SOFTWARE CAPACITY PLANNING: A METHODOLOGY FOR A PORTFOLIO OF HIGH TECHNOLOGY PRODUCT DEVELOPMENT PROJECTS

By

RAJIV MALHOTRA

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

Of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2009

DEDICATION

To my Father, he was the first person to articulate the vision.

To Cecilia, without her unconditional support and encouragement, this work would not be a reality.

To my Mother, for her complete acceptance.

To Ana and Raul, their actions have inspired me to continue learning.

To Hamilton, for his presence and smile.

To all my Friends and Well Wishers, for their support.

ACKNOWLEDGEMENTS

I want to extend my deepest gratitude to the Department of Industrial and Manufacturing Systems Engineering for providing the opportunity of a lifetime for a great academic experience. A special thanks to Dr. Don Liles, my dissertation advisor and the Chairperson of the Industrial Engineering department, for his wisdom and guidance to chart a course to success. I especially appreciate the confidence he showed in my abilities and trust to execute on the plan. His comments were always thoughtful, measured and guided me in the right direction.

I will always be grateful to all those who have helped me with their thoughtful suggestions, insight and experience. There are some that require special mention. Dr. Cecilia Temponi, she was my sounding board and not afraid to challenge my perceptions. Dr. Brian Huff, he had a fresh perspective and a unique way at looking at my proposal. Dr. John Priest, he gave me many insights into Neural Nets. Dr. H.W. Corley, he gave encouragement and support as my Graduate Advisor. Dr. Sheik Imrhan, he made the subject of statistics meaningful.

A special thanks goes to all the other members of my dissertation committee for their comments to guide and improve this research.

April 10, 2009

ABSTRACT

SOFTWARE CAPACITY PLANNING: A METHODOLOGY FOR A PORTFOLIO OF HIGH TECHNOLOGY PRODUCT DEVELOPMENT PROJECTS

Rajiv Malhotra, PhD.

The University of Texas at Arlington, 2009

Supervising Professor: Don Liles

High technology product development projects make extensive use of engineering software during the product development process. The suite of engineering software tools deployed during product development represents a significant portion of the product development costs. The ability to forecast the engineering software license capacity required to support product development plans is crucial for budgeting, return on investment (ROI) calculations, contract negotiations with the software suppliers, and the provision of an IT infrastructure necessary to support the execution of the software tools.

This research shows that the usage of engineering application software in high technology is cyclical due to the characteristics of high technology product development. A rigorous methodology to compute the cycle boundaries based on usage history is proposed. Information of the usage cycles is used to modify the existing trend forecast to increase the prediction accuracy of future usage predictions. Data of the current usage of the projects in execution is then used to forecast the overall software capacity needed to support all current projects. Cyclic usage patterns also predict how the usage is expected to change in the future for the time period under study.

Usage data is collected for the three main projects concurrently in execution. Of the three, one of the projects goes through its full development cycle. In this research we show (a) a rigorous methodology using Fourier analysis to extract the cyclical variations of the project that goes through its full development cycle and (b) the application of the cyclical variations to a trend forecast to improve the quality of the forecast.

Single project forecasts are then combined to generate an overall engineering software capacity forecast for the enterprise. All engineering software licenses for both project and non-project work are shared by the enterprise from a central license pool. To determine the enterprise capacity requirement, the relationship between the key individual projects and the total usage is determined and used to predict future capacity.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Statement of Purpose and Background

Engineers use software tools extensively in high technology product development. In a large high technology enterprise, development of new high technology products is a core competency. There is high demand from engineers for engineering application software tools.

Engineering software tools are used in nearly all phases of product development. During the concept phase, they may be used to capture (a) the constraints of the external environment that the product must meet, (b) the expected behavior of the product when it receives stimuli from the external environment, and (c) the performance goals that must be met. Similarly, at the architecture development stage, software tools are used to capture successive levels of architectural details and to develop the verification environment used to test the product during the design phase and after fabrication. During the design phase, software tools are used to implement and verify the detailed design, manage design data, and create design data in a format suitable for the next step, which is usually the manufacturing of the product. Software tools may further be used for the creation and debugging of prototypes, sample and product testing, quality control, and reliability on the production floor.

The high technology product development enterprise faces forecasting future demand for licenses for the engineering application software (EAS). Deployment and utilization of the EAS vary according to the phase of the product development. At any given time, a large enterprise typically develops multiple products, which may be in varying stages of development. From the EAS usage perspective, usage patterns appear to be erratic when the aggregate usage across multiple products is viewed.

The software vendor or supplier authorizes software usage through a licensing mechanism. A software application, upon invocation, has a built-in mechanism to request authentication from a designated license server. Besides authenticating the request, the license server, maintains a count of all license requests granted. The license request is granted if the maximum number of licenses available has not been exceeded. Similarly, once an application terminates, it informs the license server so that it may update its license count to reflect the availability of an additional license for the next license request. Usually, the license server maintains a log of all check-in and check-out requests in a file. The log file is processed to generate usage data for usage reporting.

It is common for an enterprise to serve licenses from a central pool shared by all the projects. This is done to maximize utilization of the available licenses and to facilitate the administration of licenses. License Administration is the function of installing and deleting licenses to maintain contractual compliance with the software suppliers, administer any corporate license policies, and implement safeguards against unauthorized usage.

Each high technology product development project has a demand for EAS licenses. This demand may be implicit or explicitly forecasted by the project team. The aggregate demand for software licenses of all the projects motivates the need for a software capacity plan. The objective of the software capacity plan is to determine the number of licenses to be made available on the license servers in order to meet the demand from all the projects.

High technology product development projects are characterized by a high degree of technological uncertainty that reflects itself as uncertainty in planning for the project resources and schedules. Project resources and schedules for most high technology product development projects are difficult to forecast, due to the diminished relevance of historical data from previous projects. Due to the increased technological complexity of the current project, as compared to the previous projects, there is an increase in technological uncertainty. This increase in technological uncertainty is a driver for the innovation and deployment of new development methodologies. New development methodologies may lead to both the deployment of new engineering software not previously used and changes in the patterns of usage of engineering software deployed in previous projects. Software licenses are a vital but expensive resource and must be deployed judiciously.

1.2 Research Objectives

Engineering software tools are created to perform specific engineering functions. Depending on the stage of development of a high technology product, the usage level of a particular engineering software tool may be high, low, or somewhere in between. The concept of high or low is relative and, in this case, means high or low compared to the recent past. In this research, high will be referred to as the peak, and it may be over a specified period of time. Factors such as the technological complexity of the project and the time-to-market goals, among others, may influence the level of peak usage. In the course of product development, engineering software tool usage goes through peaks and lows, thus creating a cyclical usage pattern over time. There may be multiple cyclical usage patterns of varying duration and intensity.

The first key contribution of this dissertation is to propose a methodology to extract the cyclical usage patterns for a given EAS deployed on a product development project for the purpose of forecasting future project EAS usage. An enterprise typically has multiple product development projects in the pipeline. Each project may have its own cyclical usage pattern that can be used to forecast demand. Once there is a forecast of future demand for each ongoing project and there has been a strategic decision to meet this demand, the next step is to make adequate software capacity available. This is the function of software capacity planning.

The second contribution of this dissertation is to develop a model to create a software capacity plan based on the cyclical forecasts of each individual project. Software usage data is collected for one engineering software tool over a period of time that encompasses one complete project. Usage data is also collected for all the other projects in execution during the same time period. A segment of the data is used to extract the dominant usage cycles for each project and to build models of the forecasted demand and software capacity plan. The models are then validated with the remaining usage data.

1.3 Organization of the Dissertation

The work in this dissertation shows how the concepts of demand forecasting and capacity planning, although well established in the traditional services and manufacturing industries, may be developed in the dynamic high technology product development industry. The basic concepts used in this work are derived from various disciplines, some of which have not traditionally been used for the purpose of demand forecasting and capacity planning.

Chapter 2 covers the reviewed literature related to the problem space and the basic disciplines used in this dissertation for demand forecasting and software capacity planning. It covers the key concepts and unique characteristics of high technology product development and explains how it affects the deployment of software engineering tools. This chapter also covers the traditional time series forecasting approaches and establishes the need for a fresh approach to high technology product development projects, a new method to forecast cyclical demand, and the need for neural nets.

Chapter 3 covers the development of the software usage forecasting model for each project when the usage of software is cyclical. It develops a method to extract cyclical usage patterns from the logged historic usage data and to use this information to modify the trend forecast. Chapter 4 develops a model to combine the usage data from multiple projects to develop a software capacity plan for a centralized license service configuration. Chapter 5 analyzes and validates the models developed with the actual software usage data for each project and the capacity plan. Chapter 6 presents the conclusions and the future directions for this research.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

As a first step, we conduct a review of the body of research to categorize high technology product development. The next step is to review the key characteristics of high technology product development. An understanding of the key attributes of high technology product development leads to the fundamental role of software usage in high technology product development. Software usage is directly influenced by the product development methodology.

The software usage data that is recorded is pre-processed and may be viewed as a time series data. To forecast software demand, we survey the techniques used for time series forecasting. Once a demand forecast is in place, techniques used for developing software capacity plans are reviewed.

2.2 High Technology Product Development

Product development projects spanning a varied cross section of industries and organizations, including construction, defense, chemicals, electronics, banking, or semiconductors, vary considerably in how they are planned, executed, and managed. The variation may be due to differences in size, complexity, time span, customers, and technological uncertainty. Each of these parameters merits close examination; however, in this section, we examine the impact of technological uncertainty on product development. The objective of this section is to frame the concept of high technology product development and to illustrate how this may culminate in certain types of EAS usage cycles during the product development phase.

2.2.1 Classification Framework for High Technology Product Development Projects

A framework for the classification of projects enables us to highlight the salient features of high technology projects. These features lead us to develop an understanding of the product development characteristics of such projects. A parameter of differentiation of product development projects may be the technological uncertainty of the technology employed, which we will examine in more detail after briefly reviewing other approaches.

Very well-known taxonomies have been developed by Cash, McFarlan, and McKenney (1988); Ahituv and Neumann (1984); Pearson (1990); and Wheelwright and Clark (1992).

The classification of Cash et al. (1998) selected three parameters and combined them to develop a framework for classifying projects. The parameters are (a) Project Size, (b) Experience with the Technology, and (c) Project Structure. Project size refers to the cost of the project, level of staffing, elapsed time to implement the project, and the number of departments involved with or affected by the project. The greater the project size, the greater the risk associated with the project. Experience with the technology refers to the prior experience of the project team with the technology. Such experience decreases the likelihood of unexpected technical problems. Project structure refers to the degree of precision with which the output of a project can be defined at the time of conceptualization. A well-defined output entails lower risk and the project is regarded as highly structured. The risk increases for projects that are not well structured.

Ahituv and Neumann (1984) identify several factors to be considered as part of a general framework for planning Information Systems Development projects. The key factors are Organizational Scope, Importance, Organizational Maturity, Structuredness Level, Technological Environment, and Development Type.

- *Organizational Scope:* Refers to the number of organizations involved with the project. The more organizations involved, the more prior planning is needed.
- *Importance:* The higher the significance of the project’s contribution to the operation of the organization, the higher the risk associated with the project.

- *Structuredness Level:* A very structured system requires less user participation as the project output is well defined.
- *Technological Environment:* The use of more advanced technology is riskier than using the technology with which the organization is familiar.
- *Development Type:* Refers to the type of product being developed. The development of new products will be riskier than the modification or enhancement of existing products.

Pearson (1990) has considered two parameters—uncertainty about the market focus and uncertainty about the technological approach—as key for the management of innovative projects. A high degree of uncertainty about the market focus and the technology means the project is of an exploratory or speculative nature. A project with a clear focus (low uncertainty) but a high degree of uncertainty about the technology is directed toward known potential markets where the cost and performance targets of the project are difficult to achieve. Projects with a high degree of uncertainty about the market focus but with a very low degree of uncertainty about the technology being used usually imply that a new scientific discovery reduces the uncertainty in an area of technology and opens up a range of possible opportunities. Projects with a low degree of uncertainty about the market focus and technology are low-risk projects.

Wheelwright and Clark (1992) have used the degree of change in the product and the degree of change in the manufacturing process as the two parameters for classifying projects. A greater change in any of these two parameters means a higher risk. Using these parameters, product development projects have been divided into five types.

- *Derivative projects* are usually cost-reduced versions or enhancements of existing products.

- *Breakthrough projects* involve significant changes to existing products and manufacturing processes. They establish core products and processes that differ fundamentally from the previous generation.
- *Platform projects* entail more product and/or product changes than derivatives, but they do not introduce new technologies.
- *Research and development* projects involve the creation of know-how and know-why of new technologies that may eventually translate into commercial products.
- *Alliances and partnerships* can be formed to pursue any type of product development. As a result, the resource requirements for these projects may vary widely.

Shenhar (1993) and Temponi and Malhotra (2002) used technological uncertainty as the main parameter to distinguish among various types of projects and to understand its impact on modern project management approaches. Shenhar suggested a conceptual framework that divides the entire spectrum of projects into four categories based upon their technological uncertainty.

Before defining the four categories, it is important to understand the assumptions made by Shenhar.

- Projects combine multiple resources—including materials, components, and information—into a final outcome that is either a piece of hardware, a service, or an organization, and is not merely the creation of new information or the production of a piece of paper.
- There is an existing or potential customer for the project's outcome.
- Projects of experimental nature or research projects for the purpose of evaluating, learning, or developing technological infrastructure are not considered.

The project classification scheme shown in Table 1 has been adapted from the one proposed by Shenhar (1993) and Temponi and Malhotra (2002).

During the execution of high tech projects, several new key technologies are integrated. Typically, either the new technologies exist in-house or can be acquired from outside before the beginning of the project. Although the technologies may exist, the integration of these technologies creates the product’s value in the market. Instances of high tech projects may include the development of a new family of products or the incorporation of advanced technology into the product for the first time.

In this paper we will focus our analysis on the key attributes of high tech projects and how these projects lead to cyclical usage patterns for the engineering software deployed for product development.

Table 2.1 Project Classification by Technological Uncertainty

Project Type	**Low-Tech**	**Medium-Tech**	**High-Tech**	**Super High-Tech**
Degree of Technical Uncertainty	No new technology is incorporated.	Some new technology is used.	Integrates new, but existing, technology.	Key technologies do not exist at the start of the project.
Typical Projects	Construction of roads, buildings, and bridges.	Derivatives or additional models of an existing product in the electronics, aerospace, mechanical, or electrical industries.	New military systems, new commercial product family in computers, aerospace, or electronics.	Space exploration, concept and technology breakthrough projects.
Technology Development	None required.	Some development and testing.	Substantial development, integration, and testing.	Very large amount of development required.
Design Maturity	Design specifications well defined before start of project.	Design specifications finalized early.	Design specifications finalized late.	Design specifications may change almost to the end of the project.
Project Risk	Very limited.	Some limited risk due to use of new technology.	Added risks due to integration of new technologies.	Very high risks due to new technology and integration.

2.2.2 Key Attributes of High Technology Projects

A project may be defined as a collection of interrelated and independent tasks to be executed. The development of a successful project plan requires that the list of tasks required to complete the project is defined comprehensively. The tasks must be well understood. This implies a clear understanding and documentation of the scope, time to completion, resources required, interdependencies with other tasks, and a metric of completion for each task. These conditions are universally applicable to all types of projects. Tasks in high technology projects, however, have additional challenges that must be negotiated.

- Tasks may be defined while specifications are still preliminary and not finalized.
- Tasks with mutual dependencies may have to be overlapped in time.
- Tasks may go through several iterations before they are considered complete.

Among several factors affecting the product development projects, market position is the most important. The market position refers to the market potential and penetration of the product and its compatibility with current marketing channels. In a high technology environment, the overall project schedule is driven primarily by the target market window (Johnson, 1995). The three additional challenges listed above are the result of the need for faster time to market and management of technical complexity.

2.2.2.1 Task Definition Based on Preliminary Specifications

Developing the product specifications is a very critical function. Product specifications that address the market or customer requirements and simultaneously define a product to be built that is feasible within the constraints of time and resources available are the foundation of a successful product development project. While a major portion of the specification can be quickly defined based on the customer requirements, it may take longer to finalize all the specific features. The developer of specifications may have to carry out some feasibility studies to confirm if certain features are practical to implement for the current project. However, there may be several tasks not closely related to the features that are not yet finalized. It may be possible to start these tasks immediately.

By beginning execution of tasks before the product specifications are finalized, the project team incurs a risk that some of these tasks may be modified or eliminated. The benefit versus cost of such tasks must be thoroughly reviewed by the experienced project team members.

2.2.2.2 Overlapping of Tasks

As product development schedules face persistent schedule pressure from the markets they serve, there is always a need to find techniques to compress the schedule. One such technique is the deployment of Concurrent Engineering (CE). CE involves simultaneous execution of coupled product development phases. Krishnan (1996) has proposed a conceptual framework for identifying the phases of a project that can be overlapped.

As previously defined, a project is a collection of interrelated and independent tasks to be executed. In the context of high technology projects, the interrelated or dependent tasks may be viewed as an aggregate of upstream and downstream tasks; the implication is that the upstream task must be completed to complete the downstream task. On closer analysis, however, this may not always be necessary, and the only constraint may be for the upstream task to be completed before the downstream task can be completed.

Overlapping of upstream and downstream tasks requires a careful analysis of their relationship. Once started, the upstream task may evolve to its final form either rapidly or slowly. The downstream task, if begun before the upstream task is completed, may vary in its degree of sensitivity to changes in the upstream task. The best case for overlapping is when the upstream task, once started, evolves very rapidly and the downstream task has very low sensitivity to changes in the upstream task. In this case, it is possible both to start the downstream task as soon as some preliminary information from the upstream task is available and to finalize the exchanged information early. The worst case for overlapping of tasks occurs when the upstream phase evolves very slowly to its final form and the downstream task is very sensitive to any changes of information in the upstream task. In this case, it is not desirable to start the downstream phase with the preliminary information available in the upstream phase. The other cases of task coupling lies somewhere in the middle of these two scenarios.

2.2.2.3 Iterative Tasks

It is important to summarize a high tech product development flow with its inherent risks due to technical uncertainty. Schedule deadlines dictate that upstream and downstream tasks must overlap and the flow of information between them must be managed very closely. At first glance, it may appear that the case of iterative tasks is the same as that of the overlapping tasks. The fundamental distinction between the two is that overlapping tasks are tasks that can be executed sequentially, but we try to find ways to execute them in parallel to shorten the product development cycle. Iterative Tasks must be executed in parallel because the information from the execution of one dynamically impacts the execution of the other.

2.2.3 Usage of Engineering Software Tools in High Technology Product Development

The previous sections discuss the complex interactions of tasks and subtasks in a high technology product development project. Tasks and subtasks have complex dependencies manifested by the ongoing flow of information between them. As the completion of each task progresses, it affects the progress of other tasks.

In high technology product development projects, a large number of the tasks are executed using engineering software. The software may be deployed for building simulation models for analysis at varying degrees of abstraction, creation and management of the product design database, the release of the product design to manufacturing, complex mathematical computations, and several other applications.

The concurrent and iterative nature of product development tasks usually requires each product development task to be executed multiple times. In fact, without the ability to automate the execution of the product development tasks, it is impractical to implement time to market strategies and to manage product development complexity. As many product development tasks as possible are executed concurrently. This increases the usage of software for a longer duration during the product development cycle. However, due to task dependencies and evolving product specifications, product development tasks have a dominant phase. The dominant phase creates longer software usage cycles, and the iterative and overlapping tasks create shorter duration software usage cycles. The next section is a brief survey of demand forecasting techniques, followed by an approach to forecast the demand cycle by using Frequency Domain analysis.

2.3 Demand Forecasting

To predict the software capacity to be implemented to support the software demand, we review the techniques that may be used to forecast the software demand. Software demand may be forecasted based on historical usage of the current and past projects. In the framework of this research, software usage is captured and stored in a database. The weekly peak usage data is then computed to monitor usage and plan capacity.

General methods for demand forecasting are regression models and time series analyses, which require sufficient historical data (Saito and Kakemoto, 2004). Due to rapid changes in the market, applying these traditional methods tends to be challenging in today's development of multiple new products, short product lifecycles, and highly erratic demand. Therefore, new forecasting methods for high technology product development are necessary for demand planners. In the next section, we survey the traditional time series forecasting methods.

2.3.1 Time Series Forecasting

In time series analysis, the measurements are taken at a regular interval. Although time series data generally exhibits random fluctuations, the time series may still show gradual shifts to relatively higher or lower values over an extended period of time. The gradual shifting of the time series is referred to as the *trend* in the time series. Although a time series may exhibit a trend over long periods of time, all future values of the time series may not fall exactly on the trend line. Any recurring sequence of points above and below the trend line can be attributed to the *cyclical* component of the time series. Similarly, there may be a *seasonal* component in which the time series data varies due to seasonal influences. Finally, a time series has a *random* component that accounts for deviations of the actual values from those expected, given the effects of the trend, cyclical, and seasonal components (Anderson, Sweeney, and Williams, 2004).

The following sections survey the techniques available to forecast demand based on a historical record of a time series of discrete numbers.

2.3.2 General Forecasting Methods

Using a moving average of the values may smooth the random component in a time series. The sensitivity of the moving average to changes in the trend and cycles depends on the lag associated with the moving average and the relative weights assigned to the data of each time series.

A simple moving average (SMA) is one where the data for the lag period of each time series is assigned equal weights. In contrast, the weighted moving average (WMA) is one in which the data of each time series may be assigned a different weight as long as the sum of the weights adds up to 1. Exponential smoothing average (EMA) uses a weighted moving average of the past time series values as the forecast; it is a special case of the weighted moving average in which only one weight is selected—the weight for the most recent observation. The weights for the older data values are automatically computed and get smaller as the observations move further into the past.

$$F_{t+1} = \frac{\sum_{i=p}^{t} Y_i}{t-p} \quad \text{……… SMA with lag, } n = t - p \qquad \text{Equation 2.1}$$

$$F_{t+1} = \alpha Y_t + (1-\alpha) F_t \quad \text{……EMA} \qquad \text{Equation 2.2}$$

F_{t+1} = forecast of the time series for period t+1

Y_t = actual values of the time series in period t.

F_t = forecast of the time series for period t

$\alpha =$ smoothing constant $(0 \le \alpha \le 1)$

Causal forecasting methods are based on the assumption that the variable we are trying to forecast exhibits a cause-effect relationship with one or more other variables. *Regression analysis* is a technique used to develop a model that shows how variables are related. The variable that is being predicted is called the *dependent* variable. The variable or variables being used to predict the value of the dependent variable are called the *independent* variables. Regression analysis involving two or more independent variables is called *multiple regression analysis*. If only the past values of the variable being forecast are used to create the forecast, then the use of regression analysis is not a causal forecasting method. In this research, we investigate and construct a model to forecast future demand for licenses without directly studying the factors that may cause the variation in demand. The factors that may cause the variation in demand for licenses are a topic for further research and are beyond the scope of this research.

2.3.3 Neural Networks

An Artificial Neural Network (ANN) is an information-processing paradigm that is based on the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information-processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like humans, learn by example. An ANN is configured for a specific application, such as a pattern recognition or data classification, through a learning process.

Neural networks, with their uncanny ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze. This expert can then be used both to provide projections when given new situations of interest and to answer "what if" questions.

Neural networks provide several other key advantages:

- *Adaptive learning*: An ability to learn how to do tasks based on the data given for training or initial experience.
- *Self-Organization*: An ANN can create its own organization or representation of the information it receives during the learning time.

- *Real Time Operation*: ANN computations may be carried out in parallel. Special hardware devices that take advantage of this capability are being designed and manufactured.
- *Fault Tolerance via Redundant Information Coding*: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

An ANN consists of three groups, or layers, of units: *input* units, which are connected to a layer of *hidden* units, which is connected to a layer of *output* units.

- They receive the raw information that is fed into the network.
- The processing at each hidden unit is dependent on the output from the hidden units and the weights of the connections between the input and hidden units.
- The processing at each output unit is a function of the processing activity at each hidden unit and the weights between the hidden and output units.

In a single-layer organization, all units are connected to each other. In a multilayer network, units are grouped into layers instead of a flat structure. There are many functions that cannot be represented by a single-layer network. The limitations can be overcome by adding more layers, as in a multilayer network.

A multilayer network has two or more layers of units, with the output from one layer serving as the input to the next. The layers with no external connections are referred to as hidden layers. Any multilayer system with fixed weights that has a linear activation function is equivalent to a single-layer linear system. In a two-layer linear system, the input vector to the first layer is X, and the output Y=W1*X of the first layer is given as input to the second layer. The second layer produces output Z=W2*Y. Thus,

$$Z = W2*(W1*X) = (W2*W1)*X \qquad \text{Equation 2.3}$$

The system is equivalent to a single-layer network with weight matrix W= W2*W1. By induction, a linear system with any number *n* of layers is equivalent to a single-layer linear system in which the weight matrix is the product of the n intermediate weight matrices. A multilayer system that is not linear can provide more computational capability than a single-layer system. Any Boolean function can be implemented by a multilayer network (McClelland and Rumelhart, 1988).

2.3.3.1 Forecasting with Neural Networks

Neural networks have been used for time series modeling and forecasting for quite some time. Since neural networks are best at identifying trends in data, they are well suited for prediction or forecasting. The power and usefulness of artificial neural networks have been demonstrated in several applications, including speech synthesis, diagnostic problems, medicine, business and finance, robotic control, signal processing, computer vision, and many other problems that fall under the category of pattern recognition. To forecast, neural networks learn past patterns from historical data and recognize these patterns when they reoccur.

The common method to train a neural network is called supervised learning and provides a set of inputs and expected outputs to a multilayer network. No learning algorithm had been available for multilayer networks until Rumelhart, Hinton, and Williams introduced the backpropagation training algorithm, also called the generalized delta rule (Rumelhart et al., 1988). At the output layer, the output vector is compared to the expected output. If the difference is zero, no changes are made to the weights of the connections. If the difference is not zero, the error is calculated from the delta rule and is propagated back through the network. The idea, similar to that of the delta rule, is to adjust the weights to minimize the difference between the real output and the expected output. Such networks can learn arbitrary associations by using differentiable activation functions. A theoretical foundation of backpropogation can be found in McClelland and Rumelhart (1986) and in Rumelhart et al. (1988).

The success or failure of neural networks models in time series forecasting may depend on (a) the type of data, (b) the skill of the analyst in selecting a suitable neural network model, and/or (c) the numerical methods used to fit the model and compute predictions (Chatfield, 1998). Chatfield and Faraway (1996) and Faraway and Chatfield (1998) published two case studies that analyze some sales data and airline data. Both series were fairly short by neural network standards (83 and 144 observations, respectively) and might be thought by some computer scientists to be too short for ANN modeling. However, the lengths of the series were typical of data found in many forecasting situations.

Kuvulmaz et al. (2005) published a comparative assessment of some well known linear and nonlinear techniques in modeling and forecasting financial time series with trend and seasonal patterns. Then they investigated the effect of pre-processing procedures, such as seasonal adjustment methods, to the improvement of the modeling capability of a nonlinear structure implemented as ANNs in comparison to the classical Box-Jenkins seasonal autoregressive integrated moving average (ARIMA) model. There was no significant statistical difference between forecasting performances of ANN trained with unprocessed data or ARIMA models. However, the results improved considerably after applying seasonal adjustment to the ANN input data (Kuvulmaz, Usanmaz, and Engin, 2005).

