        2.1

[MSDEMANDS]
;Junction      Demand model
Town           demandP1

[MSFIRES]
;Junction      Fire model

[MSFAILURES]
;Link     Fail model
Feeder    FailM1


[MSOPTIONS]
 Duration              365000
 Days per year         365
 Days per week         7
 Seasons per year      12
 Critical pressure     10
 Reservoir             Reservoir
 Minimum size          2160
 Maximum size          100000
 Multiply              1.3
 Add                   0
 Write log             no
 Log file              lpipenew.log
 Event statistics      yes
 Results file          lpipenew.out
 Random seed           0

[MSSTATISTICS]
;Type          ID             Parameter       Minimum      Maximum      No of bins
node           Town           demand          0              350          10
node           Town           pressure        80             110          10
node           Town           head            1170           1210         10
node           Town           deficit         0              350          10
link           Feeder         status           0             1            2
link           Feeder         flow            0              150          10
link           Feeder         uheadloss        0             6 -          6

[MSEND]
```

D.2 Simple Water Suppy System Output File

```
TANK RELIABILITY ANALYSIS FOR NODE Reservoir
Time unit = 3600 s
Capacity        No of Failures Ave Duration   Dur Std Dev     Minimum         Maximum
2160             96730          6.15194        3.36767        0.000631671     52.9588
2808             38795          5.73158        3.59041        0.000495053     50.9027
3650.4           28575          5.81677        3.60074        0.000495053     49.6707
4745.52          20956          5.94883        3.67193        0.000495053     48.0364
6169.18          15416          6.0943         3.66855        0.000689185     44.3811
8019.93          11477          6.15905        3.64784        0.00085695      37.4797
10425.9          8358           6.22382        3.63129        0.00173607      36.17
13553.7          5890           6.2965         3.60028        0.00173607      36.17
17619.8          3929           6.34473        3.54136        0.00301137      33.8304
22905.7          2332           6.39352        3.60153        0.00301137      33.7047
29777.4          1101           6.42784        3.72674        0.00301137      31.2489
38710.7          415            6.19362        3.40747        0.0613029       23.0663
50323.9          78             5.92751        3.05214        0.0613029       14.0124
65421            11             5.24489        3.39214        0.571804        9.92734
85047.3          0              0              0              0               0

NODE STATISTICS
Demand at junction Town
No of points   Average         Std Deviation  Minimum         Maximum
8760024        97.5486         58.8839        12.5026         465.043
Bins:   <= 0   0       35      70      105     140     175     210     245     280     315
        350
Frequency:     0       1.07462e+06     2.34121e+06     2.10555e+06     1.39721e+06     830680
               526457  280435  141294  54211.6 11827.1 1429.39

Pressure at junction Town
No of points   Average         Std Deviation  Minimum         Maximum
8760024        103.849         3.52288        62.6805         106.945
Bins:   <= 80  80      83      86      89      92      95      98      101     104     107
        110
Frequency:     936     3369.28 12375.3 36648.1 85787.2 169777 349785 708718 1.69167e+06
        5.70586e+06    0       0

Head at junction Town
No of points   Average         Std Deviation  Minimum         Maximum
8760024        1203.85         3.52288        1162.68         1206.95
Bins:   <= 1170 1170    1174    1178    1182    1186    1190    1194    1198    1202    1206
        1210
Frequency:     6       23      269     2328.86 14053.8 58392.7 164383  419216  1.0931e+06
        4.17507e+06    2.83808e+06    0

Demand deficit at junction Town
No of points   Average         Std Deviation  Minimum         Maximum
8760024        0               0              0               0
Bins:   <= 0   0       35      70      105     140     175     210     245     280     315
        350
Frequency:     8.76493e+06     0       0       0       0       0       0       0       0
        0      0       0


LINK STATISTICS
Status of pipe Feeder
No of points   Average         Std Deviation  Minimum         Maximum
8760024        0.991402        0.0913082      0               1
Bins:   <= 0   0       0.5     1
Frequency:     77766   0       8.68716e+06    0