2.3.4 Forecasting of Cyclical Events

In section 2.3, we discussed the concept of a cyclical component in time series data. Prior to that, in section 2.2.2, we reviewed the fundamental nature of high technology product development and the usage of engineering software tools. If we accept the premise that the usage of software engineering tools on a product development project is cyclical, we need a way to forecast the cycles. This motivates us to review techniques to forecast cycles in time series data.

2.3.4.1 ARMA

Autoregressive moving average (ARMA) models are mathematical models of the persistence, or autocorrelation, in a time series. ARMA models are widely used in many areas and are also known as the Box-Jenkins approach.

ARMA models can be described by a series of equations. The equations are usually simplified if the time series is first reduced to zero-mean by subtracting the sample mean.

$y_t = Y_t - \overline{Y}$, t = 1,……N Equation 2.4

where Y_t is the original time series, $\overline{Y}$ is its sample mean, and y_t is the mean-adjusted series. One subset of ARMA models is the so-called *autoregressive*, or AR models. An AR model expresses a time series as a linear function of its past values. The *order* of the AR model tells how many lagged values are included. The first-order AR model, AR(1), is

$$y_t + a_1 y_{t-1} = e_t$$ Equation 2.5

where y_t is the mean-adjusted series in time interval t, y_{t-1} is the series in the previous time period, a_1 is the lag-1 autoregressive coefficient, and e_1 is the *noise*, also known as the *random-shock* or *residual*. The residuals e_t are assumed to be random in time (not autocorrelated) and normally distributed. If we rewrite the AR(1) model as

$$y_t = -a_1 y_{t-1+} e_t$$ Equation 2.6

we see that y_t is regressed on its previous value and that e_i is analogous to the regression residuals. The term *autoregressive* refers to regression on self (auto).

Higher-order autoregressive models include more lagged y_t terms as predictors. The pth order autoregressive model, AR(p), includes lagged terms for time periods t-1 to t-p.

The *moving average* (MA) model is a form of ARMA model in which the time series is regarded as a moving average of a random shock series e_t. MA(1) is the first order moving average model and is given by

$$y_t = e_t + c_t e_{t-1}$$ Equation 2.7

As with AR models, the higher-order MA models include higher lagged terms. A qth order moving average model is denoted by MA(q).

The autoregressive moving average model includes lagged terms on the time series itself, and the moving average model includes lagged terms on the residuals. If both types of lagged terms are included, the model is called *autoregressive-moving-average*, or ARMA. An ARMA(1,1) model is

$$y_t + a_1 y_{t-1} = e_t + c_t e_{t-1}$$ Equation 2.8

ARMA models have traditionally been used for forecasting, but these models require sufficient historical data. Due to rapid changes in the business environment, applying traditional methods is more challenging, particularly due to short product lifecycles and erratic demand (Saito & Kakemoto, 2004).

2.3.4.2 Frequency Domain Analysis

In its simplest form, any time series data varies over time. If the variation is periodic and the same pattern repeats itself over time, the data is periodic. A sine wave is a simple example of a periodic date. The time series data is periodic if and only if

$$s(t+T) = s(t) \qquad -\infty < t < +\infty$$ Equation 2.9

where the constant T is the period of the time series data. A general periodic data series can be represented by three parameters: amplitude (A), frequency (f), and phase (ø). The *amplitude* is the peak value of the data series over a period; this is typically measured in the unit of the data series. The *frequency* is the rate at which the signal repeats. An equivalent parameter is the period (T) of a signal, which is the amount of time it takes for one repetition; therefore, T=1/f. *Phase* is a measure of the relative position in time within a single period of a signal. The general sine wave can be written as

$$s(t) = Asin(2\pi ft + \emptyset$$ Equation 2.10

It can be shown, using Fourier analysis, that any periodic data series is made up of components at various frequencies, in which each component is a sinusoid. For example, the periodic data series

$s(t) = \sin(2\pi f_1 t + \frac{1}{3}\sin(2\pi(3f_{1)}t)$ Equation 2.11

contains two individual components of frequencies f_1 and $3f_1$.

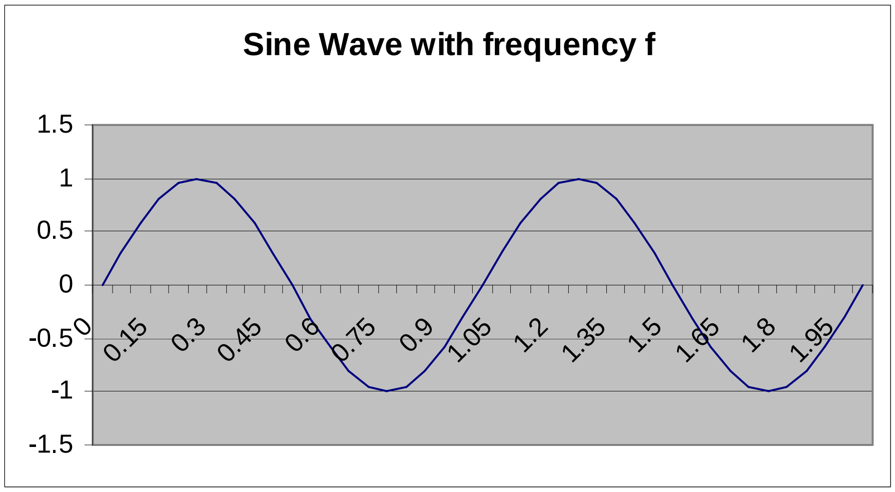


Figure 2.1 Sine wave with frequency $f_{1,}$ $\sin(2\pi f_1 t)$

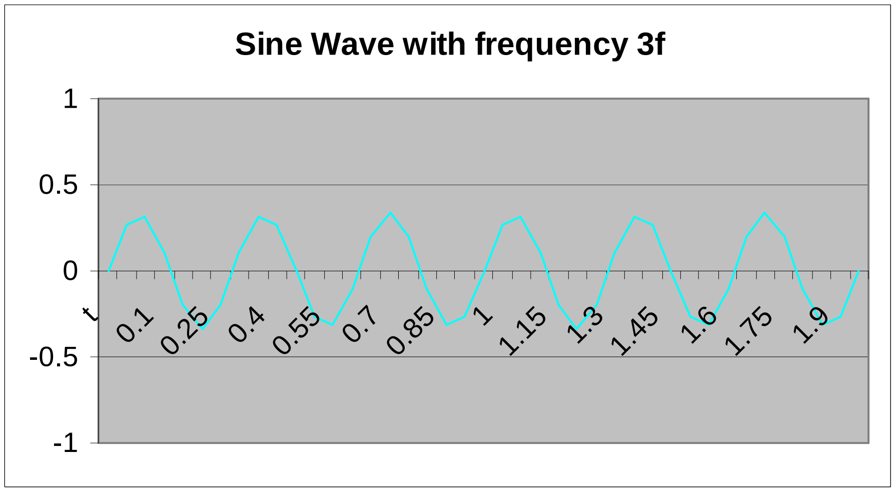


Figure 2.2 Sine wave with frequency $3f_{1,}$ $\sin(2\pi(3f)_1 t)$

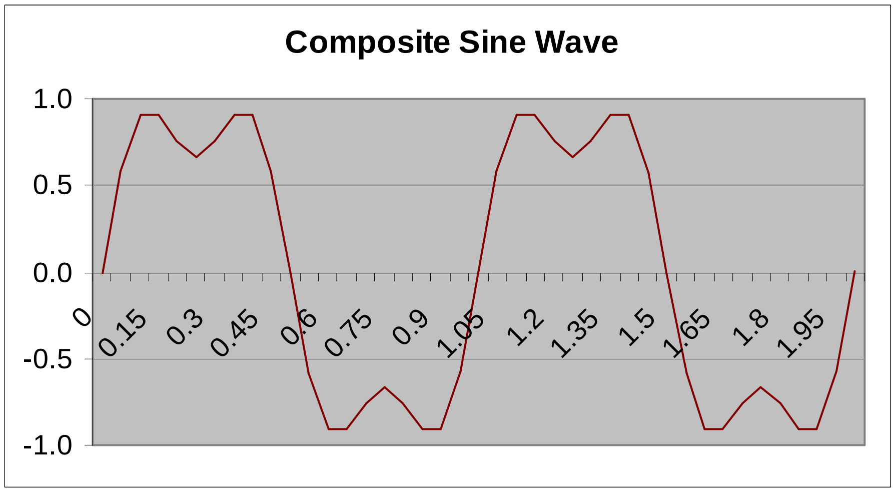


Figure 2.3 Composite sine wave, $\sin(2\pi f_1 t) + \frac{1}{3}\sin(2\pi(3f_1)t)$

There are two key points to note about this figure: (a) the second frequency is an integer multiple of the first frequency (when all of the frequency components of a signal are integer multiples of one frequency, the latter frequency is referred to as the fundamental frequency) and (b) the period of the total signal is equal to the period of the fundamental frequency.

It can be shown, using the discipline of Fourier analysis, that any signal is made up of components at various frequencies, in which each component is a sinusoid. Therefore, the effects of various time series data may be expressed in terms of frequencies (Stallings, 1997).

2.3.4.3 Fourier Analysis

Any period signal can be represented as a sum of sinusoids, known as a Fourier series.

$$x(t) = \sum_{n=0}^{\infty} a_n \cos(2\pi n f_0 t) + \sum_{n=1}^{\infty} b_n \sin(2\pi n f_0 t)$$ Equation 2.12

where f_0 = 1/T, where T is the period of the signal. The frequency f_0 is referred to as the *fundamental frequency*; multiples of f_0 are referred to as *harmonics*. Thus, a periodic signal of period T consists of the fundamental frequency f_0 plus harmonics of that frequency.

The value of the coefficients is calculated as follows:

$a_0 = \frac{1}{T}\int_0^T x\,(t)dt$ Equation 2.13

$a_n = \frac{2}{T}\int_0^T x\,(t)\cos\,(2\pi f_o t)dt$ Equation 2.14

$b_n = \frac{2}{T}\int x(t)\sin\,(2\pi f_0\, t)dt$ Equation 2.15

Fourier analysis techniques have been applied in time series forecasting, usually to improve results obtained from using other techniques. Saito et al. (2004) proposed a new demand forecasting method using ANN and Fourier Transform. In this method, the time series data of sales is transformed into a combination of frequency data and identified from objective indexes, which consist of product properties or economic indicators. The results demonstrated that the proposed method was superior to forecasting that uses only the Neural Network. The reason for superior results was attributed to the fact that the complex relationship between the objective indexes and the sales amount data was easier to learn by replacing the sales amount data with the frequency parameter.

In this research, Fourier analysis is used to extract the dominant usage cycles of engineering software used in high technology product development.

2.4 Software License Capacity Planning

Capacity is the amount of output that a system is capable of achieving over a specified period of time. Insufficient capacity may result in a decline in the level of service, while excess capacity may result in unproductive assets. Managing capacity, therefore, is a balancing act and must be aligned with the organization’s stated objectives and policies.

The discipline of capacity planning has been applied extensively in the manufacturing and service industry. This section covers some applications of this discipline in the high technology area. Forecasting the demand for each individual project allows for the development of a capacity plan for software license capacity for a software package used in high technology product development. Once there is a forecast of future demand for each ongoing project and there has been a strategic decision to meet this demand, the next step is to make available adequate software capacity. This is the function of software license capacity planning.

The literature on capacity planning is directed at three planning levels: strategic, tactical, and operational. Most of the research is concentrated on the tactical and operational levels, using analytical modeling (Yang, Haddad, and Chow, 2001). Laguna (1998) examines expanding the capacity of a single facility of a telecommunication network. The approach uses dynamic programming and the shortest path procedure. A stochastic dynamic programming model is deployed to determine the amount of current technology needed to meet future growth in demand (Rajagopalan, Singh, and Morton, 1998). Kennington et al. (1999) use a node-arc network programming model for the spare capacity allocation problem in a self-healing, mesh SONET telecommunication network.

Strategic problems tend to be broader and more complex in scope to allow for ease of analytical modeling. Simulation modeling has been used to model strategic aspects of capacity planning (Yang, Haddad, and Chow, 2001). Simulation involves designing a model that captures the essential features of a real world system as much as possible. Meimban et al. (1992) use a probabilistic investment model and Monte Carlo simulation to evaluate the feasibility of various design options of wood-fired cogeneration facility.

An approach to developing a capacity planning model for multimedia service systems was proposed by Park and Kim (2002). Multimedia systems are deployed by network access providers, such as public switched telephone networks (PSTN), cable TV networks' asymmetric digital subscriber lines (ADSL), or cable modems, to provide services such as video on demand. The capacity plan model was developed using an open queuing network. Based on the arrival rates of the clients and the failure rates of the multimedia resources, a model of the service capacity of the system was developed. The analytically developed model was validated by using a simulation approach.

2.4.1 Software Licensing

A license checkout takes place every time a user invokes an application and the Application Software queries the license server for a license. If a license is available, it is granted, and a checkout is recorded. When the application is terminated, the license is checked back into the license pool and is available to any other user who may invoke the same application. In an enterprise, a large number of users are typically checking in and out the licenses for any given application. The objective is to forecast demand to ensure that sufficient licenses have been procured to meet the maximum expected demand. A license denial may or may not be acceptable, depending on the policy in place for the given project. Therefore, the key statistic to be monitored is the peak demand for the time period of interest.

The selected time period for a single data point for this research was one week. A log was kept of all licenses checked in and out during the week, and the peak usage was recorded. Thus, the license usage history consists of a sequence of discrete numbers over time even though the licenses may be used continuously.

CHAPTER 3

SOFTWARE LICENSE USAGE FORECASTING MODEL

The first objective of this chapter is to propose a usage model for the EAS (Engineering Application Software) that is used concurrently on multiple projects. As described in chapter 2, usage data is collected on a weekly basis. Each data point represents peak usage for the week. The usage data is used to develop models to predict project usage.

The second objective is to utilize the proposed usage model to predict the future license requirements. In this chapter, we detail the methodology used to collect usage data and construct project usage models for the projects that are in execution concurrently. In chapter 4, the concurrent project usage models, in conjunction with capacity data are utilized to forecast capacity requirements.

To achieve the objectives outlined in the previous paragraph, the first step is to collect the software usage data for one EAS for a period of time that encompasses one complete project. Usage data is also collected for all of the other projects in execution during the same time period for the same EAS. The second step uses a segment of the usage data to propose a rigorous approach to establish the dominant usage cycles for each project and to build models to predict the EAS demand. The last step is to analyze and validate the models with the remaining usage data. This will be the topic of discussion for chapter 5. Figure 3.1 shows the methodology to develop the ESA usage models and a capacity plan.

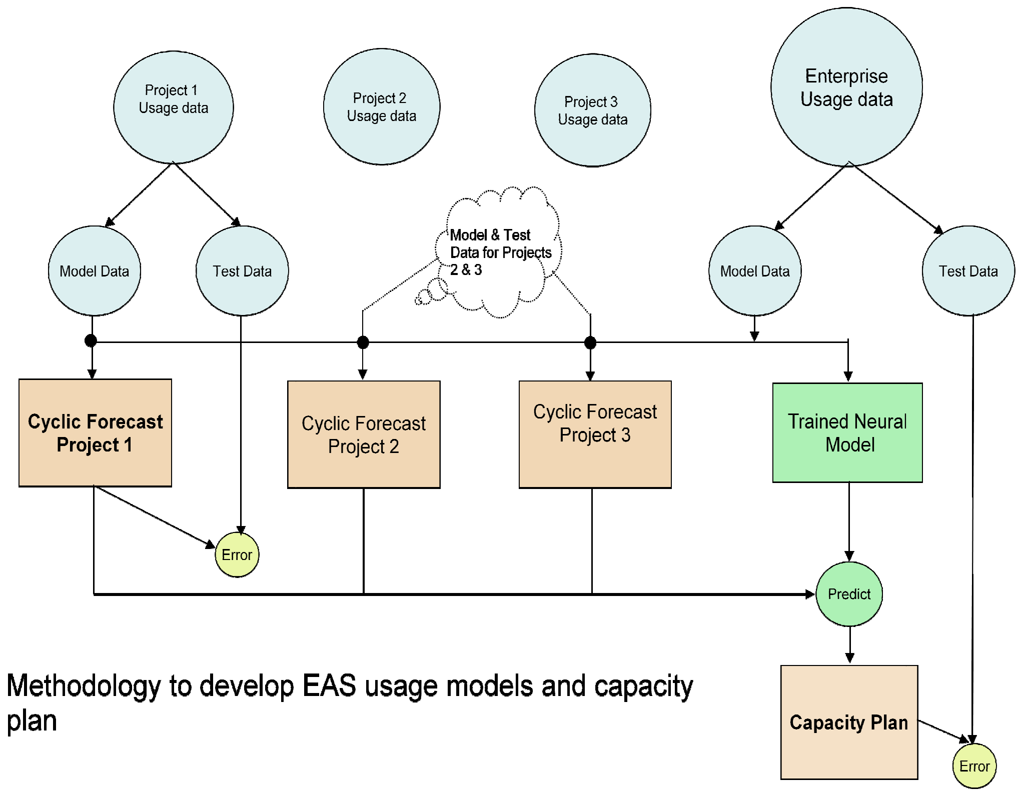


Figure 3.1 Proposed Methodology

Figure 3.2 is a detailed flowchart that shows how the methodology to build a forecast model for a single project is implemented. The following sections discuss in detail the process steps outlined in the flowchart.

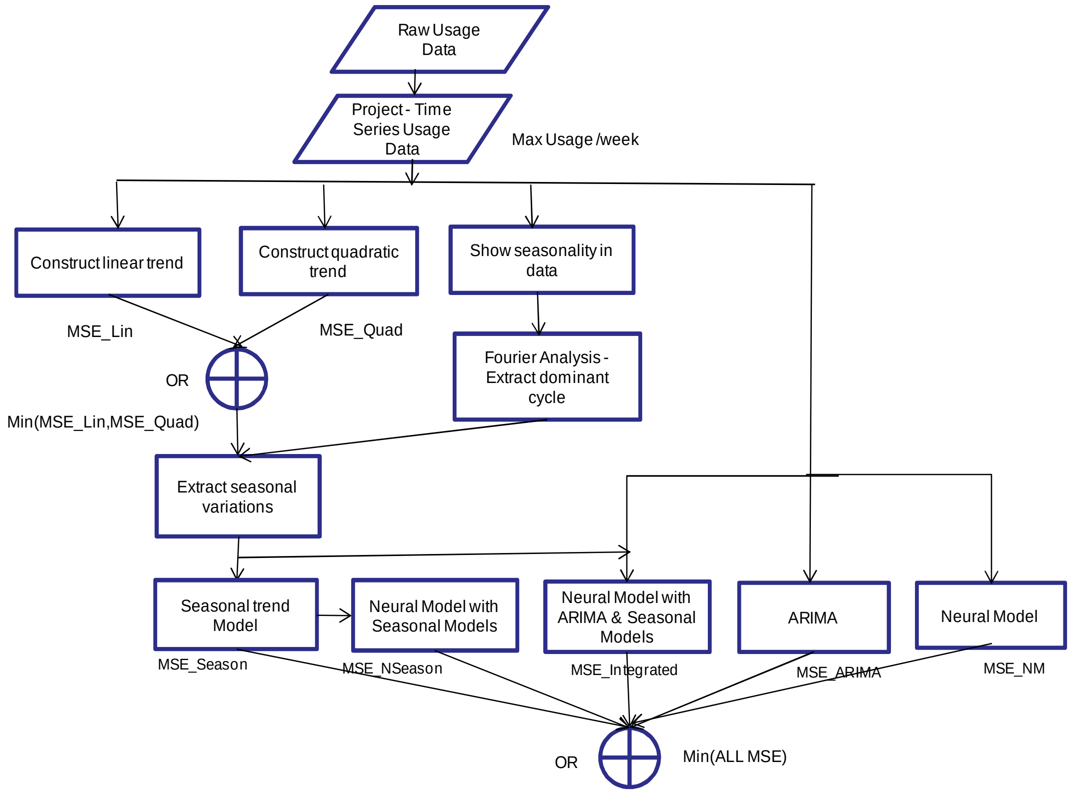


Figure 3.2 Project License Usage Methodology

3.1 Software Usage Data Collection

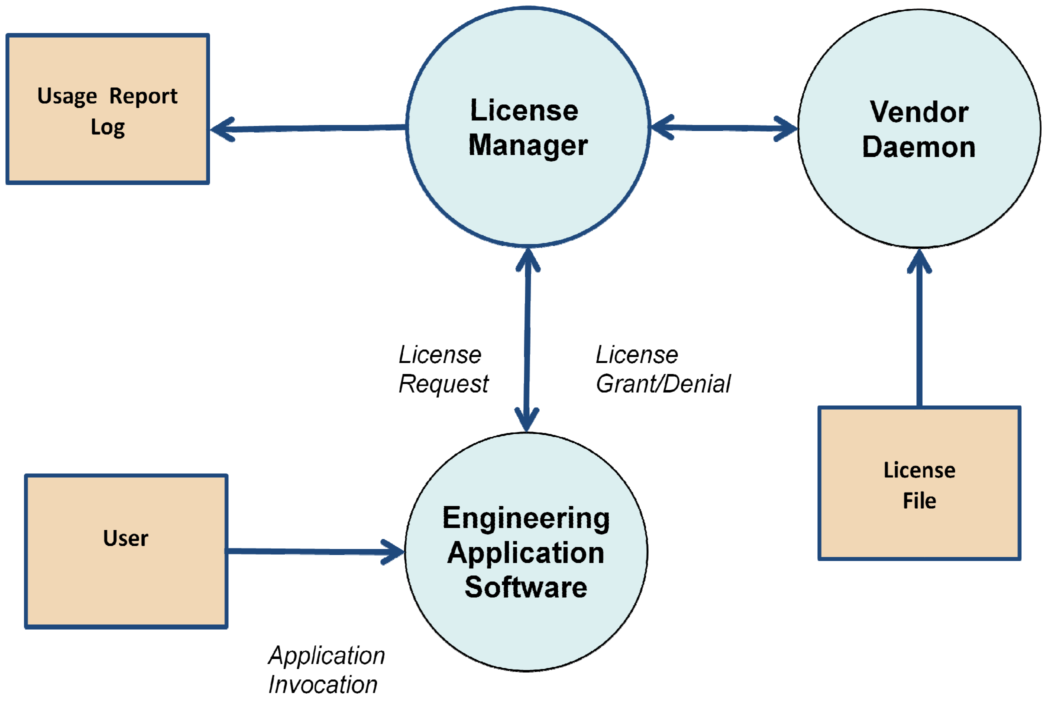
Usage data is essentially a record of the license check-out and check-in time of an Application Software.

Check-out time – check-in time = time duration of software usage

When Application Software is invoked, it requests a license from a License Manager program. The License Manager software runs perpetually in the background on a License Server. It is the function of the License Manager to grant or deny a license request from an EAS.

To assist the License Manager, the publisher of the EAS provides another background program commonly referred to as the Vendor Daemon. The key function of the Vendor Daemon is to track the total number of licenses available and the number that have been checked out.

When requested by the License Manager, the Vendor Daemon uses an established protocol to communicate this information.



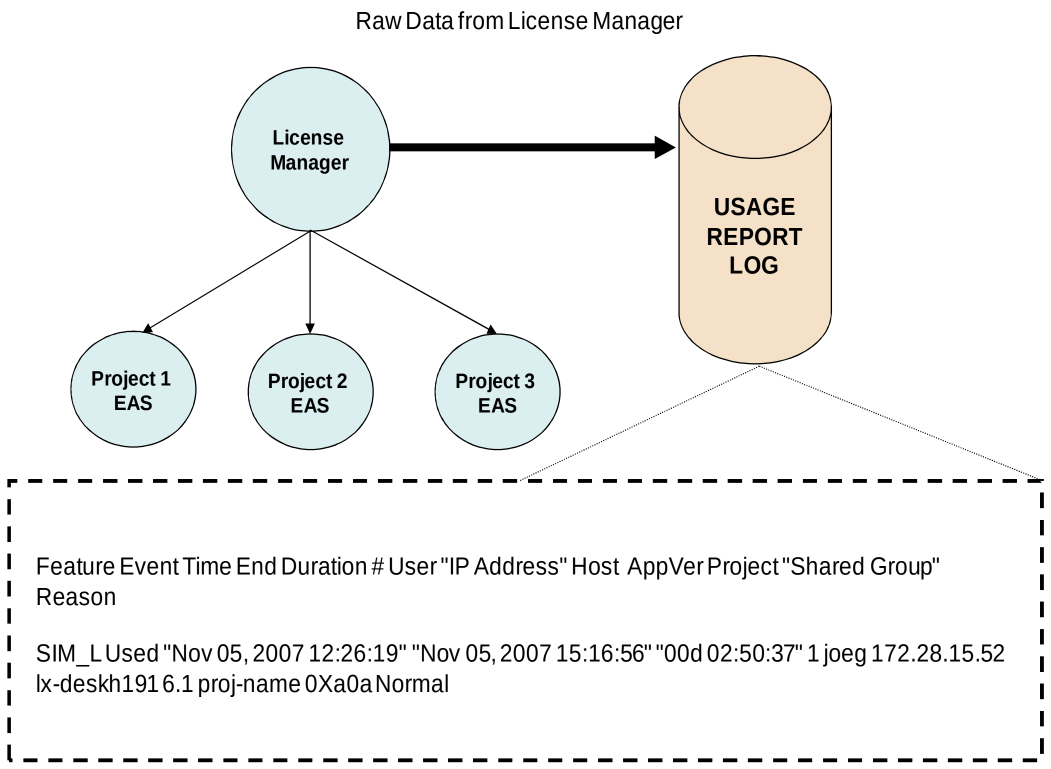
License server Architecture

Figure 3.3 EAS License Architecture

The License Manager grants or denies the license based on the information received from the Vendor Daemon. A license grant means that the EAS is authorized by the Software Publisher and that licenses are available. A denial means that either all of the available licenses are checked out or the requested EAS is unauthorized. A license check-out happens when a license request results in a license grant. A check-in occurs when an EAS relinquishes the license.

3.1.1 Raw Data

The License Manager records each check-in and check-out with the appropriate date and time stamp in a Usage Report Log. The Usage Report Log contains the raw data that is then read into a database for further processing.



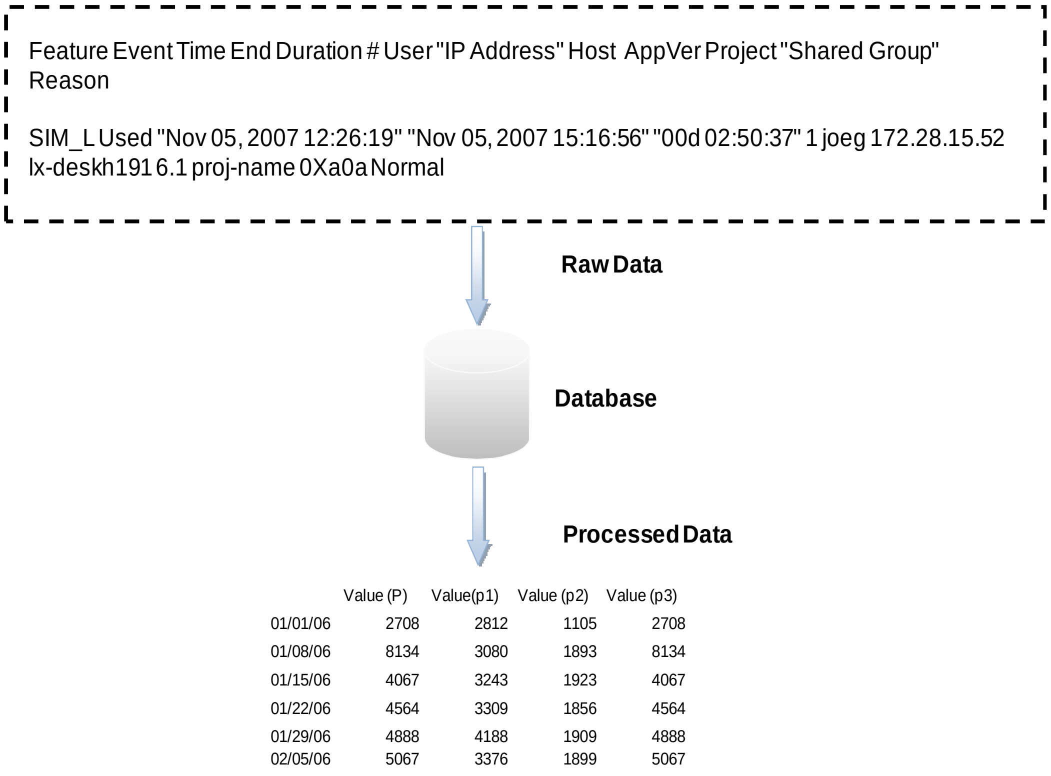
```
Feature Event Time End Duration # User "IP Address" Host AppVer Project "Shared Group"
Reason

SIM_L Used "Nov 05, 2007 12:26:19" "Nov 05, 2007 15:16:56" "00d 02:50:37" 1 joeg 172.28.15.52
lx-deskh191 6.1 proj-name 0Xa0a Normal
```

Figure 3.4 Raw Data Format

3.1.2 Processed Data

The raw data is processed to compute the weekly peak usage for each week for the two-year period under study for the EAS. The data is exported to a Microsoft Excel spreadsheet. Figure 3.4 shows the generation of processed data from raw data.



```
Feature Event Time End Duration # User "IP Address" Host AppVer Project "Shared Group"
Reason

SIM_L Used "Nov 05, 2007 12:26:19" "Nov 05, 2007 15:16:56" "00d 02:50:37" 1 joeg 172.28.15.52
lx-deskh191 6.1 proj-name 0Xa0a Normal
```

	Value (P)	Value(p1)	Value (p2)	Value (p3)
01/01/06	2708	2812	1105	2708
01/08/06	8134	3080	1893	8134
01/15/06	4067	3243	1923	4067
01/22/06	4564	3309	1856	4564
01/29/06	4888	4188	1909	4888
02/05/06	5067	3376	1899	5067

Figure 3.5 Processed Data Generation

3.2 License Usage Cycles

As a first step, to ensure that the data is not random and to visualize the trend and cyclical components we create a correlogram that shows the autocorrelation of the time series data with different time lags. In the analysis of the time series, a correlogram, also known as an autocorrelation plot, is a plot of the sample autocorrelations r_h versus h, the time lags. The correlogram is a tool commonly used for checking randomness in a data set. This randomness is ascertained by computing autocorrelations for data values at varying time lags. If random, such autocorrelations should be near zero for any and all time lag separations. If non-random, then one or more of the autocorrelations will be significantly non-zero. Figures 3.6 to 3.8 show the non-random nature of the usage data for all three projects.

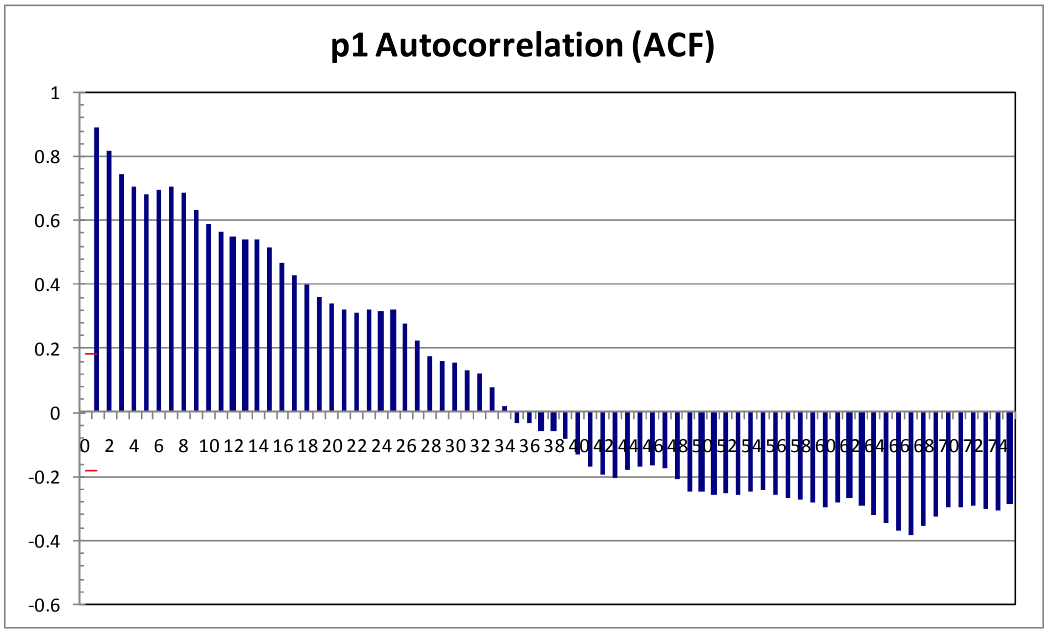


Figure 3.6 Correlogram of p1 project usage

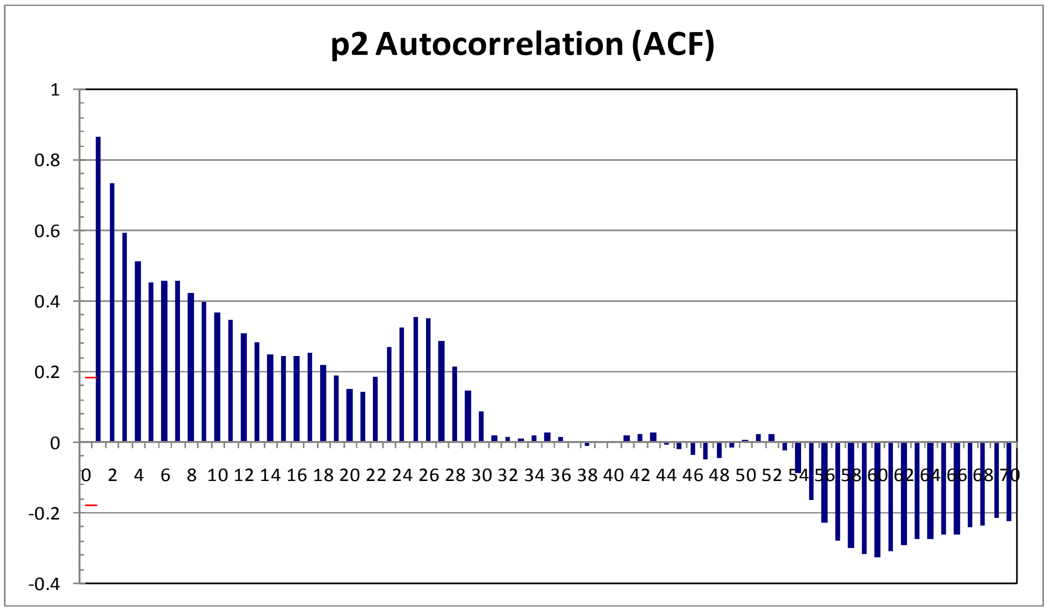


Figure 3.7 Correlogram of p2 project usage

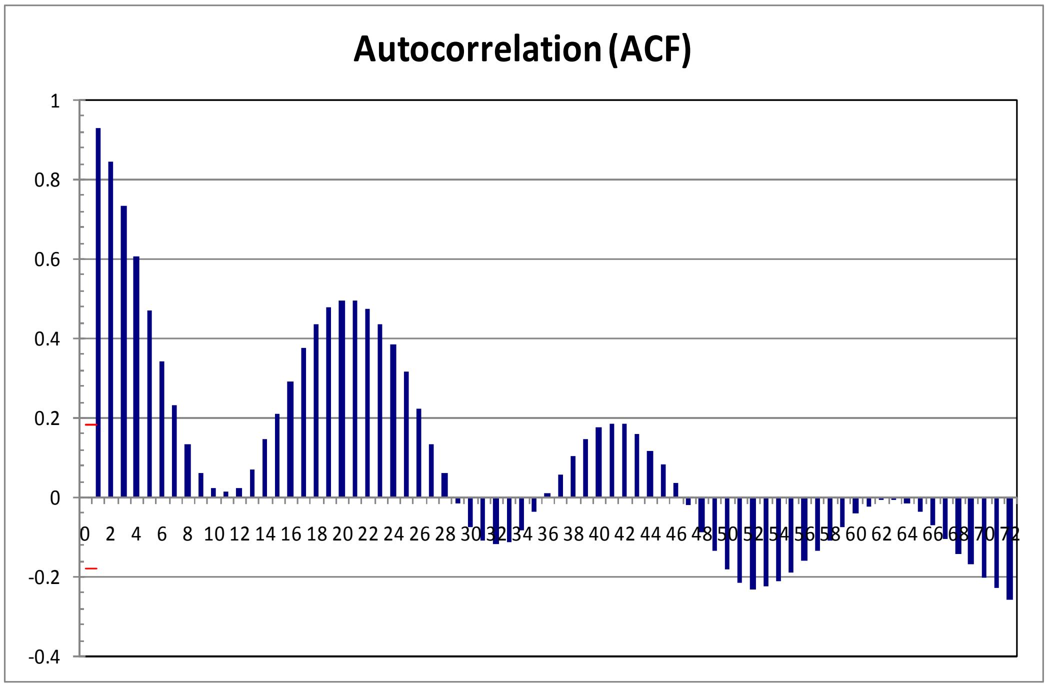


Figure 3.8 Correlogram of p3 project usage

We propose to extract the cyclical behavior over time of the weekly peak license usage variable to predict future usage of licenses. This is to ensure that an adequate number of licenses are available to meet the project's requirements. We show that Fourier Analysis is an effective tool to capture the cyclical behavior that could be due to several factors discussed in chapter 2, section 2.2. Key attributes of high technology product development projects are that the development tasks may be either iterative to drive convergence to the desired results or repetitive to respond to changes in the specifications, requirements, and evolving knowledge of the problem domain. Executing various EAS at each step is usually required to complete the product development in high technology. Due to the attributes of high technology projects, an EAS may be executed periodically and repeatedly, thus exhibiting cyclical usage patterns.

The Fourier analysis is a form of multiple regression analysis. The cycle is described by a sine function with the general form of

$$y = a_0 + \sum_{j=1}^{J}\left(a_j \cos(2\mathrm{j}\pi\mathrm{t}) + b_j \sin(2\pi\mathrm{jt}) + \epsilon\right) \qquad \text{Equation 3.1}$$

where

y = weekly peak license usage parameter

a_0 = intercept of the regression equation

a_j = coefficient of the cosine variable

b_j = coefficient of the sine variable

Є = residual term

For the cyclical variation over a period of time equal, the function takes the form

$$y = a_0 + \sum_{j=1}^{J} \left(a_j \cos(\tfrac{2j\pi t}{365}) + b_j \sin(\tfrac{2\pi jt}{365}) + Є\right) \qquad \text{Equation 3.2}$$

where t = Julian days

y = weekly peak license usage parameter

We will use this form to analyze the cyclical variation of license usage.

3.2.1 Extraction of Cycles

To extract the dominant cycles, the spreadsheet is structured in the Excel spreadsheet software as shown in Table 3.1.

Table 3.1 Snapshot of Excel spreadsheet Fourier Analysis

								Value	Value(	Value	Value(				
01/01/00	**t**		**2pi*t/365**	**cos(e)**	**sin(e)**	**cos(2e)**	**sin(2e)**	**(P)**	**p1)**	**(p2)**	**p3)**	**reg(P)**	**reg(p1)**	**reg(p2)**	**reg(p3)**
01/01/06	38717		666.482426	0.893919	0.448229	0.59818091	0.801361	2708	2812	1105	2708	18176.90354	6167.044347	4420.6076	18176.904
01/08/06	38724		666.602926	0.833556	0.552435	0.38963045	0.920971	8134	3080	1893	8134	17572.45189	5963.294387	4243.4868	17572.452
01/15/06	38731		666.723425	0.761104	0.64863	0.15855939	0.987349	4067	3243	1923	4067	16959.03008	5768.770086	4133.1818	16959.03
01/22/06	38738		666.843924	0.677615	0.735417	-0.0816764	0.996659	4564	3309	1856	4564	16353.27341	5593.423411	4092.7854	16353.273
01/29/06	38745		666.964424	0.584298	0.811539	-0.3171913	0.948362	4888	4188	1909	4888	15771.56851	5446.482266	4121.6543	15771.569
02/05/06	38752		667.084923	0.482508	0.875892	-0.5343726	0.845249	5067	3376	1899	5067	15229.37631	5335.93027	4215.4891	15229.376
02/12/06	38759		667.205423	0.37372	0.927542	-0.7206671	0.693281	4866	4531	1923	4866	14740.60176	5268.052699	4366.6211	14740.602
02/19/06	38766		667.325922	0.259512	0.96574	-0.8653073	0.501242	5512	4204	2038	5512	14317.04299	5247.074714	4564.4885	14317.043
02/26/06	38773		667.446422	0.14154	0.989932	-0.9599327	0.280231	5670	4717	2318	5670	13967.94797	5274.912768	4796.2728	13967.948
03/05/06	38780		667.566921	0.021516	0.999769	-0.9990741	0.043022	5805	4634	2412	5805	13699.70095	5351.053628	5047.6599	13699.701
03/12/06	38787		667.687421	-0.09882	0.995105	-0.9804692	-0.19667	5911	4589	2317	5911	13515.6534	5472.568148	5303.6811	13515.653

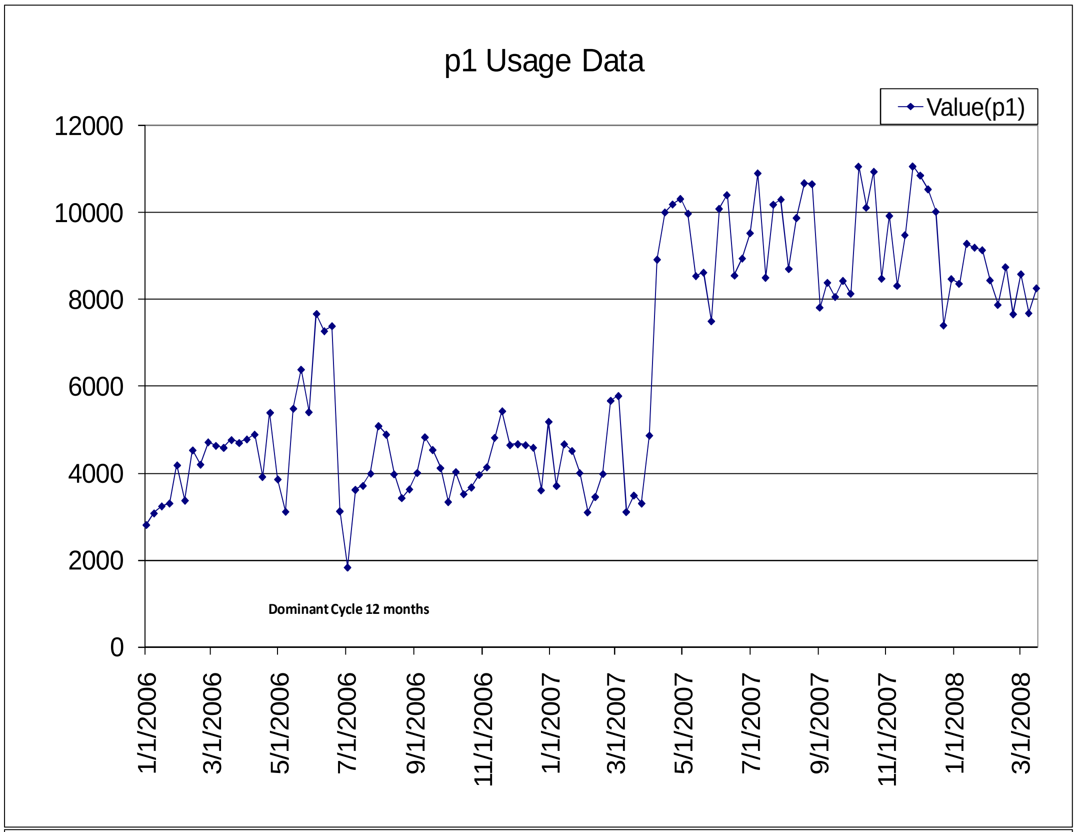
01/01/00 is the reference date used in the calculations; column t shows the Julian days. The formulas used to compute the values for the Fourier analysis in the Excel spreadsheet are shown below in Table 3.2.

Table 3.2 Formula view of Excel spreadsheet Fourier analysis

1	t		2pi*t/365	cos(e)	sin(e)	cos(2e)	sin(2e)
38718	=A2-A1		=2*PI()*B2/365	=COS(D2)	=SIN(D2)	=COS(2*D2)	=SIN(2*D2)
=A2+7	=A3-A1		=2*PI()*B3/365	=COS(D3)	=SIN(D3)	=COS(2*D3)	=SIN(2*D3)
=A3+7	=A4-A1		=2*PI()*B4/365	=COS(D4)	=SIN(D4)	=COS(2*D4)	=SIN(2*D4)
=A4+7	=A5-A1		=2*PI()*B5/365	=COS(D5)	=SIN(D5)	=COS(2*D5)	=SIN(2*D5)
=A5+7	=A6-A1		=2*PI()*B6/365	=COS(D6)	=SIN(D6)	=COS(2*D6)	=SIN(2*D6)
=A6+7	=A7-A1		=2*PI()*B7/365	=COS(D7)	=SIN(D7)	=COS(2*D7)	=SIN(2*D7)

The next step in the analysis is to perform the multiple linear regression (MLR) analysis to calculate the values for the coefficients a_0, a_j and b_j in equation 3.2. In the Excel spreadsheet software, we use the built-in Regression function under the “Data Analysis” menu. The values under the column headings cos(e), sin(e), cos(2e), and sin(2e) are the independent variables. For each of the three projects, the historical actual usage data is under the column headings Value(p1), Value(p2), and Value(p3). Value(P) is the historical actual usage for the enterprise during the time period in which the three projects p1, p2, and p3 were active. The predicted values for each of the three projects are under the column headings reg(p1), reg(p2), and reg(p3).

We use Fourier Analysis to capture the cyclical component of the EAS usage on each project. Equation 3.2 was expanded for the values of j=1 and j=2. The value of j=1 gives the longest cycle that can be extracted from the data. Similarly, the value of j=2 gives the next cycle of shorter length. This process may be continued for subsequent values of j=3, j=4, etc. as long as the period of the cycle is relevant in the context of the data. The graphs in Figure 3.5, Figure 3.6, and Figure 3.7 show the dominant cycle when j=1 and the next cycle when j=2. The cycles are plotted with the actual license usage data.



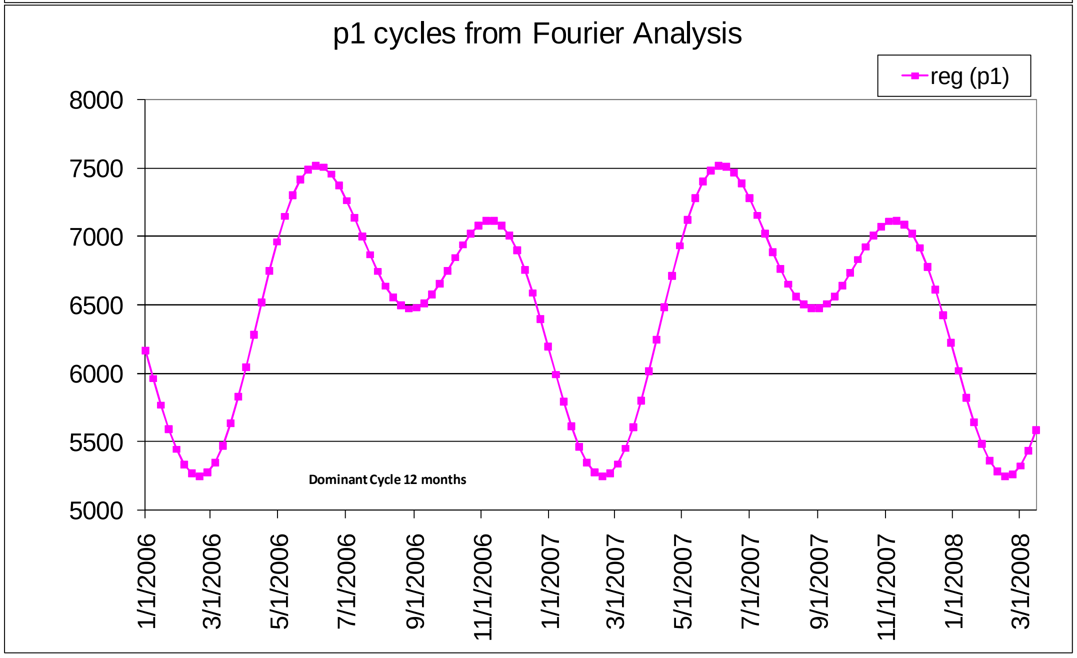
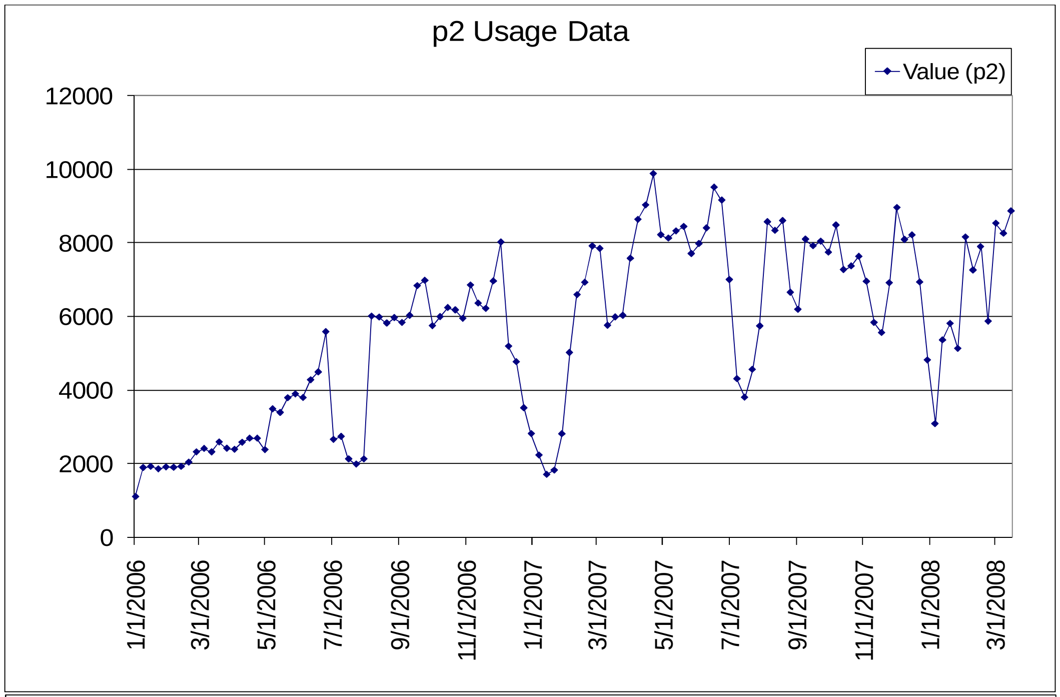


Figure 3.9 Graph of usage cycles and actual usage for project p1.



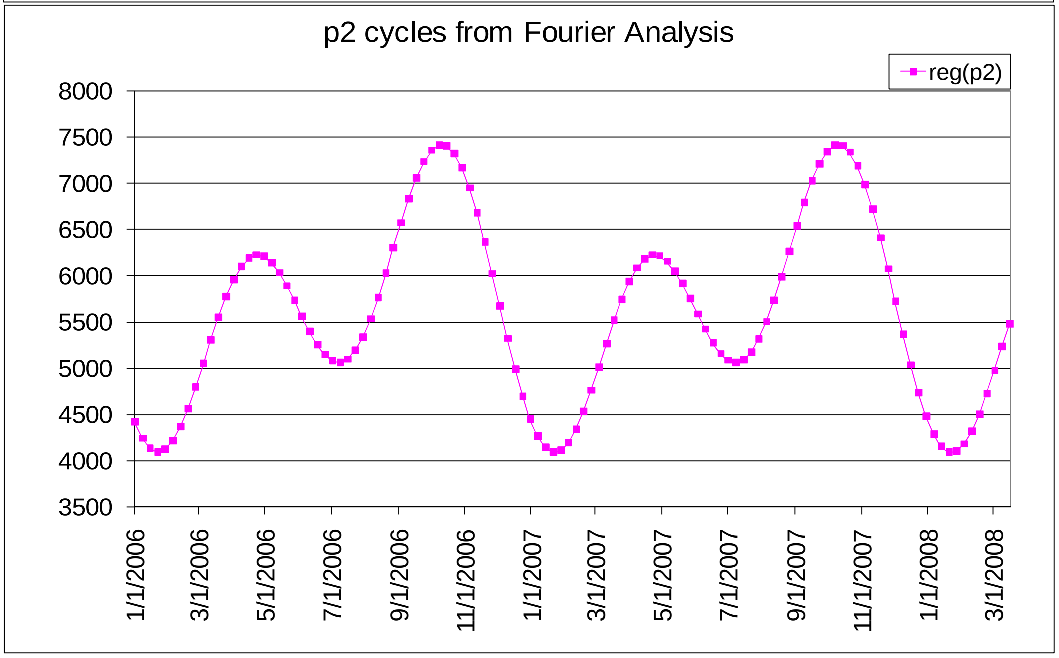
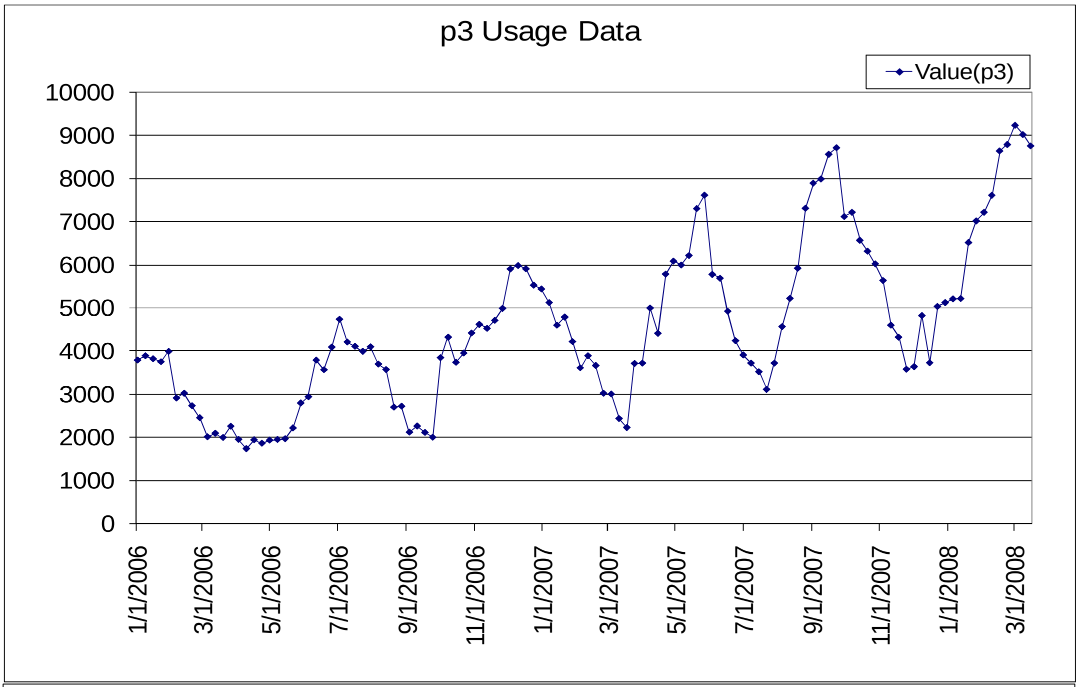


Figure 3.10 Graph of usage cycles and actual usage for project p2.