Flowrate in pipe Feeder
No of points   Average         Std Deviation  Minimum         Maximum
8760024        148.709         13.6961        0               149.999
Bins:   <= 0   0       15      30      45      60      75      90      105     120     135
        150
Frequency:     77766   0       0       0       0       0       0       0       0       0
        8.68716e+06    0

Unit headloss in pipe Feeder
No of points   Average         Std Deviation  Minimum         Maximum
8760024        15.8624         1.46093        0               16
Bins:   <= 0   0       1       2       3       4       5       6
Frequency:     77766   0       0       0       0       0       0       8.68716e+06
```

```
EVENT STATISTICS
Days between failure events for pipe Feeder:
No of points   Average        Std Deviation  Minimum        Maximum
4925           74.1013        41.9252        0.625009       145.95

Duration of failure events for pipe Feeder:
No of points   Average        Std Deviation  Minimum        Maximum
4925           15.293         2.16442        8.72809        24.9164
```

D.3 Simple Water Suppy System With Fires Input Files

The input files for the simple water supply system with fires is the same as for the Simple system (Section D.1), except for the following section.

System with small fires

```
[MSFIREMODELS]
;ID          Sp dist    Sparm1  Sparm2  Dur dist   Dparm1  Dparm2  Demand dist    Drparm1
Drparm2
SmallFire     lognormal  1       1       normal     2       0.4     normal       15         3

[MSFIRES]
;Junction      Fire model
Town           SmallFire
```

System with large fires.

```
[MSFIREMODELS]
;ID          Sp dist    Sparm1  Sparm2  Dur dist   Dparm1  Dparm2  Demand dist    Drparm1
Drparm2
LargeFire    lognormal 30.4    6       normal     6       1.2     normal        200       40

[MSFIRES]
;Junction      Fire model
Town           LargeFire
```

Other related WRC reports available:

Sizing of bulk water supply systems with a probabilistic method

Haarhoff J; van Zyl JE

Stochastic analysis of systems is a technique that has been applied with great success in numerous different fields, including water supply. The main advantage of the technique is that it is possible to link the risk of a bulk water supply system failing to the cost of the system. This means that if the acceptable risk of system failure is known, as is the case with design guidelines for developing communities, the optimum reservoir capacity might be determined using the stochastic technique. Experience with the technique has shown that in many cases the "48 h of annual average daily demand" rule specified in the "Red Book" for the sizing of reservoirs results in over-estimation (with resulting over-spending) of the required reservoir capacity. As per the set objective the study has culminated in the following:

- Estimates were derived for a typical urban bulk water supply system. The number of supply interruptions is typically between 0.1 and 1 interruptions per year, and the total annual interruption duration ranges between 1 and 10 h per year.
- The approach can be extended to a typical branched configuration as found in many cities, i.e. where water is supplied to a primary reservoir, from where it is further distributed to secondary reservoirs. The analysis of the Windhoek reservoir system, as an example, showed that about 8 h of the annual average daily demand (AADD) of the combined secondary reservoirs should be allocated to the primary tank if the primary tank has only an Atransfer@ function.
- The analysis of the Mabeskraal system showed how the method is applied to a long, linear Abackbone@ system with small side reservoirs. In such systems, tank sizes should increase with distance from the source. Generally, the tanks can be significantly reduced in size (to between 12 h and 24 h of AADD) without a compromise in supply security.
- The Mabeskraal analysis also pointed to some important differences between urban and rural systems. The demand peaks are much more attenuated, due to on-site storage and the use of standpipes. There is also stronger serial correlation between consecutive days than in urban areas. Pipe breaks are less frequent in rural areas, and repair times generally longer.

Report Number: 985/1/02 **ISBN: 1 86845 837 7**

TO ORDER: Contact **Publications** - Telephone No: 012 330 0340
Fax Number: 012 331 2565
E-mail: publications@wrc.org.za

1770052127

Water Research Commission
Private Bag X03, Gezina, 0031, South Africa
Tel: +27 12 330 0340, Fax: +27 12 331 2565
Web: http://www.wrc.org.za