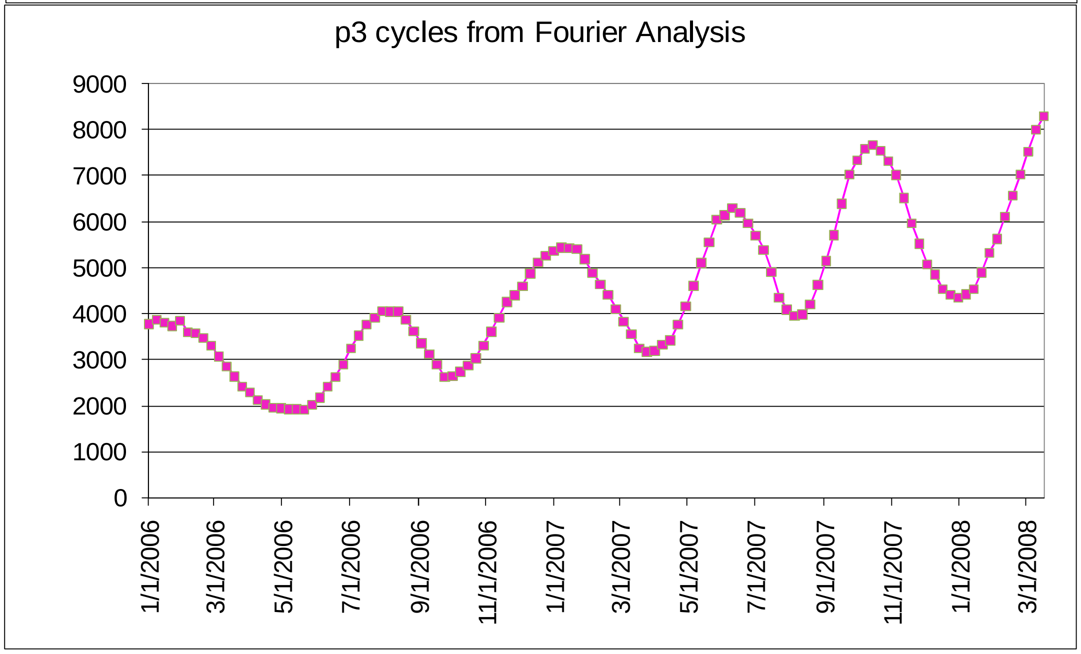


Figure 3.11 Graph of usage cycles and actual usage for project p3.

From the graphs we make the following observations about the first dominant cycle and the second minor cycles for each of the three projects.

Table 3.3 Cycles in EAS usage

Projects	**Start Date**	**End Date**	**Duration**	
P1	3/1/2006	3/1/2007	12 months	Dominant cycle – 12 mo
	3/1/2007	3/1/2008	12 months	
	3/1/2006	9/1/2006		Minor cycle – 6 mo
	9/1/2006	3/1/2007		
	3/1/2007	9/1/2007		
	9/1/2007	3/1/2008		
P2	2/1/2006	2/1/2007	12 months	Dominant cycle – 12 mo
	2/1/2007	2/1/2008	12 months	
	2/1/2006	8/1/2006		Minor cycle – 6 mo
	8/1/2006	2/1/2007		
	2/1/2007	8/1/2007		
	8/1/2007	2/1/2008		
P3	51/2006	10/1/2006		Dominant cycle – 5mo
	10/1/2006	3/1/2007		
	3/1/2007	8/1/2007		
	8/1/2007	1/1/2008		

3.3 Project Cycles Forecasts

We use trend forecasting techniques based on the actual usage data to predict future values. In the previous section, we established the presence of cyclical variations in the usage data. For forecasting, these cyclical variations can be treated the same way as trend forecasting with seasonal variations. Fourier analysis, utilized in the previous section to compute the cycles and the cycle period, establishes the length of the "seasons."

In the following sections, we develop forecasts by using linear and quadratic trends for project p1. Using the Mean Square Error (MSE) criteria, we select the trend that has the smaller MSE. We then apply the seasonal variations to obtain a better fit as measured by the MSE. The same methodology is used for projects p2 and p3.

3.3.1 Project Forecast using Linear and Quadratic Trends

The first step in creating a forecast is to formulate a trend that can then be extrapolated to predict the future value of the dependent variable. We illustrate the methodology for project p1. The same methodology is applied to projects p2 and p3, and the results are summarized in Appendix 3.

Consider the following linear regression model, where time is used as an independent variable.

$Y_t = \beta_0 + \beta_1 X1_t + \varepsilon_t$ Equation 3.3

where

$X1_t$= t and represents the time period t ($X1_1 = 1$, $X1_2 = 2$, and so on)

ε_t = error term that represents the random variation in the time series

β_0 = Y intercept of the regression line

β_1 = slope of regression line

Yt = dependent variable, whose systematic variation can be described by the regression function $\beta_0 + \beta_1 X1_t$.

The best estimate of □□ using the least squares to estimate the parameters in equation 3.3 for any time period t is:

$Y'_t = b_0 + b_1 X1_t$ Equation 3.4

where

Y'_t is the best estimate of Y_t.

b_0 = best estimate of β_0

b_1 = best estimate of β_1

To fit a curved trend line to the data, we use the following quadratic model:

$Y_t = \beta_0 + \beta_1 X1_t + \beta_2 X2_t + \varepsilon_t$ Equation 3.5

where

$X1_t$=t and $X2_t$=t^2

β_2 = quadratic regression parameter

The rest of the terms are the same as in Equation 3.3.

The resulting estimated regressions function for this model is:

$Y't = b_0 + b_1 X1_t + b_2 X2_t$ Equation 3.6

where

b_2 = is the best estimate of β_2

The rest of the terms are the same as in Equation 3.4.

Table 3.4 Excel setup to compute Linear and Quadratic trends

Date	Cycle (p1)	t	t^2	Actual Usage (p1)	Linear Trend	Quadratic Trend
01/01/06	4	1	1	2812	2988	3416
01/08/06	4	2	4	3080	3049	3454
01/15/06	4	3	9	3243	3109	3493
01/22/06	4	4	16	3309	3169	3532
01/29/06	4	5	25	4188	3230	3571
02/05/06	4	6	36	3376	3290	3610
02/12/06	4	7	49	4531	3350	3650

The columns in Table 3.4 represent:

Linear Trend = Y'_t, the best estimate of Y_t

Actual Usage (p1) = Y_t, dependent variable

t = independent time variable

t^2 = squared value of t

The Linear and Quadratic trends are calculated using the equations 3.4 and 3.6 and plotted against the actual usage as shown in figure 3.8. The mean MSE for the linear trend is 2953683 and the MSE for the quadratic trend is 2915088. Therefore:

MSE(Quadratic Trend) < MSE(Linear Trend)

We elect to apply the seasonal or cyclic adjustments to the quadratic trend to further decrease the MSE.

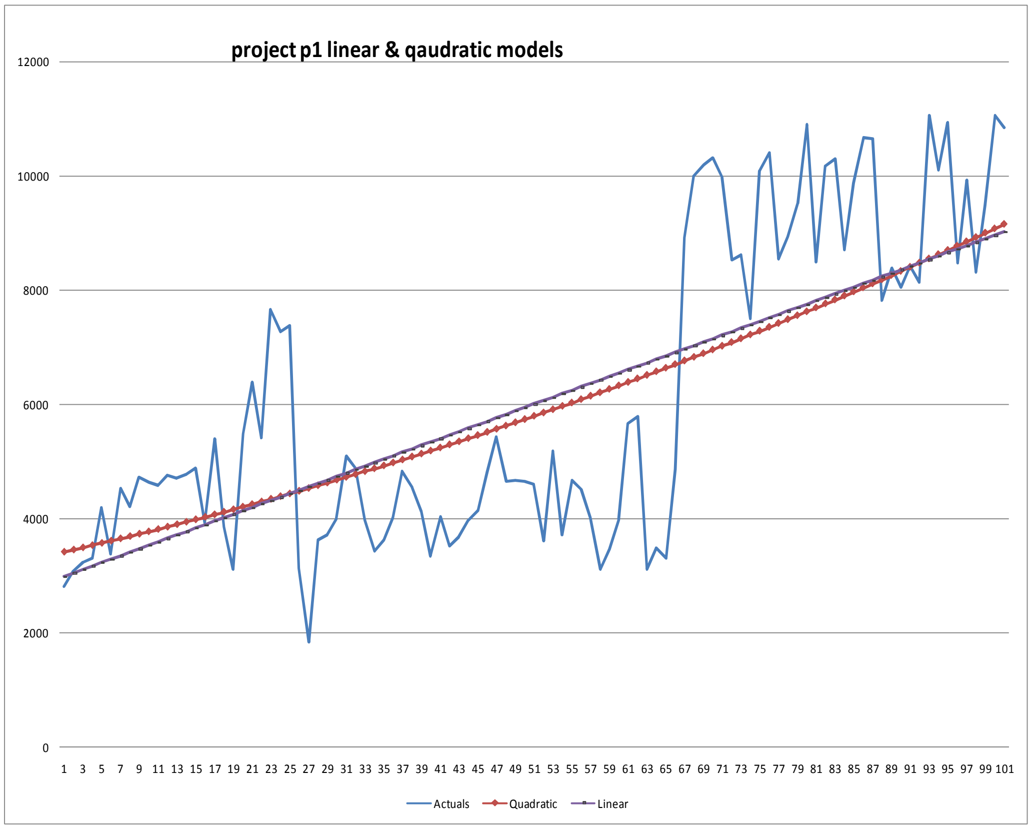


Figure 3.12 Linear and Quadratic trends against actual usage

3.3.2 Project Forecast using Cyclic Variations

In this section, we attempt to further refine the predictions by including the effect of cyclic variation in the usage of EAS. We incorporate the effect of cycles in trend forecasting, similar to the effect of seasons in forecasting trends. To use the seasonal approach, we view the seasons as phases of the cycles computed using the Fourier analysis approach. Refer to Figure 3.9, which shows the four phases of the cycles in project p1. For the purpose of modifying the trend to include the effects of the four phases, we consider these phases to be analogous to seasonality effects.

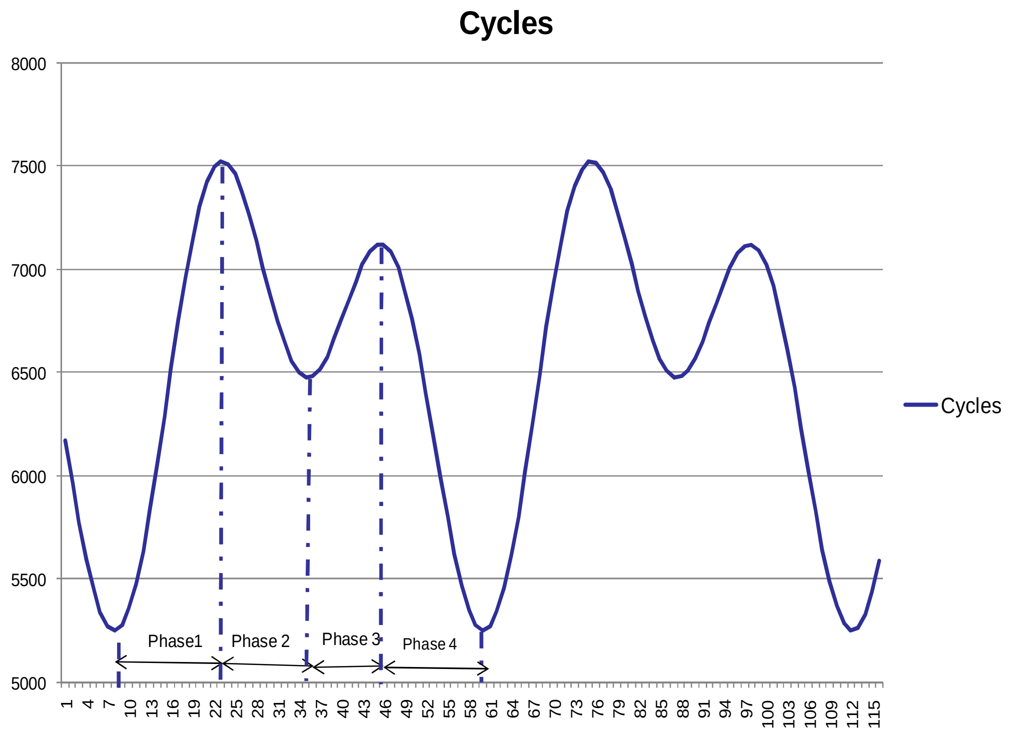


Figure 3.13 Phases of the cycles for seasonal effect

Table 3.5 Excel setup for forecasting with seasonal effects

Date	Cycle phase	t	t^2	Actual Usage (p1)	Quadratic Trend	**Actual as % of trend**	Seasonal Trend	Cycle Phase	Seasonal Index
01/01/06	4	1	1	2812	3416	82%	2998	1	108.67%
01/08/06	4	2	4	3080	3454	89%	3031	2	113.72%
01/15/06	4	3	9	3243	3493	93%	3065	3	93.55%
01/22/06	4	4	16	3309	3532	94%	3099	4	87.74%
01/29/06	4	5	25	4188	3571	117%	3133		
02/05/06	4	6	36	3376	3610	94%	3168		

In Table 3.5, under the column “Actual as % of trend,” we calculate the % deviation of the Quadratic Trend value from the actual usage. For each cycle phase, 1 through 4, we calculate the average of the deviation listed under the column “Seasonal Index.” Each Quadratic Trend value is multiplied by the Seasonal Index of the corresponding phase. The new trend forecast is shown in Figure 3.10 and gives a MSE of 2451997.

MSE(Quadratic Trend with seasonal effects) < MSE(Quadratic Trend)

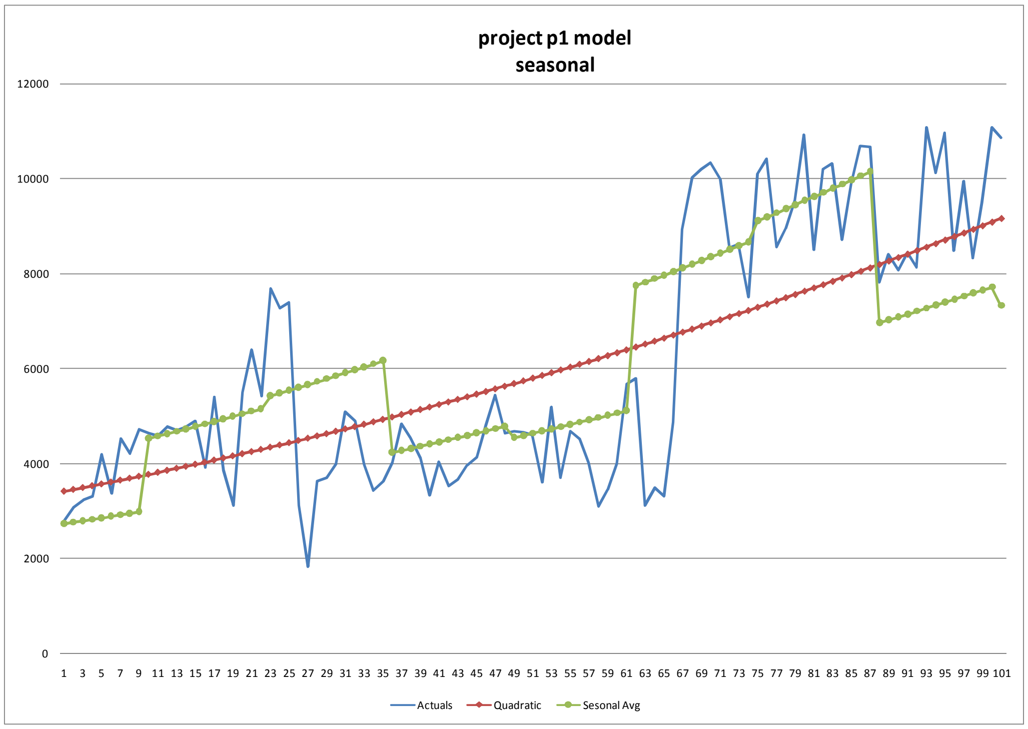


Figure 3.14 Quadratic trend with seasonal variation

In the next section, we use an ARIMA model to investigate if we can achieve a lower MSE.

3.3.3 Project Forecast using ARIMA

ARIMA models can be described by a series of equations. The equations are somewhat simpler if the time series is first reduced to zero-mean by subtracting the sample mean. Therefore, the *mean-adjusted* series is represented by

$$y_t = Y_t - \bar{Y}, \qquad t = 1, \ldots..N \qquad \text{Equation 3.7}$$

where

Y_t is the original time series

$\bar{Y}$ is the sample mean

y_t is the mean-adjusted series.

One subset of ARMA models are so-called autoregressive, or AR, models. An AR model expresses a time series as a linear function of its past values. The order of the AR model tells how many lagged past values are included. The simplest AR model is the first-order autoregressive, or AR(1), model. The equation for this model is

$$y_t + a_t y_{t-1} = e_t \qquad \text{Equation 3.8}$$

where

y_t is the mean-adjusted series in t

y_{t-1} is the series in the t-1

a_t is the lag-1 autoregressive coefficient

e_t is the residual.

The residuals $\square_{\square}$ are assumed to be random in time (not autocorrelated) and normally distributed. By rewriting the equation for the AR(1) model as

$$y_t = -a_t y_{t-1} + e_t \qquad \text{Equation 3.9}$$

we see that the AR(1) model has the form of a regression model in which y_t is regressed on its previous value and e_t is analogous to the regression residuals. The name autoregressive refers to the regression on self.

The *moving average* (MA) model is a form of ARMA model in which the time series is regarded as a moving average of a random shock series, $\square_{\square}$. The first order moving average, or MA(1), model is given by

$$y_t = e_t + c_1 e_{t-1} \qquad \text{Equation 3.10}$$

where

e_t and e_{t-1} are the residuals at times t and t-1 and

c_1 is the first-order moving average coefficient

The autoregressive model includes lagged terms on the tie series itself, and the moving average model includes lagged terms on the residuals. By including both types of lagged terms, we arrive at the *autoregressive integrated moving average* (ARIMA) models. The order of the

ARIMA model is included in parentheses as ARIMA(p,q), where p is the autoregressive order and q is the moving-average order. The simplest and most frequently used ARIMA model is ARIMA(1,1) model

$$y_t + a_1 e_{t-1} = e_t + c_1 e_{t-1} \qquad \text{Equation 3.11}$$

In this approach, we use ARIMA on the project p1 usage data using a built-in Excel macro. Table 3.6 shows the setup in Excel build an ARIMA model of autoregressive order (p) of 1 and moving average order (q) of 1. We obtain the graph shown in Figure 3.15.

The critical value of t for degrees of freedom equal to 100 and α equal to 0.05 is 1.9840. Since the t-statistic at $78.11231596 > t_{0.05,100} = 1.984$, we reject the null hypothesis that there is no correlation between autoregressive variable AR(1) and the fitted model. Similarly, we reject the null hypothesis that there is no correlation between the moving average variable MA(1) and the fitted model. The R-squared value of 0.800132 shows that 80% of the variation in the model can be explained by the variables AR(1) and MA(1).

Table 3.6 Excel setup for an ARIMA model

p1 ARIMA

timeseries: y
Method: Nonlinear Least Squares (Levenberg-Marquardt)
date: 02-21-09 time: 12:17
Included observations: 102
p = 1 - q = 1 - no constant - manual selection

	Coefficient	Std. Error	t-Statistic	R-squared
AR(1)	0.996331853	0.01275512	78.11231596	0.800132
MA(1)	-0.214429051	0.09314654	2.302061344	

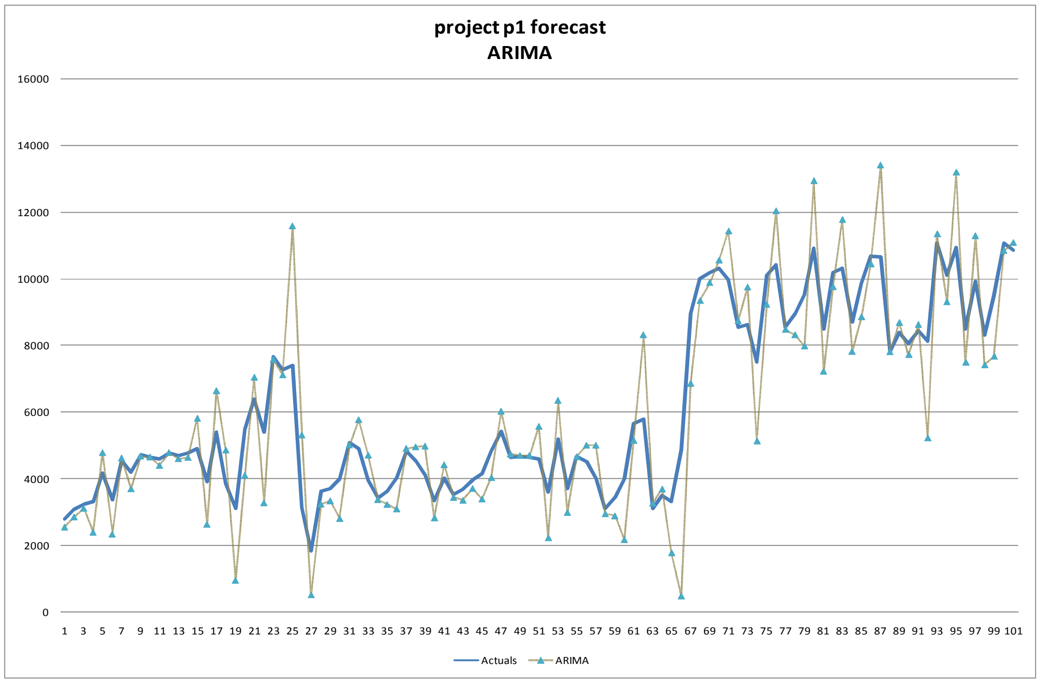


Figure 3.15 Plot of fitted ARIMA model against actual usage

The MSE for the ARIMA model is 1466147.

Therefore

MSE(ARIMA) < MSE(Quadratic Trend with seasonal effects)

Table 3.7 MSE values for projects p1, p2 and p3

MSE p1, p2 and p3

	MSE (Linear)	***MSE (Quadratic)***	***MSE (Seasonal)***	***MSE (ARIMA)***
p1	2,953,683	2,915,088	2,451,997	1,466,147
p2	2,920,601	2,869,720	2,248,934	1,264,494
p3	1,747,894	1,704,019	1,529,607	367,417

An inspection of Table 3.7 reveals that introducing seasonal variations improves the MSE for all three projects. For instance the value of MSE(seasonal) for p1 is 2,451,997, which is less than the MSE(Quadratic) of 2,915,088. Also, the ARIMA models give the least MSE for all three projects. The Linear, Quadratic, Seasonal, and ARIMA forecasting models for projects p2 and p3 are attached in Appendix C. Figures 3.16, 3.17, and 3.18 show the residual plots of the forecasting models for projects p1, p2, and p3. The plots show the errors to be random.

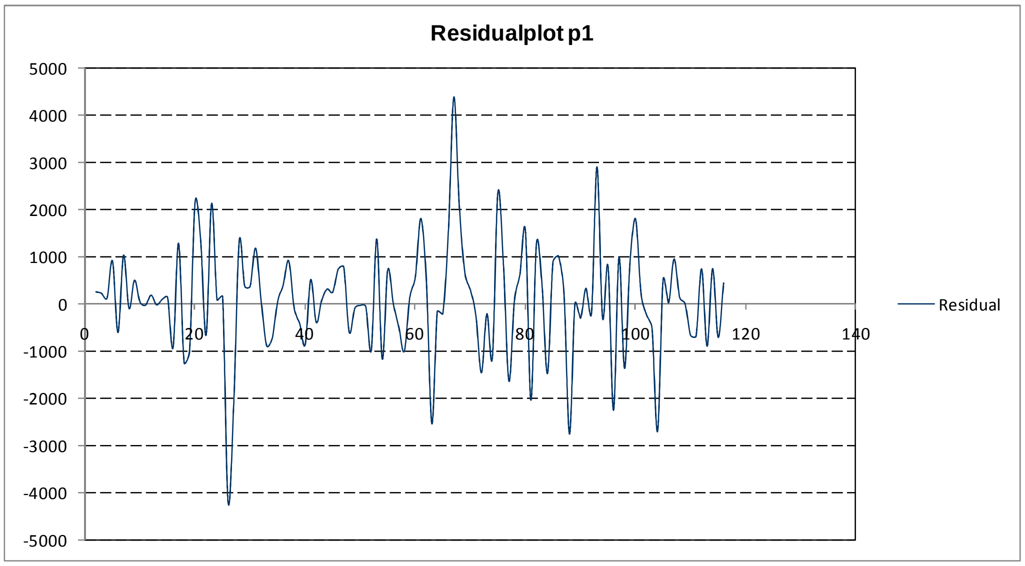


Figure 3.16 Residual plot of p1

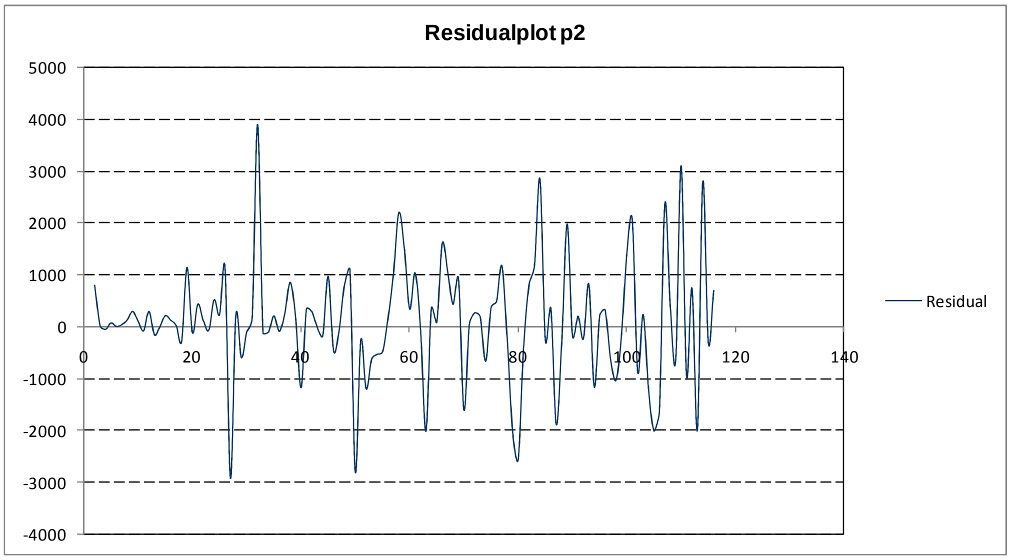


Figure 3.17 Residual plot of p2

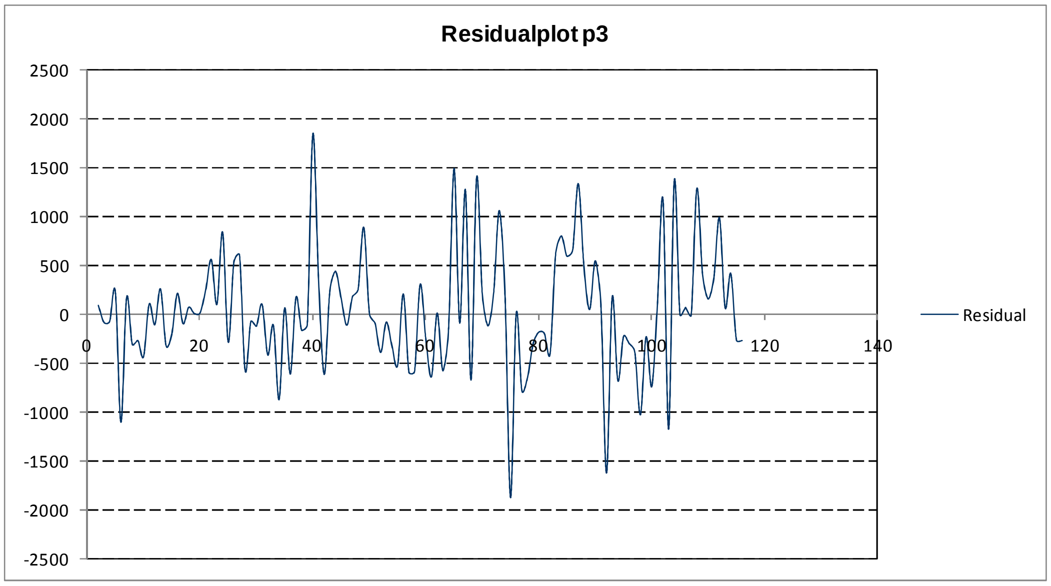


Figure 3.18 Residual plot of p3

We analyzed the characteristics of high technology product development projects and concluded that tasks were executed multiple times due to their iterative nature and the need for convergence. As a result, EAS usage had a cyclic component. Even though the cyclic component may not be apparent from a casual inspection of the usage data, the cycles were extracted using Fourier analysis. From the extracted cycles, we were able to compute the duration of the cycles. The cyclic variations were used to obtain a better fit of the model. We also built an ARIMA model for each of the three projects. In the next section, we develop a capacity forecast for the enterprise.

CHAPTER 4

SOFTWARE LICENSE CAPACITY PLANNING MODEL

In any manufacturing or service enterprise, the type and size of capacity to install have a strong direct influence on the company's bottom line. Capacity adjustments are required to accommodate a change in the total volume or the product mix of demand (Hopp and Spearman, 2001). In a high technology product development enterprise, EAS license capacity has a strong direct influence on engineering productivity. Adjustments to the license capacity are required to support changes in the number of product development projects, the complexity of the projects, and the cyclical variations inherent in high technology projects.

There are several options available for a manufacturing or service facility to adjust capacity. In the short term, this may be done through the use of overtime, addition or deletion of shits, subcontracting, and workforce size changes (Hopp and Spearman, 2001). The options available to a high technology enterprise to adjust EAS license capacity may be procurement of additional short-term peak-demand licenses or allocation of existing licenses among projects based on project priority or some other policy.

Besides the short-term aspects of managing capacity, there are also some strategic considerations. Before an enterprise can decide how much and what type of capacity to install, it must articulate a capacity strategy. For a manufacturing or service enterprise, the strategic considerations may be whether to enter a new market or remain in an existing market, to lead or follow in the product innovation process, or to make or outsource a product. Strategic considerations also include what segment of the market to pursue and several other questions. For a high technology product development enterprise, planning EAS capacity, the strategic considerations may be whether to lease or purchase the software, acquire the software from a sole source or multiple suppliers, acquire adequate sustained capacity with a plan to add peak-demand licenses, negotiate a large capacity at a volume discount to cover worst-case scenario, or develop the EAS in-house. For the purpose of this research, we assume that the above strategic decisions have been made and the issue is how to forecast the EAS capacity to support the strategic direction.

Once the decision has been made to add additional capacity, the next step is to decide how much capacity should be added and when to add it. Should capacity be added to meet the demand that has already emerged or to account for the anticipation of future demand? If we choose not to anticipate demand, should the periods of overcapacity be addressed by procuring peak-demand licenses? If we decide to anticipate demand, how far into the future should we try to cover? Adding capacity in large increments will satisfy demand farther into the future, cause future disruptions due to lack of EAS licenses, and take advantage of volume discounts. However, large increments require a larger upfront financial commitment, poor utilization of capacity, and greater exposure to risk if the anticipated demand does not materialize. The appropriate approach also depends on how the licenses are administered by the enterprise. We discuss the various license administration approaches in section 4.1.

4.1 Enterprise Software License Administration Model

EAS licenses in an enterprise may be shared from a central pool or restricted to a local network, a group, or an individual user. The capacity planning process and the capacity plan are directly influenced by how the enterprise plans to administer the licenses. Another possibility is that the EAS licenses in an enterprise may be completely unrestricted, regardless of how they are administered. This is a special case where no capacity plan is required, as the enterprise has access to an unlimited number of EAS licenses. Most enterprises use a combination of administration models, depending on the scope and breadth of deployment of the EAS. Typically, licenses for an EAS with a large deployment across an enterprise are administered centrally through a central license pool. For the purpose of this research, we will focus on the central license pool (CLP) model and will briefly discuss why other license administration models result in a different capacity plan.

4.1.1 Central License Pool

In a CLP administration model, all EAS license are served from a central license server. All requests for EAS licenses, irrespective of where they originate in the enterprise, are serviced by one license server. This means that there is only one count of all EAS licenses. The key reason to implement a CLP is to maximize utilization. EAS license utilization in a CLP is higher because in practice not all projects reach peak demand levels at the same time. Thus, instead of implementing capacity to meet the aggregate peak demand of all concurrent projects, less capacity will be required.

Another reason to implement a CLP is to benefit from the time zone differences in a global enterprise. The peak requests from users across the globe will now be seen as a rolling peak, rather than a single peak, during a 24-hour period.

Figure 4.1 shows a CLP server model for three locations that collectively host three projects. A project may be at one or several locations, but EAS licenses are shared from a central pool.

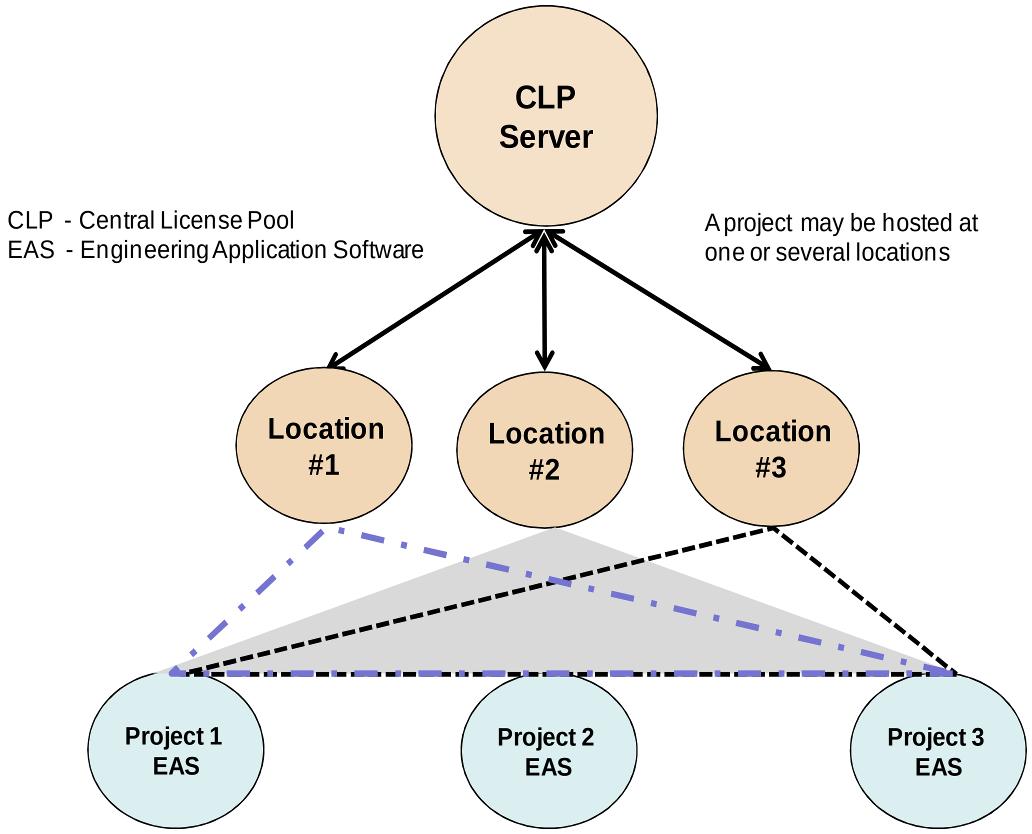

CLP Server
CLP - Central License Pool
EAS - Engineering Application Software
A project may be hosted at one or several locations
Location #1
Location #2
Location #3
Project 1 EAS
Project 2 EAS
Project 3 EAS

Figure 4.1 Central License Pool (CLP) server model

The key disadvantages to a CLP are lack of redundancy and latency. The lack of redundancy becomes an issue when the EAS cannot communicate with the License Server due to a network failure or License Server failure. Latency may become an issue when a license request originates from a location that is geographically far away. The long network latency may cause the license request to time out. Both of these issues can be addressed by dividing the EAS licenses across multiple License Servers located in different geographical locations. In this scheme, the user requests a license from the geographically nearest License Server. If an EAS license is not available, the request rolls over to the next closest geographical location. It is important to note that, although there are multiple License Servers, this is still a conceptual CLP approach for the purpose of capacity planning.

4.1.2 Other License Administration Models

In addition to the CLP approach, there are also decentralized approaches to EAS license approaches. Licenses may be decentralized based on a Local Area Network (LAN), a geographic region, a group, or a project. Licenses may also be locked to a specific computer system or user. The primary reason to decentralize licenses is cost. EAS suppliers charge less for decentralized licenses because they are perceived to deliver less value to the customer. Licenses may be decentralized if enterprise-wide deployment is not required. This may realize savings for the enterprise at the expense of increased license administration complexity. The scope of this research is to create the capacity plan for a CLP model.

4.2 Central License Pool capacity plan

As discussed in section 4.1.1, the CLP server model maximizes utilization. EAS licenses are shared by multiple projects across multiple locations. Due to projects reaching their cyclical peaks at different times and the random nature of license access, the aggregate number of licenses required is less than the sum of the peak demand for each project. In this section, we propose an approach to develop a short-term capacity forecast. A short-term capacity forecast will anticipate demand and allow a plan to be put in place to provide adequate EAS license capacity. In the next section, we propose a methodology to develop a capacity plan for a CLP server.

4.2.1 Capacity plan methodology

In chapter 2 and chapter 3 we investigated the fundamental characteristics of high technology product development and proposed that EAS usage is cyclical due to these characteristics. The characteristics discussed were the iterative and repetitive nature of product development tasks to achieve product design convergence and reduce time to market (Krishnan, 1996). A capacity plan requires forecasting of the aggregate EAS usage of multiple projects and all non-project related usage. Multiple projects may peak in arbitrary cycles based on their development schedules, as the development cycle of one project may be orthogonal to another project. Non-related usage could be due to the work required to (a) support legacy products, (b) investigate errata and bug fixes to support field staff, and (c) research and investigate tasks in support of the definition of future projects.

We cannot make a case from a fundamental perspective about the presence of EAS usage cycles in a CLP server. This is confirmed by running a correlogram of the time series data, as shown in Figure 4.2. Visual inspection shows a clear trend, but no evidence of cyclic usage. Therefore, we propose the use of neural nets to forecast the capacity required for a CLP. Neural nets capture the complex nonlinear relationships that may exist between the project EAS usage and the aggregate capacity required in a CLP (Saito and Kakemoto, 2004).

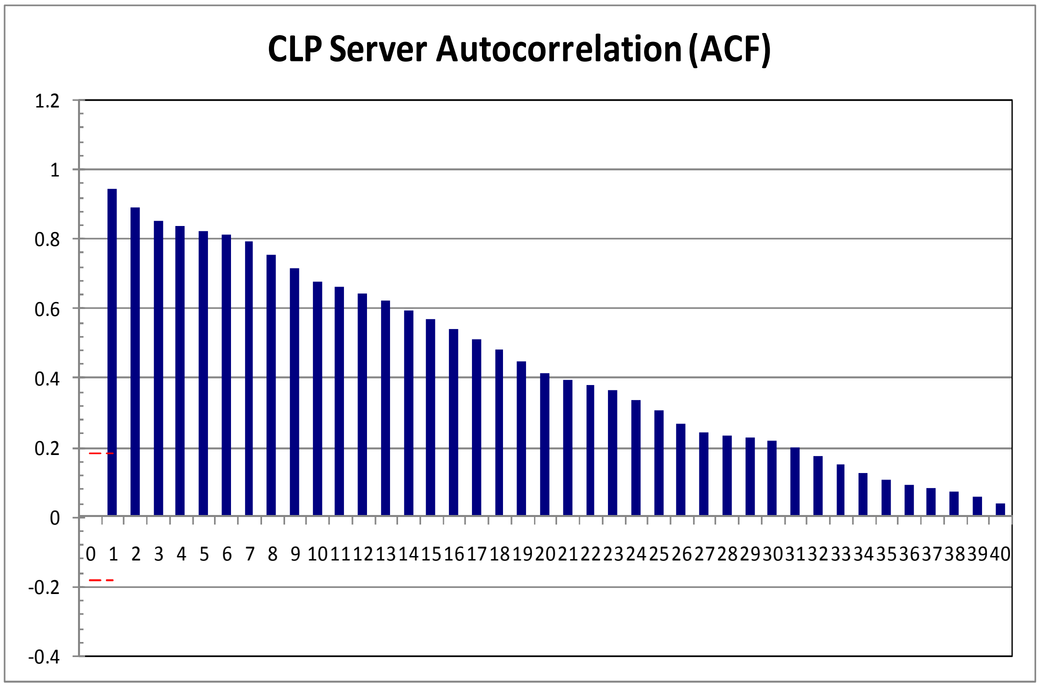


Figure 4.2 CLP server usage correlogram

In Figure 4.2, the correlogram for a CLP server shows that the usage data is non-random and has a linear trend.

An Artificial Neural Network (ANN) is an information-processing paradigm that is based on the way biological nervous systems, such as the brain, process information. Neural networks have an uncanny ability to derive meaning from complicated or imprecise data and can be used to extract patterns or detect trends that are too complex to be noticed by either humans or other computer techniques.

An ANN consists of a set of highly interconnected entities called units. Each unit is designed to mimic its biological counterpart, the neuron. Each accepts a weighted set of inputs and responds with an output.

$X = (x1, x2, \ldots\ldots, xn)$ Equation 4.1

where

X is the input vector

xi are real numbers, represent the set of inputs presented to the unit U

Each unit has an associated weight that represents the strength of that particular connection.

$W = (w1, w2, \ldots., wn)$ Equation 4.2

where

W is the weight vector

wi are real numbers that represent the weight vector corresponding to the to the input vector X.

Applied to U, these weighted inputs produce a net sum at U given by

$S = SUM(wi*xi)$ Equation 4.3

where

S is the net sum

The state of a unit U is represented by a numerical value A, the activation value of U. An activation function f determines the new activation value of a unit from the net sum to the unit and the current activation value. A neural network is composed of such units and weighted unidirectional connections between them. The output of one unit typically becomes an input for another. There may also be units with external inputs and/or outputs. For a simple linear network, the activation function f is a linear function, so that

$f(cS) = cf(S)$ Equation 4.4

$f(S1 + S2) = f(S1) + f(S2)$ Equation 4.5

A single layer network consists of a set of units organized in a layer. Each unit Ui receives a weighted input xj with weight wji. The m x n weight matrix is

$$W=\begin{bmatrix} w_{11} & w_{12} & w_{1n} \\ w_{21} & w_{22} & w_{2n} \\ w_{m1} & w_{m2} & w_{mn} \end{bmatrix} \qquad \text{Equation 4.6}$$

Thus, the output Y_k at unit U_k is

$$Y_k = (w_{1k}, w_{2k}, \ldots., w_{mk}) \begin{bmatrix} x_1 \\ x_2 \\ x_m \end{bmatrix} \qquad \text{Equation 4.7}$$

So the output vector Y = (y1, y2, ….,yn)T is given by

$$Y = WT*X \qquad \text{Equation 4.8}$$

The ANN used to predict the EAS capacity is a multilayer network, an extension of the single-layer network. A multilayer network has two or more layers of units, with the output from one layer serving as input to the next. The layers with no external connections are referred to as hidden layers. See Figure 4.3.

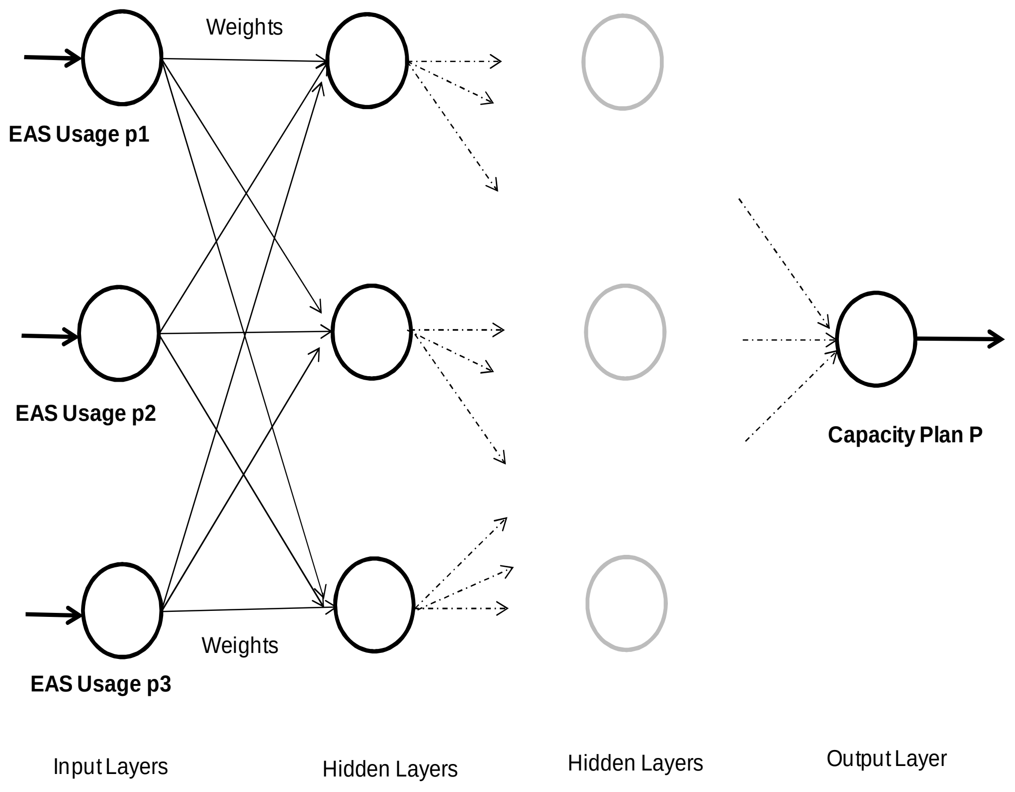


Figure 4.3 Multilayer ANN for CLP capacity plan

The ANN to predict CLP capacity has multiple inputs and a single output. The multiple inputs are the project usage data for projects p1, p2, and p3, and the output is the aggregate capacity P for the training data. During the training, the ANN will adjust its connection weights in order to associate given input vectors with the corresponding output vectors. In the training phase, the input vectors are repeatedly presented, and the weights are adjusted according to the learning rule until the network learns the associations, i.e. until Y = WT*X.

Let X = (x1, ….xm) be the input vectors and Y = (y1,…..yn)T be the output vectors. Then, in each training the weights are adjusted by

dwij = e*xi*yj Equation 4.9

where e is a constant called the learning rate.

Consistent with the methodology deployed in building integrated models to improve the ANN forecast, we develop an ARIMA model. The flowchart in Figure 4.4 shows the methodology in detail.

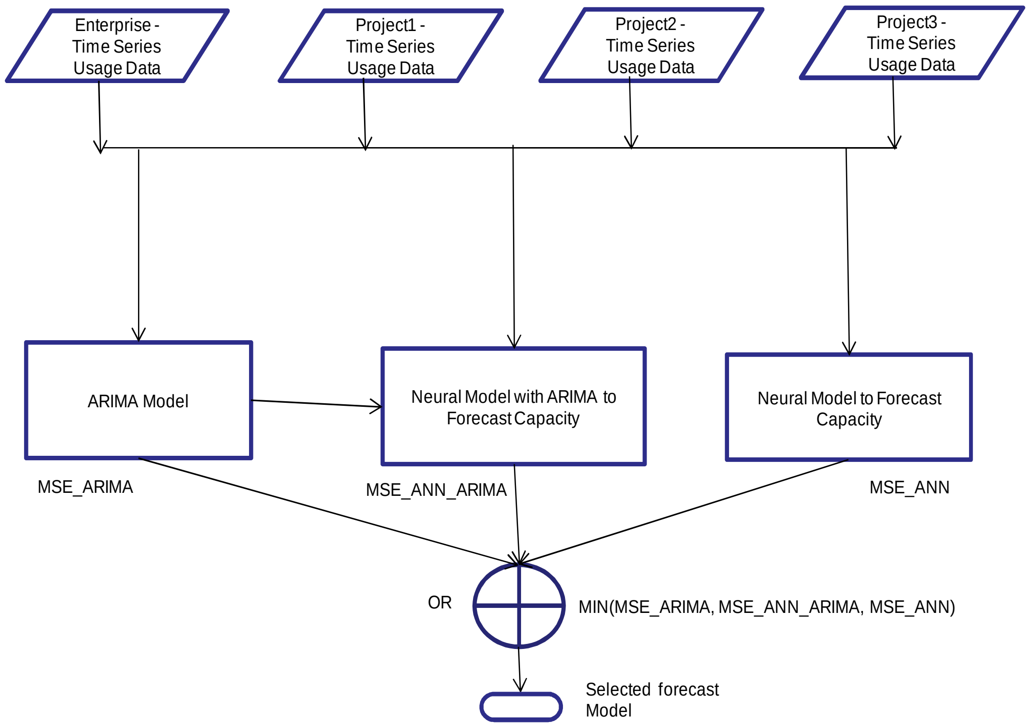

Enterprise - Time Series Usage Data
Project1 - Time Series Usage Data
Project2 - Time Series Usage Data
Project3 - Time Series Usage Data
ARIMA Model
Neural Model with ARIMA to Forecast Capacity
Neural Model to Forecast Capacity
MSE_ARIMA
MSE_ANN_ARIMA
MSE_ANN
OR
MIN(MSE_ARIMA, MSE_ANN_ARIMA, MSE_ANN)
Selected forecast Model

Figure 4.4 Software License Capacity forecast methodology

4.2.2 Capacity plan development

The methodology described in section 4.2.1 is implemented using the Microsoft Excel spreadsheet software. There are two key steps in this process.

a) Train a neural net to predict software capacity by using the model data for all three projects. The neural net used is a Microsoft Excel add-in macro.

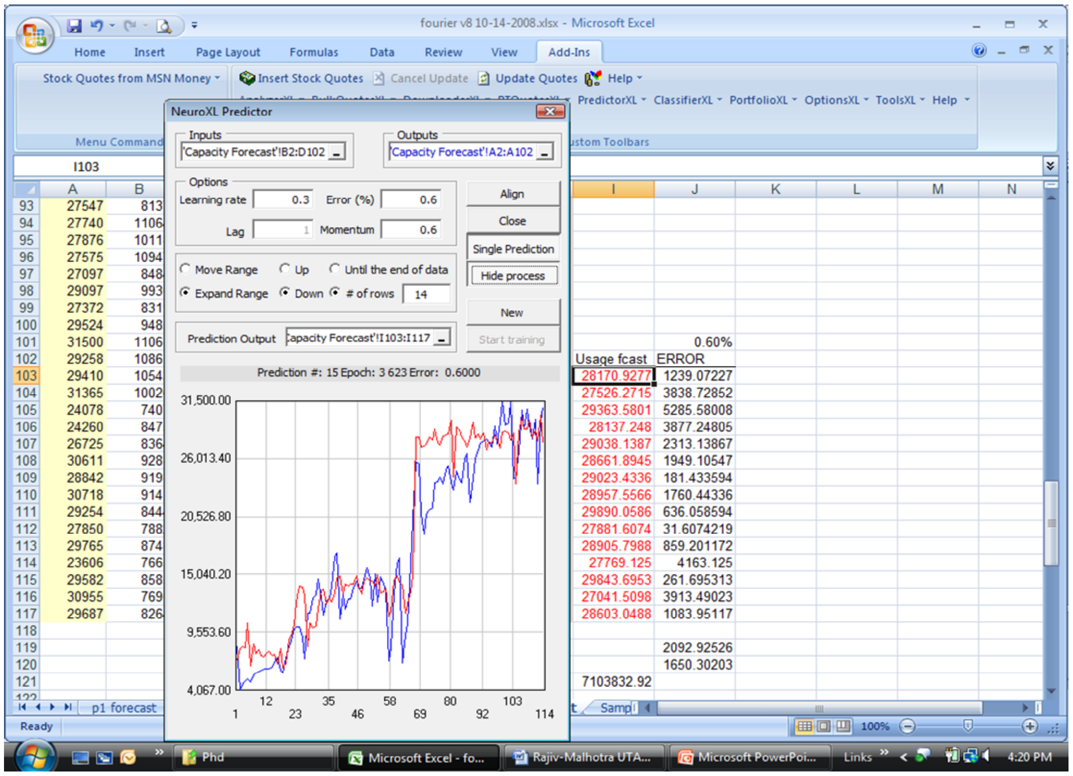
b) Use the trained neural net from the previous step, but instead of using the actual data, use the forecasted data for the three projects to forecast the capacity for the subsequent cycles.

The setup to train the neural nets is shown in Table 4.1. Actual(P) is the license usage from the CLP. Actual(p1), Actual(p2), and Actual(p3) are the license usage for projects p1, p2, and p3, respectively. To train the neural nets, the EAS usage of projects p1, p2, and p3 are the inputs to the neural net and the usage of CLP is the output.

Table 4.1 Neural net training for capacity plan

Actual (P)	Actual(p1)	Actual (p2)	Actual(p3)
2708	2812	1105	3780
8134	3080	1893	3880
4067	3243	1923	3811
4564	3309	1856	3742
4888	4188	1909	3981
5067	3376	1899	2899
4866	4531	1923	3011
5512	4204	2038	2721
5670	4717	2318	2441
5805	4634	2412	1998

From the 116 weeks of recorded data, 101 weeks of data were used to train the neural nets. Once the neural nets were trained, they were used to predict the CLP values for the subsequent 15 weeks. A 15-week forecast was selected to accommodate a planning horizon of one calendar quarter. Figure 4.1 shows the screen shot of the NeuroXL-Predictor software used to train the neural net and predict CLP values.



NeuroXL Predictor

Inputs: 'Capacity Forecast'!B2:D102
Outputs: 'Capacity Forecast'!A2:A102

Options
Learning rate 0.3 Error (%) 0.6
Lag 1 Momentum 0.6

( ) Move Range ( ) Up ( ) Until the end of data
[✓ Expand Range] [✓ Down] [✓ # of rows] 14

Prediction Output 'apacity Forecast'!I103:I117

Prediction #: 15 Epoch: 3 623 Error: 0.6000

I103

	A	B	I	J
93	27547	813		
94	27740	1106		
95	27876	1011		
96	27575	1094		
97	27097	848		
98	29097	993		
99	27372	831		
100	29524	948		
101	31500	1106		0.60%
102	29258	1086	Usage fcast	ERROR
103	29410	1054	28170.9277	1239.07227
104	31365	1002	27526.2715	3838.72852
105	24078	740	29363.5801	5285.58008
106	24260	847	28137.248	3877.24805
107	26725	836	29038.1387	2313.13867
108	30611	928	28661.8945	1949.10547
109	28842	919	29023.4336	181.433594
110	30718	914	28957.5566	1760.44336
111	29254	844	29890.0586	636.058594
112	27850	788	27881.6074	31.6074219
113	29765	874	28905.7988	859.201172
114	23606	766	27769.125	4163.125
115	29582	858	29843.6953	261.695313
116	30955	769	27041.5098	3913.49023
117	29687	826	28603.0488	1083.95117
118				
119				2092.92526
120				1650.30203
121			7103832.92	

Figure 4.5 Screen shot of NeuroXL-Predictor software

In Figure 4.5, the blue line shows Actual(P), the actual CLP usage, and the red line shows the usage predicted by the neural net software. NeuroXL-Predictor provides a powerful mechanism for making multiple successive predictions, known as Bulk Predictions. With this mechanism, each successive prediction is based on a range of data, either shifted or expanded by a single cell from the previous range. The following options were used to control the training and bulk predictions.

Learning rate: a value between 0 and 1 that affects the rate at which the network learns. The larger the learning rate, the faster the network will converge. Oscillation and non-convergence can occur if the learning rate is set too high.

Error %: specifies the point at which the neural network will stop training. Reducing this value can improve accuracy, but the prediction will take longer.

Lag: specifies the number of predicted values to return (e.g. 1 will return next week's predicted closing price, 2 will return the next two weeks, etc.).

Momentum: High learning rates often lead to weight change oscillations in the training process, which can cause non-convergence or return a non-optimal solution. Momentum makes it less likely for such undesirable cases to occur by making the next weight change a function of the previous weight change to provide a smoothing effect. The value for momentum (between 0 and 1) determines the proportion of the last weight change that is added to the next weight change.

Expanded Range: increases the range by one cell for each step. With **"Down"** selected, the range "A1:A10" will become "A1:A11" in the next step. Selecting **"U*p*"** will expand the range in the opposite direction.

Number of rows: specify here the exact number of moves/expansions desired. The number of resulting predictions will be 1 more than this number.

Table 4.2 shows the Excel setup to compute the predicted values. The predicted values are shown under the column "Usage fcast," and the column "Abs Error" shows the absolute values of the error. Also shown are the average values of the error and the standard deviation.

Table 4.2 Neural network predicted values and error

Actual (P)	Actual(p1)	Actual (p2)	Actual(p3)		Usage fcast	Abs Error
29410	10543	8088	4814		28171	1239
31365	10026	8213	3718		27526	3839
24078	7407	6933	5028		29364	5286
24260	8475	4815	5117		28137	3877
26725	8364	3084	5201		29038	2313
30611	9288	5359	5211		28662	1949
28842	9198	5807	6512		29023	181
30718	9141	5129	7014		28958	1760
29254	8444	8156	7213		29890	636
27850	7882	7258	7610		27882	32
29765	8748	7894	8640		28906	859
23606	7665	5867	8789		27769	4163
29582	8585	8529	9233		29844	262
30955	7690	8257	9017		27042	3913
29687	8264	8865	8756		28603	1084
				Avg Err		2093
				Std dev		1650
				MSE		7,103,833

Table 4.3 shows the setup to compute the ARIMA model for the capacity forecast. In this approach, we use ARIMA on the capacity usage data using a built-in Excel macro. Table 4.3 shows the setup in Excel build an ARIMA model of autoregressive order (p) of 1 and moving average order (q) of 1. We obtain the graph shown in Figure 4.6.

The critical values of t for degrees of freedom equal to 100 and α equal to 0.05 is 1.9840. Since the t-statistic at $112.2213 > t_{0.05,100} = 1.984$, we reject the null hypothesis that there is no correlation between autoregressive variable AR(1) and the fitted model. Similarly, we reject the null hypothesis that there is no correlation between the moving average variable MA(1) and the fitted model. The R-squared value of 0.91554 shows that 91% of the variation in the model can be explained by the variables AR(1) and MA(1).

Table 4.3 ARIMA estimation of Capacity time series

Capacity ARIMA

timeseries: y
Method: Nonlinear Least Squares (Levenberg-Marquardt)
date: 02-21-09 time: 12:54
Included observations: 102
p = 1 - q = 1 - no constant - manual selection

	Coefficient	Std. Error	t-Statistic	R-squared
AR(1)	1.006131777	0.008965601	112.2213467	0.915545551
MA(1)	-0.275274525	0.090874034	3.02918792	

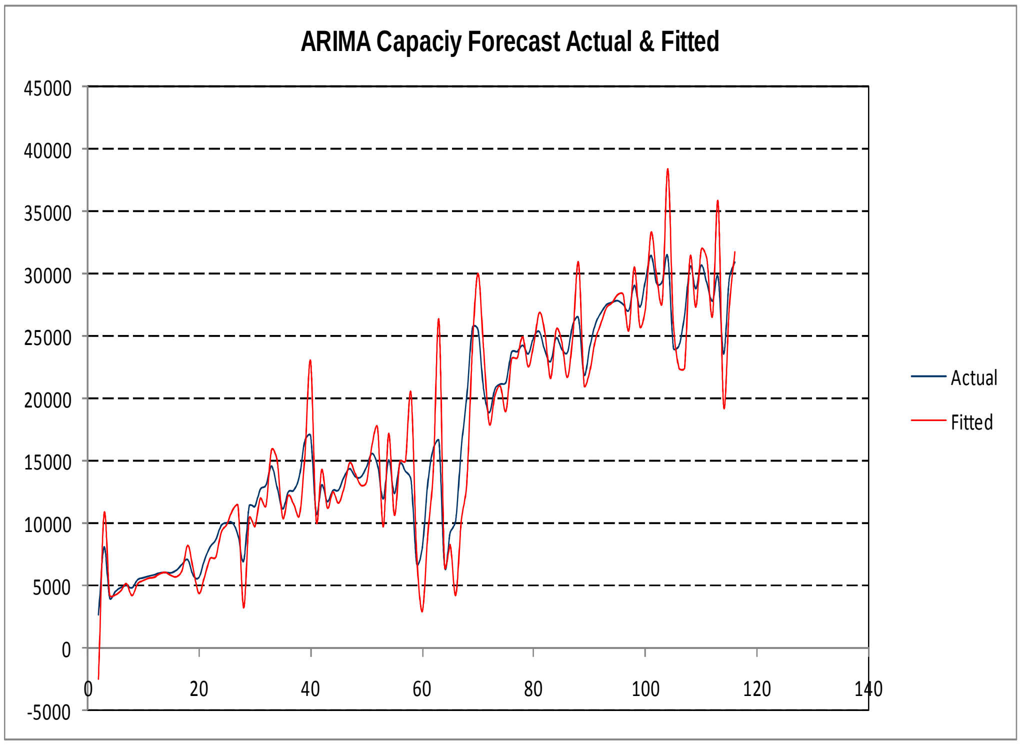


Figure 4.6 ARIMA capacity forecast model

Table 4.4 ARIMA predicted and values and error

Actual (P)		ARIMA fcast	Abs Error
29410		30092.89	683
31365		30277.42	1088
24078		30463.07	6385
24260		30649.87	6390
26725		30837.80	4113
30611		31026.89	416
28842		31217.14	2375
30718		31408.56	691
29254		31601.15	2347
27850		31794.92	3945
29765		31989.88	2225
23606		32186.04	8580
29582		32383.39	2801
30955		32581.96	1627
29687		32781.75	3095
	Avg Abs Err		3117
	Std Dev		2396.43
	MSE		16,869,054

Table 4.5 Neural network predicted values and error with ARIMA input

ARIMA	Actual (P)	Actual(p1)	Actual (p2)	Actual(p3)		Usage fcast	Abs Error
27780	29410	10543	8088	4814		29235	175
38395	31365	10026	8213	3718		30596	769
25979	24078	7407	6933	5028		28022	3944
22467	24260	8475	4815	5117		26424	2164
22509	26725	8364	3084	5201		27507	782
31407	30611	9288	5359	5211		28431	2180
27362	28842	9198	5807	6512		28849	7
31962	30718	9141	5129	7014		29767	951
31180	29254	8444	8156	7213		29851	597
26635	27850	7882	7258	7610		29510	1660
35772	29765	8748	7894	8640		29571	194
19428	23606	7665	5867	8789		29002	5396
27240	29582	8585	8529	9233		29013	569
31768	30955	7690	8257	9017		30340	615
30092	29687	8264	8865	8756		29175	512
					Avg Abs Err		1368
					Std Dev		1519.07
					MSE		4,024,005

Table 4.6 MSE of capacity plan model

	MSE capacity plan		
	MSE (ARIMA)	***MSE (ANN)***	***MSE (ANN_ARIMA)***
p1	16,869,054	7,103,833	4,024,005

Table 4.6 summarizes the three prediction models. We get the least MSE with the ANN model integrated with ARIMA. In chapter 5, we analyze and validate the results.

CHAPTER 5

ANALYSIS AND VALIDATION OF THE MODELS

In this chapter, we validate the EAS project forecast and the CLP EAS capacity plan. We collected the weekly peak EAS usage data for a high technology product development project, p1, for the complete development cycle. The data for p1 was collected over a period of 116 weeks. EAS usage data for the other key product development projects, p2 and p3, was also collected during the 116-week period. In chapter 3, we developed a model to forecast the project usage for an EAS; in chapter 4, a model to forecast the EAS capacity requirements when multiple projects share a central license pool.

EAS usage data was collected for 116 weeks. 101 weeks of data were used to build the project forecast and capacity models. The remaining 15 weeks of data were predicted using the developed models. For the purpose of validation, the predicted data was compared to the actual usage. In section 5.1, we validate and analyze the results for the EAS forecasting model; in section 5.2, we do the same for the capacity plan.

5.1 Validation of EAS Project Forecast Model

A forecasting error, or residual, is a difference between the actual and the fitted observation that was produced using a forecasting method:

$$e_t = y_t - F_t \qquad \text{Equation 5.1}$$

where

e_t= residual at the period ‘t’

y_t= actual observation in the original time series in the period ‘t’

F_t= forecast for the period ‘t’

n = number of errors

$\bar{e}$ = the mean value of all errors in the series

$\bar{y}$ = the mean value of all original observations in the series

The first and most common statistic that can be calculated is called the mean error (ME):

$$ME = \frac{\sum_{t=1}^{n} e_t}{n} \qquad \text{Equation 5.2}$$

The formula for ME implies that, by definition, all the positive deviations are eliminated by all the negative ones. This means that if there are dramatic fluctuations in both directions, the end result could still be zero, despite large fluctuations. Therefore, we introduce the next statistic called mean absolute deviation (MAD):

$$MAD = \frac{\sum_{t=1}^{n} |e_t|}{n} \qquad \text{Equation 5.3}$$

MAD measures deviations from the series in absolute terms, which means that, regardless of whether the errors are positive or negative, we observe them as positive.

The next error statistic is the mean square error (MSE):

$$MSE = \frac{\sum_{t=1}^{n} {e_t}^2}{n} \qquad \text{Equation 5.4}$$

MSE is dimensionless and is used when comparing with MSE values calculated for other forecasts. It has a tendency to prefer a series of small errors and penalizes forecasts if there are one or two large errors in the series. It encourages safe forecasting that produces smoother forecasting lines, rather than a bolder approach that tries to follow the dynamic pattern of the series, even at the expense of making one or two bigger mistakes.

To calculate the percentage deviation of a forecast from the original series, the mean absolute percentage error (MAPE) statistic is used. MAPE does not take into account positive or negative variations, but rather a typical percentage deviation expressed as a percentage.

$$MAPE = \frac{\sum_{t=1}^{n} \left|\frac{e_t}{y_t}\right|}{n} 100 \qquad \text{Equation 5.5}$$

The standard error (SE) statistic is used to measure the dispersion of the error from the mean.

$$SE = \sqrt{\frac{\sum_{t=1}^{n}(e_t - \bar{e})^2}{n}} \qquad \text{Equation 5.6}$$

The error variance s, the standard error of the mean estimate, is

$$s^2 = (SE)^2 \qquad \text{Equation 5.7}$$

As SE measures the dispersion of errors around their mean values

$$s^2 = MSE - (\bar{e})^2 \qquad \text{Equation 5.8}$$

$$MSE = s^2 + (\bar{e})^2 \qquad \text{Equation 5.8}$$

Thus, MSE is the sum of two factors. One measures the variability of errors around their mean value, and the other measures how much the errors are biased by calculating the mean. Although we do not want the errors to be too far from the original values, we do not want them to fluctuate too much either. In this respect, MSE as a statistic provides a good measure about the quality of our forecasts.

Another measure of the degree of a successful forecast is the correlation coefficient (r). Because we are not dealing with forecasts calculated on the basis of the least square method, we use the following formula:

$$r = \frac{\sqrt{\sum_{t=1}^{n}(X_t - \bar{Y})(F_t - \bar{Y})}}{\sqrt{\sum_{t=1}^{n} Y_t^2} \sum_{t=1}^{n} F_t^2} \qquad \text{Equation 5.6}$$

The coefficient of correlation takes values between -1 and +1. The closer the value is to +1, the closer the correlation between the actual series and the forecasts. If the value is closer to -1, the two series are completely opposite to each other. If the value is close to zero, there is no correlation.

The coefficient of determination, R, is given by:

$$R = r^2 \qquad \text{Equation 5.7}$$

The coefficient of determination gives us a proportion of the described variance. If we multiply it by 100, it gives us the percentage of deviations of the forecasts from the actual series. These statistics are calculated for the forecasts for projects p1, p2, and p3 and are shown in Tables 5.1, 5.2, and 5.3.

Figures 5.1 to 5.4 show the forecasted values shown with the actual usage values from week #102 to week #116 for project p1.

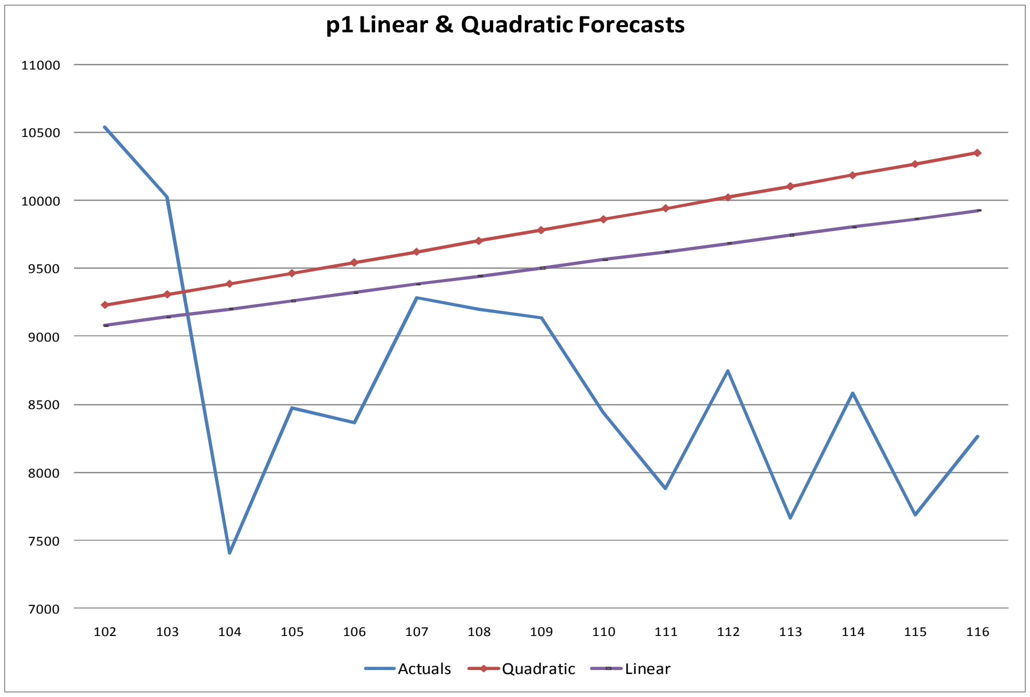


Figure 5.1 Linear and Quadratic values against the actual usage

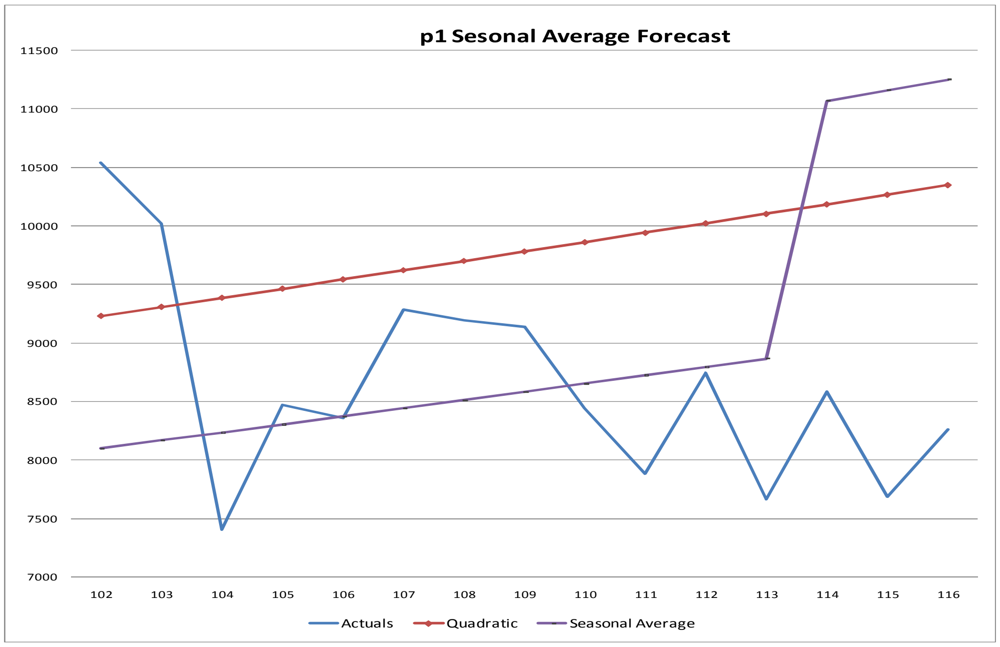


Figure 5.2 Seasonal average forecast against the actual usage

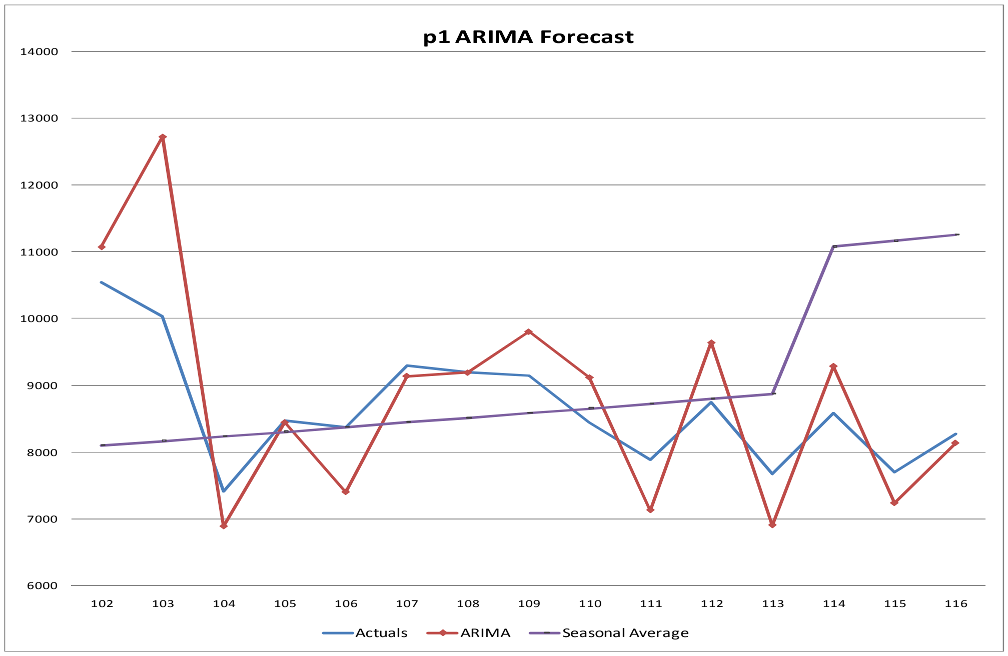


Figure 5.3 ARIMA forecast against the actual usage

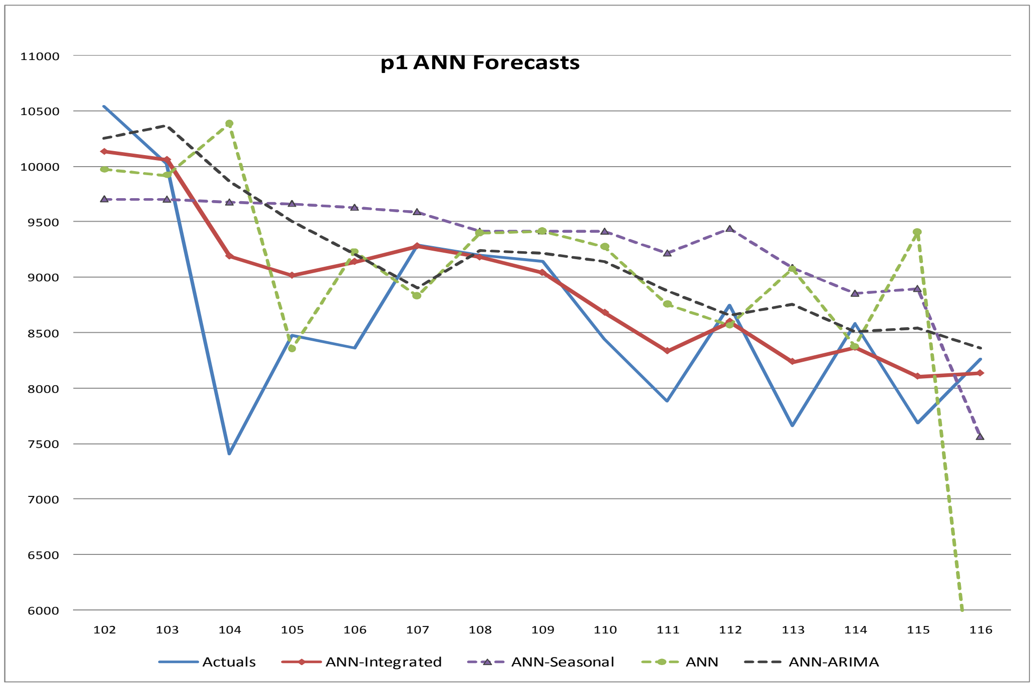


Figure 5.4 ANN forecasts against the actual usage

Table 5.1 summarizes the error statistics for the predicted values from week #102 to week # 116 for project p1. As discussed in the previous section, the EAS usage forecast model was constructed using the data from week #1 to week #101. From Table 5.1, we see that the ARIMA, ANN-ARIMA, and ANN-Integrated models have the lowest values for ME, MAD, and MSE — (-254), 390, and 340,806. As discussed earlier, the MSE is a good statistic because although we do not want the errors to be too far from the original values, we do not want them to fluctuate too much either. The ANN-integrated model has the lowest MSE value. It also shows an r value of 0.95. The closer the value is to +1, the closer the relation between the two series. Also, the R value, the coefficient of determination, for the ANN-Integrated model is 91%. This means that 91% of the variations in the forecast are caused by variations in the actual series and are inherent in our forecasting model. Only 9% of all variations in the forecasts are not attributed to the actual time series and come from an unexplained source. In the next section, we will discuss if this forecast is meaningful for predicting the EAS usage for high technology product development projects.

Table 5.1 Comparison of predicted vs. actual EAS usage data for project p1

	Project p1					
	Seasonal	ARIMA	ANN	ANN-Seasonal	ANN-Arima	ANN-Integrated
ME	-367.83	-159.53	-257.50	-638.20	-513.63	-254
MAD	1242.75	660.29	964.29	885.64	625.09	390
MAPE	14.48	7.52	11.94	10.79	7.76	5
MSE	2731384	819347	2013152	1096558	774021	340806
Std Err	846.53	379.95	893.53	853.78	756.20	562.04
r	0.3673951	0.9087	0.19042771	0.546559022	0.556534184	0.956569
R	0.13497916	0.8257356	0.03626271	0.298726765	0.309730298	0.915025

Figures 5.5 to 5.8 show the forecasted values shown with the actual usage values from week #102 to week #116 for project p2.

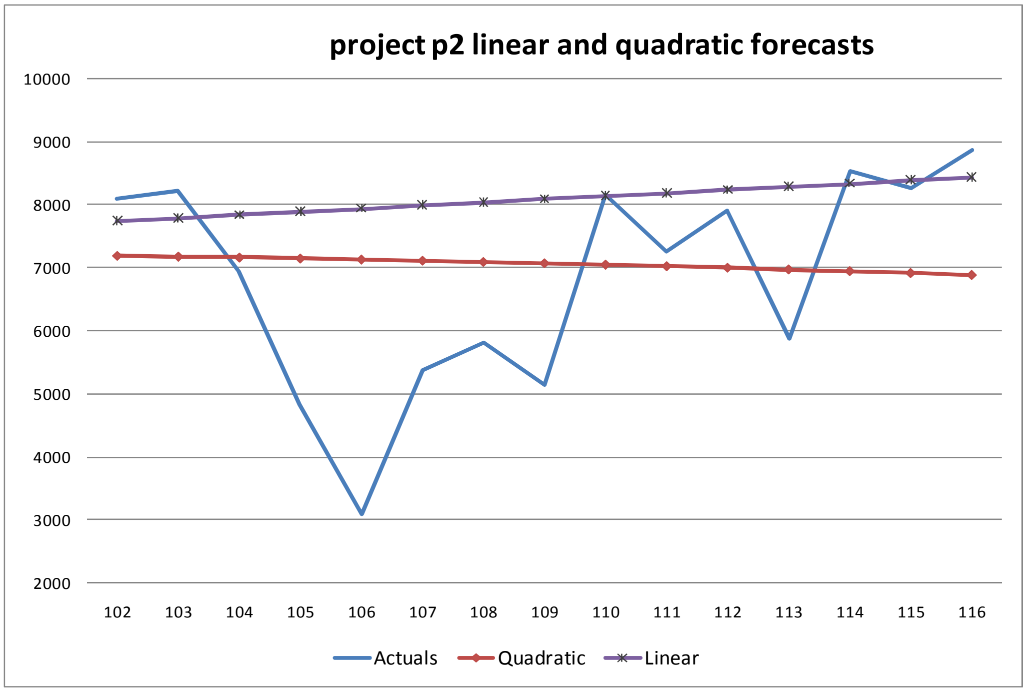


Figure 5.5 Linear and quadratic forecasts against the actual usage

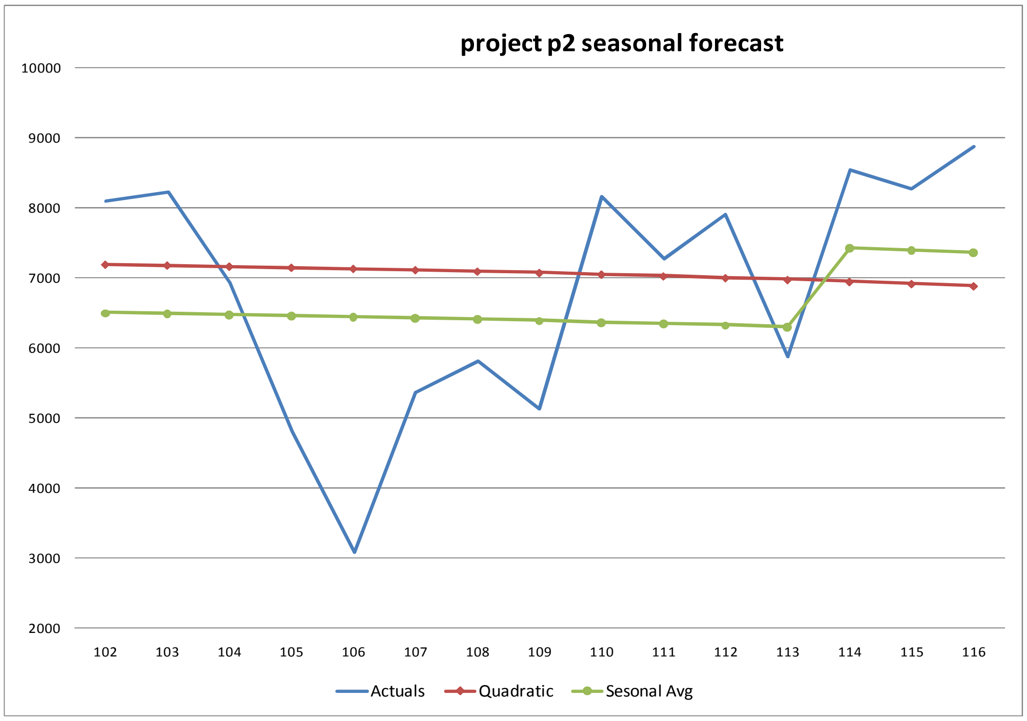


Figure 5.6 Seasonal forecast against the actual usage

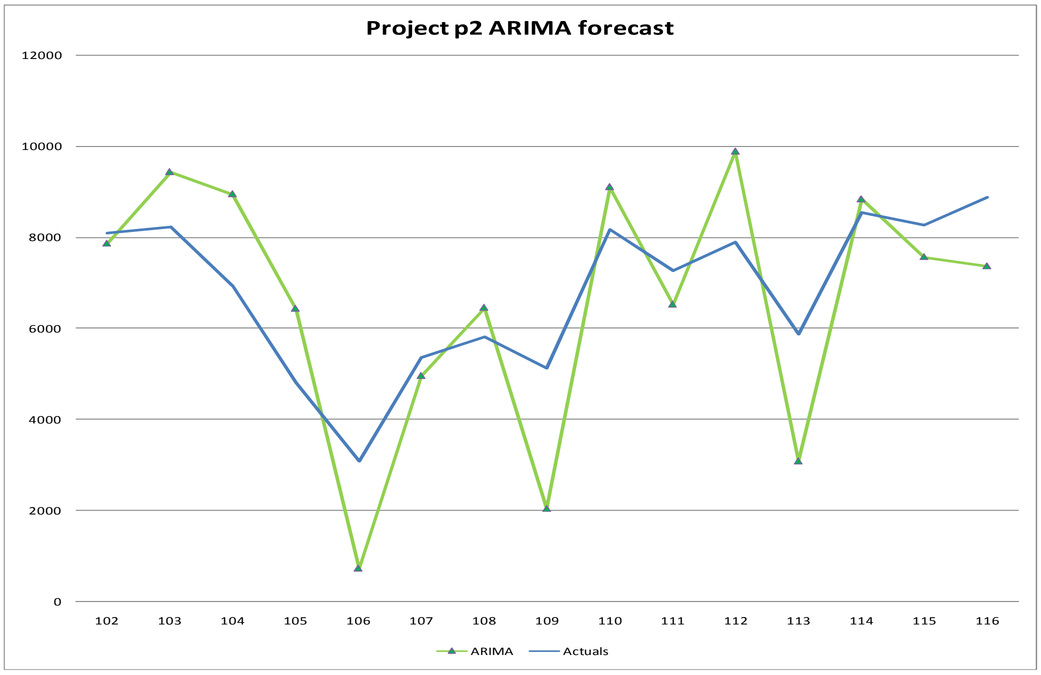


Figure 5.7 ARIMA forecast against actual usage data

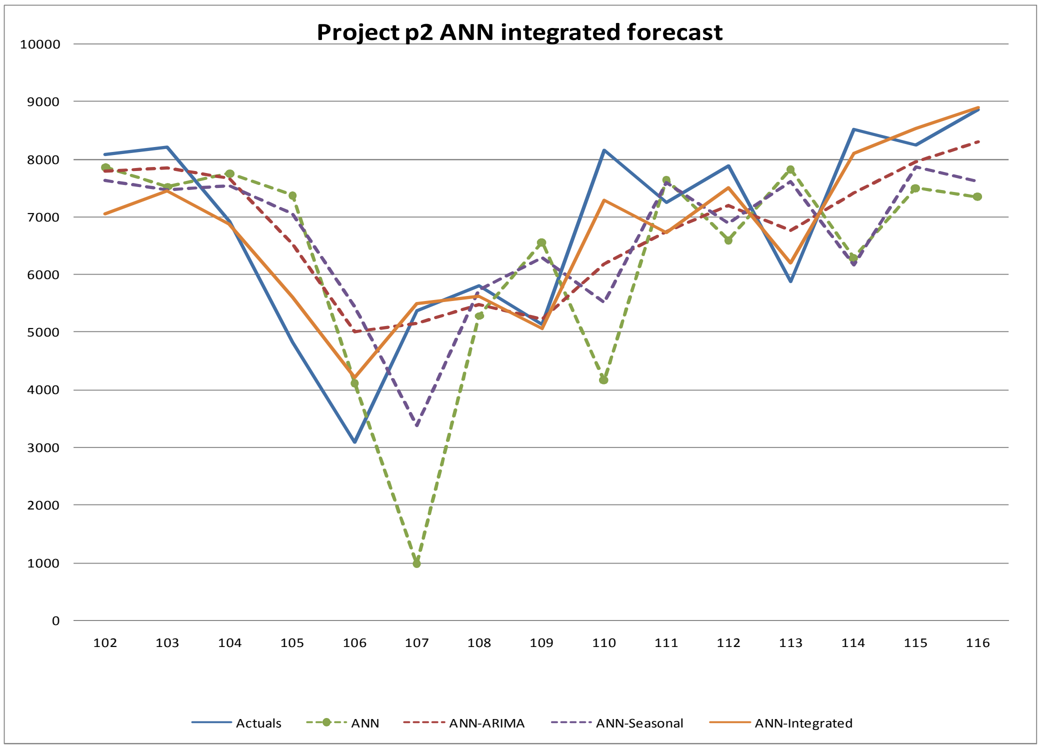


Figure 5.8 ANN forecasts against the actual usage

Table 5.2 summarizes the error statistics for the predicted values from week #102 to week # 116 for project p2. From Table 5.2, we see that the ANN-ARIMA and ANN-Integrated models have the lowest values for ME, MAD, and MSE. The ANN-integrated has the lowest MSE value. It also shows an r value of 0.95 so the original and predicted series are closely related. Also the R value, the coefficient of determination, for ANN-Integrated is 90%. This means that 90% of the variations in the forecast are caused by variations in the actual series and are inherent in our forecasting model. Only 10% of all variations in the forecasts are not attributed to the actual time series and come from an unexplained source.

Table 5.2 Comparison of predicted vs. actual EAS usage data for project p2

Project p2

	Seasonal	ARIMA	ANN	ANN-Seasonal	ANN-Arima	ANN-Integrated
ME	214.14	209.99	497.18	166.13	70.09	156.27
MAD	1326.34	1368.58	1586.41	1292.69	786.38	437.16
MAPE	23.15	23.94	25.64	22.26	13.84	6.91
MSE	2,248,934	2,660,968	3,985,054	2,366,308	982,839	348,414
Std Err	1519.63	989.39	1620.44	1572.99	1022.27	540.93
r	0.51338788	0.829354147	0.403263	0.459281248	0.816541049	0.95220084
R	0.26356712	0.687828301	0.162621	0.210939265	0.666739285	0.90668645

Figures 5.9 to 5.12 show the forecasted values shown with the actual usage values from week #102 to week #116 for project p3.

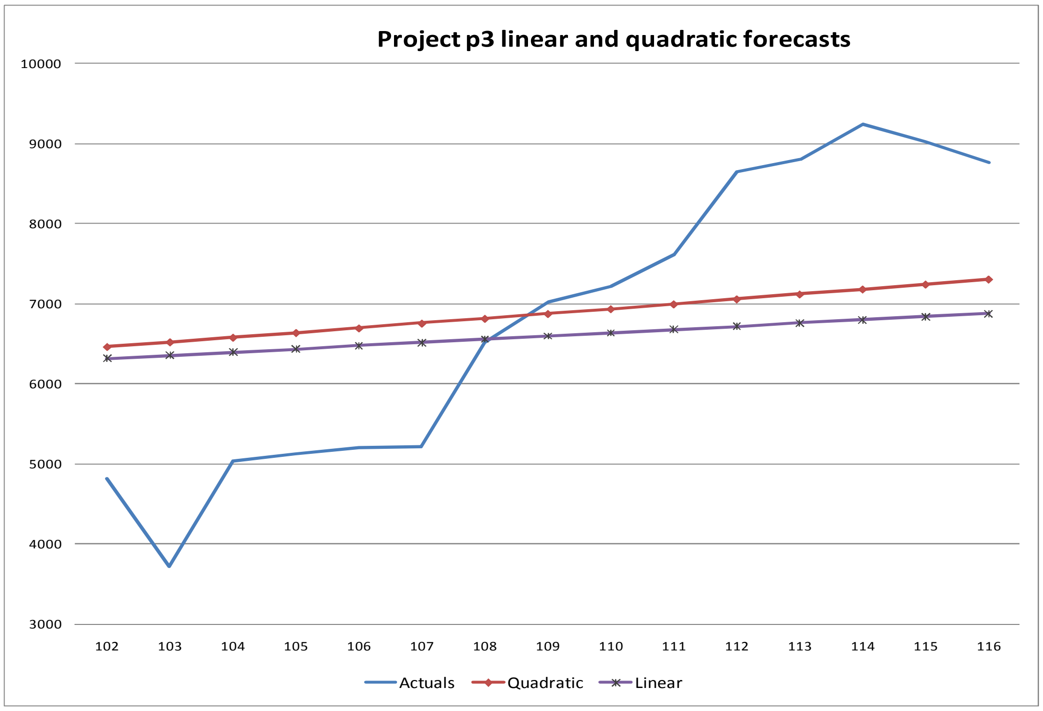


Figure 5.9 Linear and quadratic forecasts against actual usage

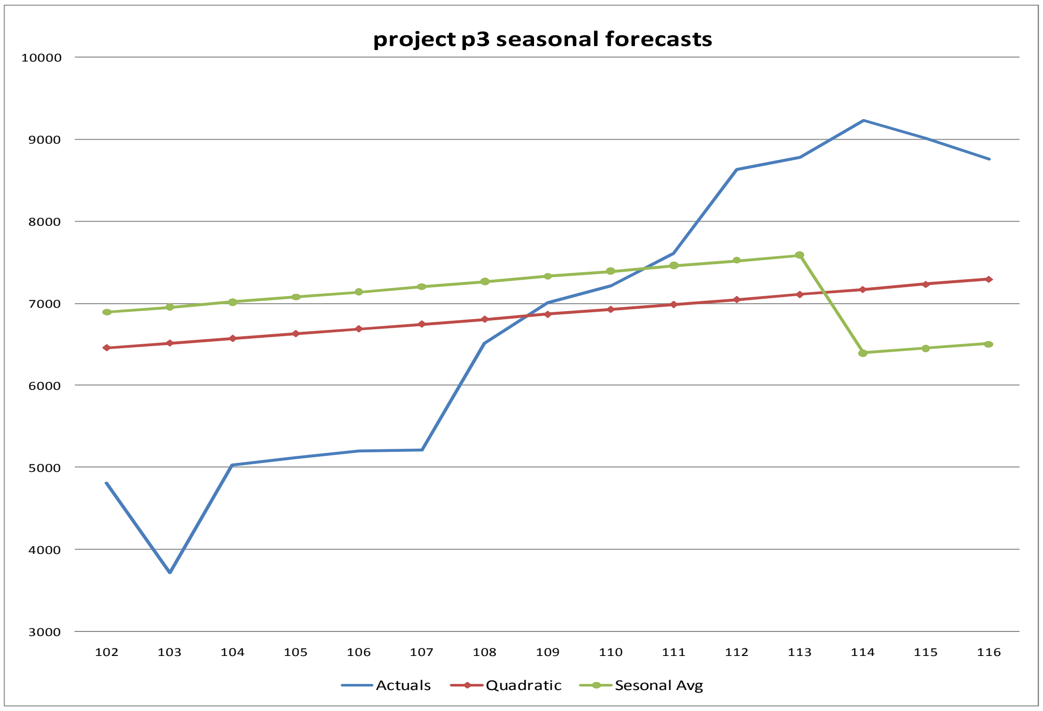


Figure 5.10 Seasonal forecasts against actual usage

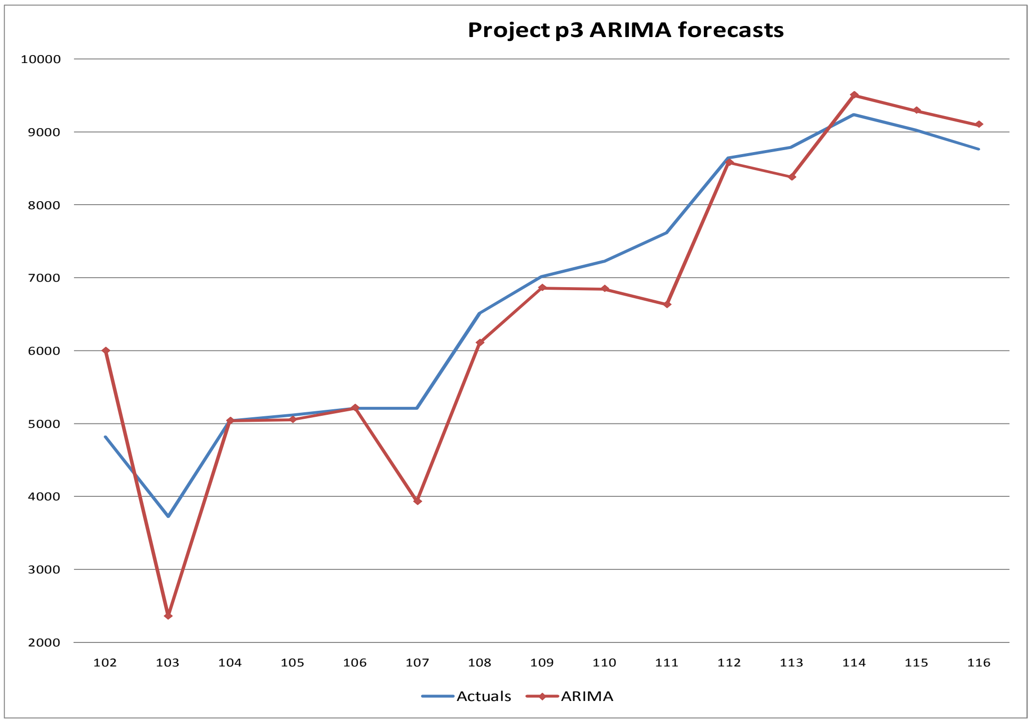


Figure 5.11 ARIMA forecasts against actual usage

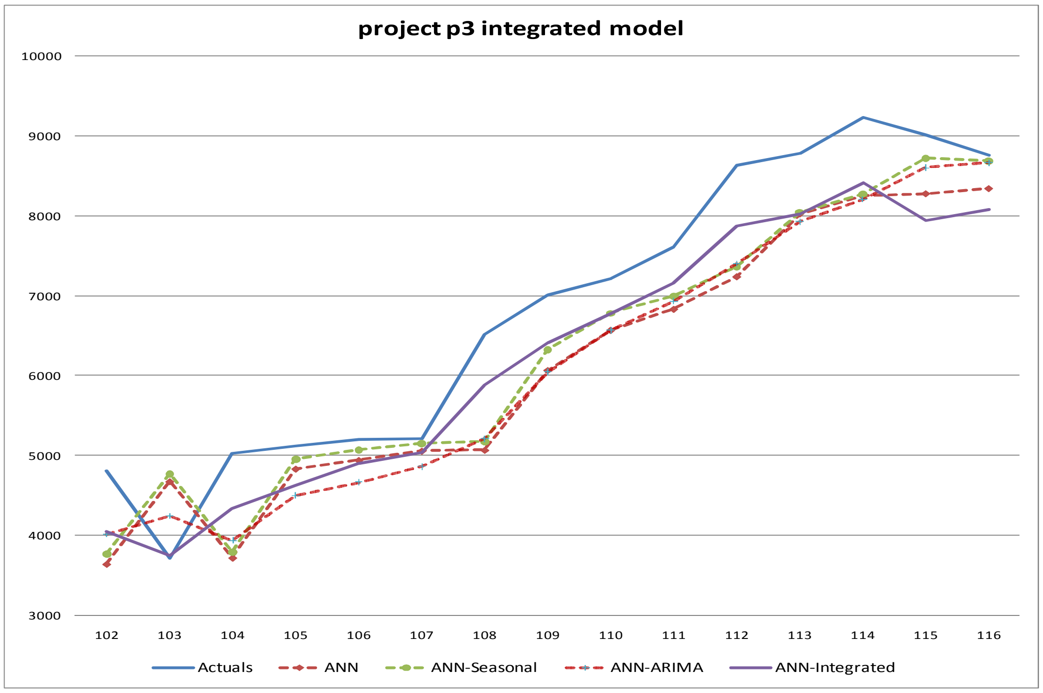


Figure 5.12 ANN Integrated forecasts against actual usage

Table 5.3 summarizes the error statistics for the predicted values from week #102 to week # 116 for project p3. From Table 5.3, we see that the ARIMA model and all of the ANN models have low MSE values. This is due to the highly regular nature of the usage patterns. However, the ANN-integrated model still has the lowest MSE value. It also shows an r value of 0.99, so the original and predicted series are very closely related. Also, the R value, the coefficient of determination, for the ANN-Integrated model is 98%. This means that 98% of the variations in the forecast are caused by variations in the actual series and are inherent in our forecasting model. Only 2% of all variations in the forecasts are not attributed to the actual time series and come from an unexplained source.

Table 5.3 Comparison of predicted vs. actual EAS usage data for project p3

Project p3

	Seasonal	**ARIMA**	**ANN**	**ANN-Seasonal**	**ANN-Arima**	**ANN-Integrated**
ME	-288.35	205.10	689.65	532.34	670.90	573.96
MAD	1638.28	479.75	817.23	672.45	741.12	578.05
MAPE	27.72	8.63	12.87	10.88	11.57	8.37
MSE	1,829,607	444,098	828,026	651,949	658,334	402,035
Std Err	1892.88	579.97	633.64	651.96	488.82	233.04
r	0.63577791	0.98036967	0.94340201	0.939977222	0.96671683	0.99253346
R	0.40421356	0.96112468	0.89000735	0.883557178	0.934541429	0.98512267

5.2 Validation of Capacity Planning Model

Figure 5.13 shows the forecasted values against with the actual usage values from week #102 to week #116 for the CLP server.

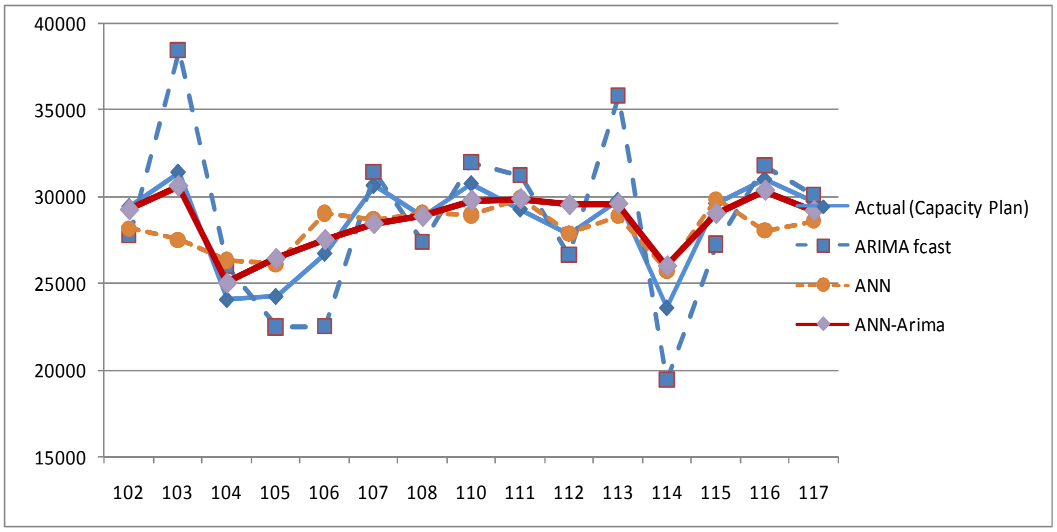


Figure 5.13 ANN and ARIMA forecasts against actual usage

Table 5.4 shows the absolute errors for the predicted values from week #102 to week # 116. As discussed in the previous section, the EAS usage forecast model for CLP was constructed using the data from week #1 to week #101.

Table 5.4 Comparison of predicted vs. actual EAS usage data for CLP

	Project ARIMA	p3 ANN	ANN-Arima
ME	-217.87	259.61	-172.29
MAD	2465.07	1559.59	967.58
MAPE	8.77	5.56	3.57
MSE	9,739,203	3,523,703	1,487,826
Std Err	1474.70	1909.75	1044.45
r	0.83681131	0.70520451	0.92176078
R	0.70025317	0.4973134	0.849642936

Table 5.4 summarizes the error statistics for the predicted values from week #102 to week # 116 for EAS usage on the CLP server. From Table 5.4, we see that both of the ANN models have the lower MSE values. However, the ANN-integrated model still has the lowest MSE value. It also shows an r value of 0.92, so the original and predicted series are very closely related. Also, the R value, the coefficient of determination, for the ANN-Integrated model is 84%. This means that 84% of the variations in the forecast are caused by variations in the actual series and are inherent in our forecasting model. Only 16% of all variations in the forecasts are not attributed to the actual time series and come from an unexplained source.

5.3 Analysis of results

As a first step, we want to have a measure of how much we can rely on the predictions based on the values forecasted by the project usage and capacity usage models. To compute our level of confidence, we use the t-statistic in conjunction with the SE_e, the standard error around the estimates. The confidence interval (CI) is given by

$$\hat{Y} \pm t\sqrt{MSE} \qquad \text{Equation 5.8}$$

where

$\hat{Y}$ is the predicted value

t is the t-statistic

As

$$SE_e = \sqrt{MSE}$$ Equation 5.9

we calculate SE_e directly from

$$SE_e(\hat{Y}) = \sqrt{1 + \frac{1}{n} + \frac{(X - \overline{X})^2}{\sum(X_i - \overline{X})^2}}$$ Equation 5.10

In Figures 5.14 to 5.19, we show the 95% confidence interval for all of the forecasts for project p1. In Figures 5.20 and 5.21, we show the 95% confidence intervals for the ANN-Integrated forecasts for projects p2 and p3.

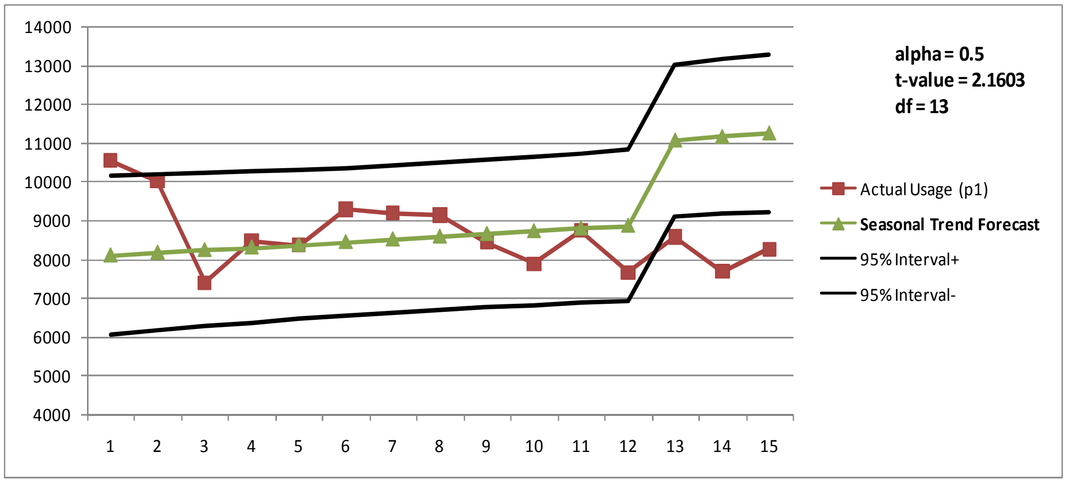


Figure 5.14 95% CI for Seasonal Trend forecast for p1

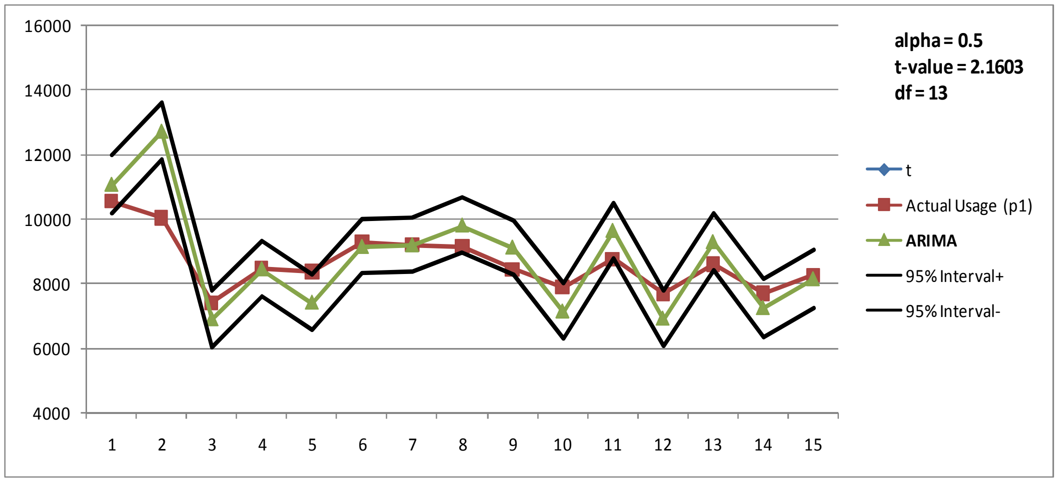


Figure 5.15 95% CI for ARIMA forecast for p1

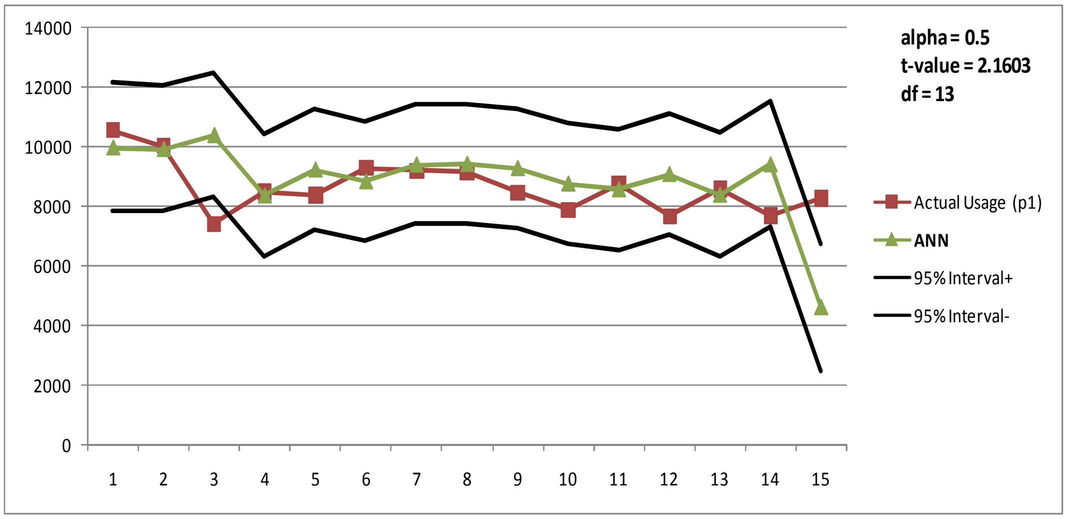


Figure 5.16 95% CI for ANN forecast for p1

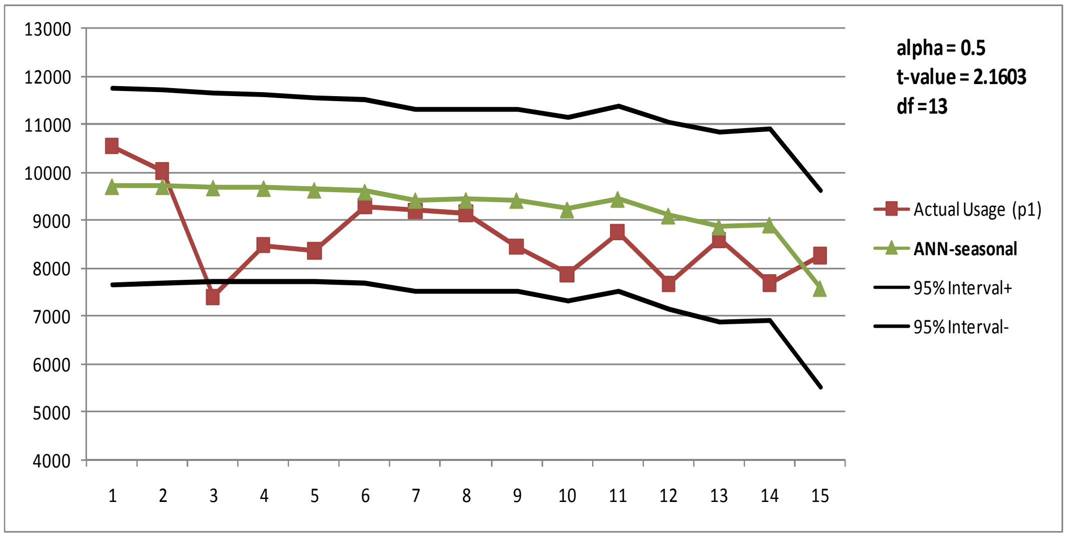


Figure 5.17 95% CI for ANN with Seasonal input forecast for p1

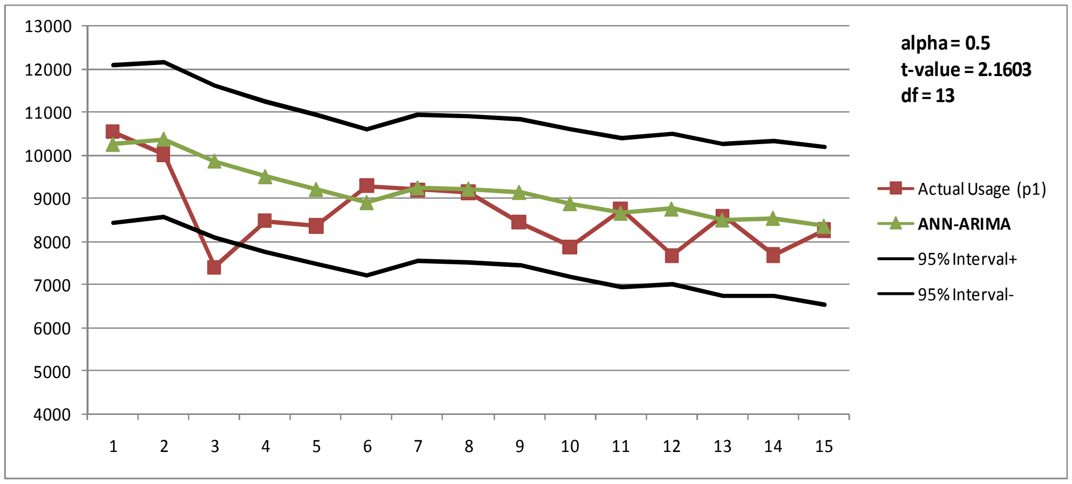


Figure 5.18 95% CI for ANN with ARIMA input forecast for p1

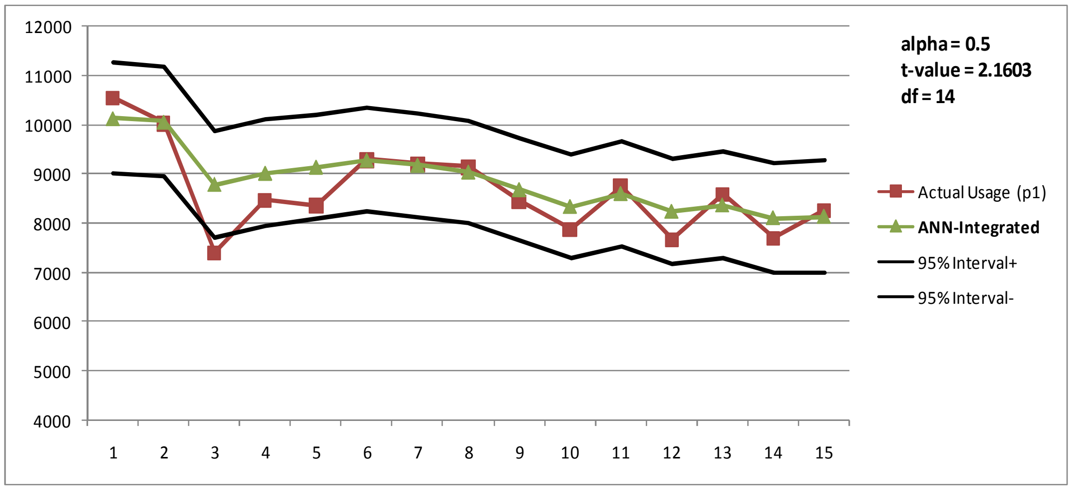


Figure 5.19 95% CI for ANN with Integrated input forecast for p1

We can see that, although the ARIMA forecast is within a narrower range, the ANN-Integrated forecast does not fluctuate as much. For planning purposes, the ANN-Integrated model has a higher level of confidence.

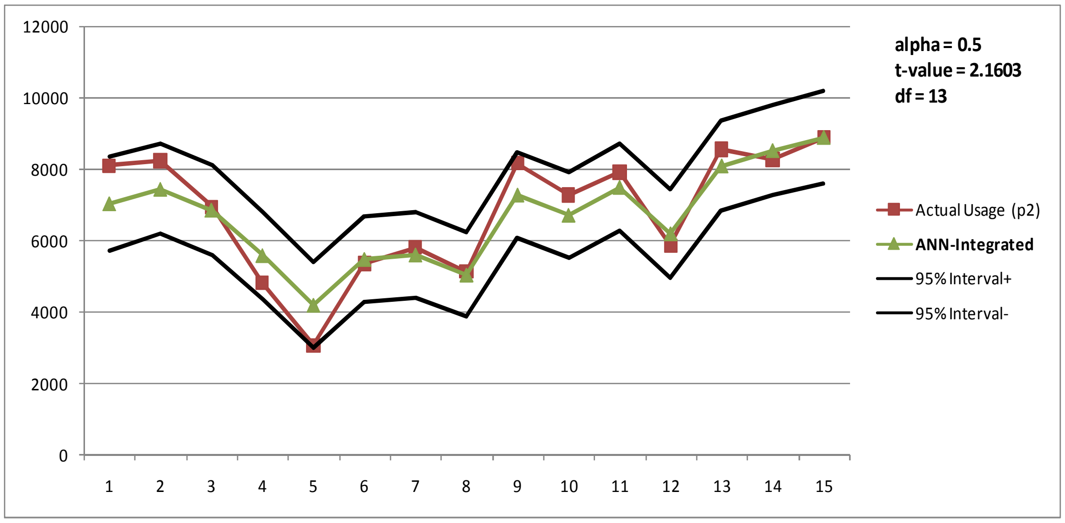


Figure 5.20 95% CI for ANN with Integrated input forecast for p2

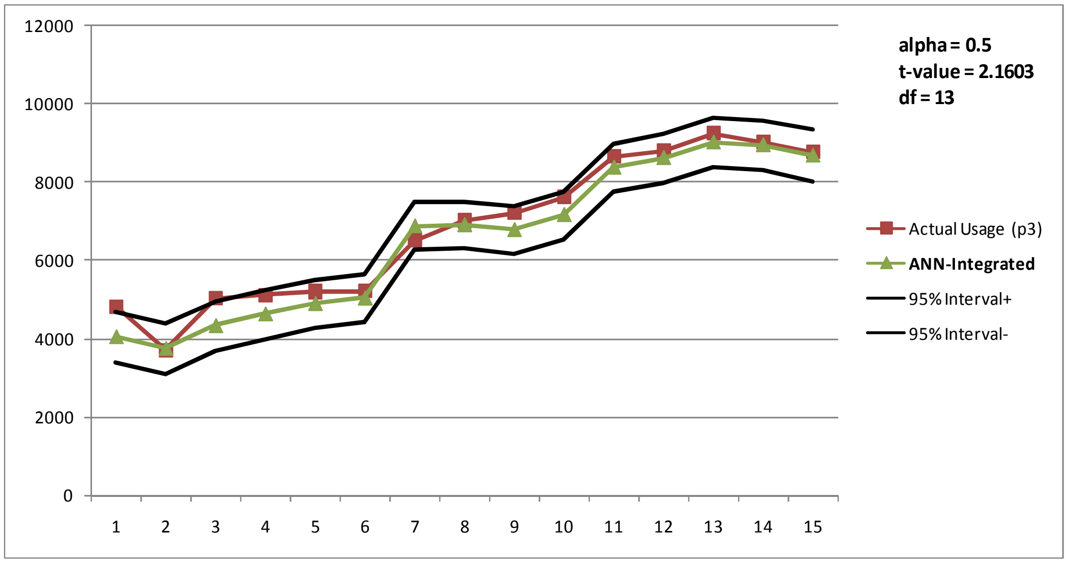


Figure 5.21 95% CI for ANN with Integrated input forecast for p3

Figures 5.20 and 5.21 show the forecasted values within their 95% CI bands for projects p2 and p3.

Figure 5.22 shows the 95% CI band for the Capacity Plan model for the CLP server.

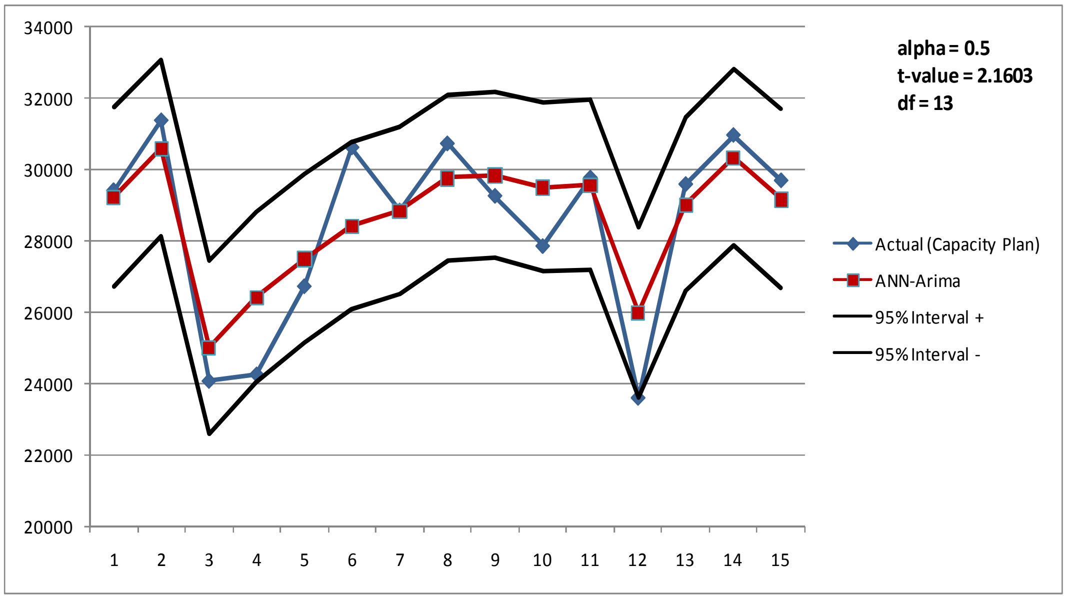


Figure 5.22 95% CI for ANN-ARIMA forecast for CLP Server capacity

The 15-week prediction cycle was selected to enable capacity planning for the length of one calendar quarter. A calendar quarter is 13 weeks. A 15-week cycle extends two weeks into the following quarter and provides continuity for planning the capacity. At the end of every calendar quarter, it allows a two-week overlap to develop the next quarter's forecast. For the purposes of planning EAS capacity, this is a short-range forecast.

In chapter 6, we discuss how the forecast for a calendar quarter may be used to develop both a medium- and long-term forecast. This is a topic for future research.

CHAPTER 6

CONCLUSIONS AND FUTURE RESEARCH

Technology companies are well known for their innovation. In some instances, the process of innovation may not lend itself to forecasting the resources needed in a predictable manner. However, as technology companies mature, they are introducing rigorous processes to manage their resources effectively. By effectively deploying their precious resources, they can improve their bottom line and ensure smooth execution of projects. Key resources for engineering projects in high technology product development may include computer hardware, software, storage, network bandwidth, and human resources. It is essential to forecast the resources required so that they are available when needed and projects under execution are not starved. The resource forecast is also required to drive the contract negotiations to acquire these resources from suppliers in the supply chain or to develop internally. In this research, we proposed a methodology for one common resource needed for high technology product development projects.

6.1 Conclusions

This research introduced a methodology to forecast license requirements for the projects in an enterprise. The project license requirements were then translated into a capacity plan for the organization. An understanding of the cyclic nature of the EAS usage by projects was used to enhance the quality of the forecast. We demonstrated that the usage patterns have a trend component, a cyclic component, and a random component. A rigorous methodology to extract the de-trended cyclic component was developed by applying Fourier analysis. Modifying the linear trend forecast with the cycles yielded a better forecast, as presented in Tables 5.1 to 5.4. However, including the cyclically modified trend forecast and an ARIMA forecast as inputs to the neural forecasting approach yielded even better results. The value of R, the coefficient of determination that quantifies the amount of variations in the forecast caused by variations in the actual series, for projects p1, p2, and p3 was 91.5%, 90.6%, and 98.5%, respectively. For the capacity plan, the R value was 84.9%. In the following section on the areas of future research, we propose the exploration of other approaches that may in some cases enhance the quality of the forecasts.

The methodology deployed to forecast EAS licenses was based on historical data. While the historical data is essential for developing forecasts, it does have some limitations. The accuracy of the forecast is a function of how well the current trends, whether linear or cyclical, persist into the future. Hence, any forecast obtained from historical data must be viewed in conjunction with other fundamental events. Fundamental events may be the cancellation of a project, resource rebalancing among projects based on the re-evaluation of priorities, or the consolidation of projects to align with changing market conditions or unforeseen technological challenges.

The Figure 6.1 usage data shows the occurrence of a fundamental event for project p1. Project p1 was a high priority for the enterprise but was running behind schedule. Around the April 2007 timeframe, the executive management decided to cancel a lower priority project and apply those resources to p1. The ability to forecast such events is beyond the scope of this methodology.

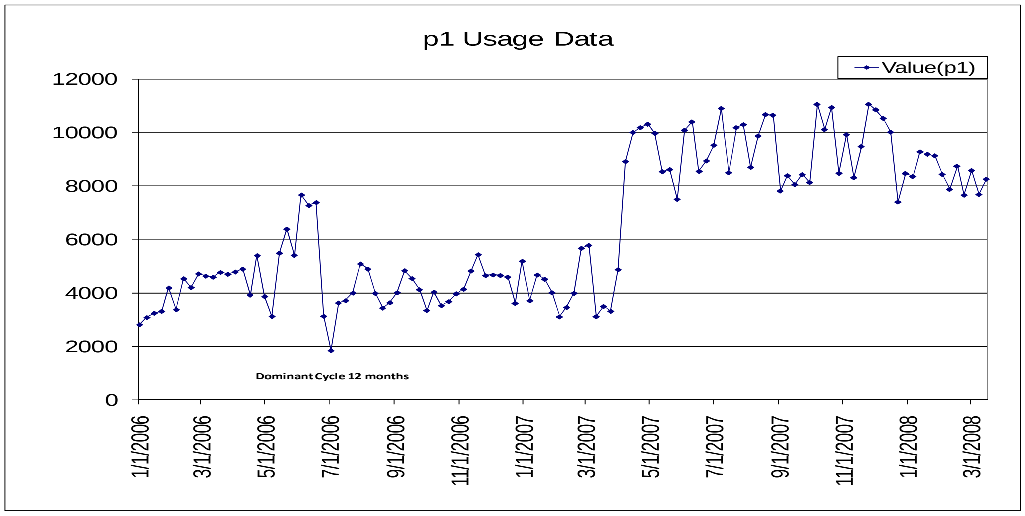


Figure 6.1 Project 1 usage data

Forecasts may be short-range, medium-range, or long-range. The exact meaning of short-range, medium-range, and long-range depends on the business practices of the enterprise. We need to consider what follow-on actions are possible, depending on the information given by the forecast. Let us consider a scenario where the duration of projects is between one to two years, the length of the business contract with the EAS suppliers is three years, and the contracts provide a mechanism to change the mix of EAS tools once every quarter. Table 6.1 lists the range of forecasts, the significance of the selected range, and the possible follow-on steps.

Table 6.1 Forecasting ranges

Range	Period	Follow-on actions	Other Inputs
Short-range	13 to 15 weeks	Change EAS Pool mix	Forecasts for all EAS
Medium-range	1 to 2 years	Align forecast with project plan	Project Plan
Long-range	3 years	Drive contract negotiations	Strategic Plan

In this research, we have concentrated on 15 weeks as the longest time period for forecasting project and enterprise capacity plans. As shown in Table 6.1, this is to address short-range requirements. The short-range forecast data enables rebalancing of the quantities of the various EAS tools required by the enterprise for the next quarter. To execute the rebalancing, we need forecasts of all of the various EAS tools in use by the enterprise from the same supplier.

The history of EAS forecasts and actual usage data can also form the foundation of the medium-range project and capacity forecasts. Once the complete usage history of a project is available, it may form the basis for subsequent projects. The project usage data is particularly relevant for products developed using the platform-derivative approach. Derivative projects based on the same platform are expected to be closely related in their EAS usage patterns. Projects based on different platforms may not exhibit similar usage patterns. This topic is also beyond the scope of this research and will be discussed in the next section on future research.

Long-range forecasts are necessary to develop a capacity forecast for long-term contract negotiations with the supply chain partners. Therefore, these forecasts impact the strategic plans of all of the suppliers in the supply chain. Suppliers can allocate R&D costs to align with the expected market size for their products. They can also better prepare for the challenges ahead. This is not a one-way process, as the suppliers also share their plans with the enterprise. The enterprise may then refine its strategic plan to align with the suppliers. If such alignment is not mutually beneficial, this may lead to the termination of the business relationship.

Figure 6.2 summarizes the impact of this research on the enterprise and the supply chain. In the next section, we review areas of future research.

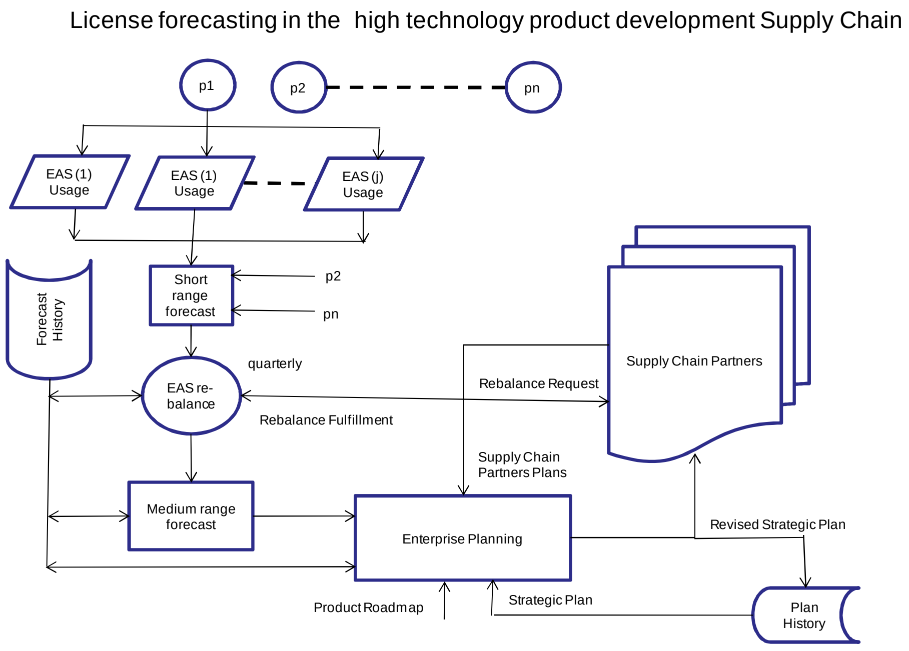


Figure 6.2 License forecasting in the high technology supply chain

6.2 Future Research

The areas for future research may be classified broadly into two categories. The first is the expansion of the scope of the research within the high technology supply chain, and the second is the investigation of other forecasting methodologies. In this section, we cover both of these categories.

6.2.1 Scope of future research in high technology supply chain

This research was focused on the forecasting of EAS licenses for high technology product development projects. There is potential to expand the scope of this research to include all of the variable resources required to complete a high technology product development project. In addition to software licenses, some other resources required by a product development project are human resources, computer hardware resources, disk storage, and third-party Intellectual Property (IP). Each of these resources, except third-part IP, may require unique forecasting approaches where historical data may be necessary.

Historical data plays a key role in creating human resource forecasts for projects. However, it may not be straightforward to deploy the historical data to high technology product development forecasts. This is due to the technological uncertainty and evolving development methodologies inherent in product development projects. Successive high technology product development projects are encumbered with increasing complexity and decreasing time to market. This is offset by the development of new, more efficient product development methodologies. The contradictory influences of increasing complexity and more efficient development methodologies make it extremely tenuous to forecast the human resource requirements.

In a global enterprise, not all of the skills required may be available at one location or even within the enterprise. Therefore, a human resource plan must take these factors into account.

As discussed earlier in chapters 2 and 3, high technology product development projects involve successive execution of EAS to converge to the desired result. The computer hardware is therefore an essential resource for projects. Here, the historical data also plays a key role in forecasting. However, the historical data must be applied carefully to come up with forecasts. For each successive project, the hardware available may be faster and more efficient. It may be necessary to normalize the performance of the new hardware available to the previous generation of hardware. Changes in product development methodology may also influence the computer hardware required for the project.

In a typical enterprise, the amount of disk storage increases regularly. This is because previous projects must be archived to support future errata work. New projects require additional storage. The storage history of past projects provides helpful estimates. However, the storage management solution in place also greatly influences what is required to support the project.

Another interesting factor is the interdependencies between the project resources. An increase in human resources may require an increase in EAS licenses, computer hardware, and disk storage. This relationship may be true for some type of EAS applications more so than others. For instance, EAS that is interactive may be directly correlated to the number of project team members of the relevant skill set deployed on the given project. For batch runs, the correlation between the number of project team members and the EAS licenses required may not be as strong.

The maximum number of EAS licenses that can be executed concurrently cannot exceed the total number of compute slots available. Another factor is the type of computer hardware, which must match the EAS job requirements. A typical example is the amount of memory installed on the machine to support the type of runs that are required to be made.

In general

$E = f(C, S, T, P)$

where

E = EAS forecast

C = number of compute slots available

S = available storage

T = size of project team

P = raw project forecast

In this research, besides regression-based trend forecasting techniques in the forecasting methodology, we also used the ARIMA and ANN techniques.

6.2.2 Other forecasting methodologies

Another area of future research is to explore other techniques. Some other techniques that may provide additional insights into forecasting for high technology product development include state space models, combined fuzzy logic and ANN, and application of chaos theory.

State space modeling is a traditional engineering approach to optimization and prediction with potential application to time series analysis. Measurements of various variables are taken, and these variables are used to define the “state” of the system. If all of the variables that define the state of the system are known, we can define the state of the system. Unfortunately, many business and economic phenomenon cannot be well defined in the state space because either not all variables are known or the relationships are obscure and ill-defined. If we can define indicators that serve as a proxy of the system, we may be able to use the Kalman filter to address not only discrete models using linear stochastic difference equations but also much more complicated relationships.

Fuzzy logic is a way to represent the ambiguity of a particular situation we are trying to define. Neural networks have frequently been combined with fuzzy logic in several applications. In the area of forecasting, we may be able to use neural networks as a forecasting tool with fuzzy variables as inputs. All of the principles of forecasting with neural networks are preserved; the only difference is that variables are not crisp, but fuzzy. A lot of the forecasting problems in the engineering and IT domain are intrinsically nonlinear, which is very difficult to model. In such cases, model-free approaches, such as fuzzy neural networks, may be the only answer.

What might appear to be random, probabilistic behavior could, in fact, be a behavioral pattern driven not by chance, but by a deterministic system. If the deterministic system is nonlinear, then potentially what appears to be random behavior is, in fact, chaotic behavior. Chaos theory that surrounds this approach is all about establishing these properties of the time series. In other words, establishing that the underlying process is linear, generated by a white noise process, or nonlinear. There is no single formula that we may use to produce forecasts by adopting this approach. However, chaotic time series may help us develop a better understanding of the time series to produce a more accurate forecast.

APPENDIX A

USAGE DATA FOR THE ENTERPRISE AND ALL PROJECTS

Week	**Enterprise (P) Usage**	**Project (p1) Usage**	**Project (p2) Usage**	**Project (p3) Usage**
1	2708	2812	1105	3780
2	8134	3080	1893	3880
3	4067	3243	1923	3811
4	4564	3309	1856	3742
5	4888	4188	1909	3981
6	5067	3376	1899	2899
7	4866	4531	1923	3011
8	5512	4204	2038	2721
9	5670	4717	2318	2441
10	5805	4634	2412	1998
11	5911	4589	2317	2081
12	6067	4769	2587	1986
13	6093	4699	2417	2244
14	6067	4785	2389	1941
15	6289	4893	2578	1723
16	6767	3921	2689	1929
17	7108	5398	2689	1850
18	5838	3862	2378	1920
19	5706	3119	3487	1936
20	7117	5490	3389	1955
21	8139	6389	3789	2206
22	8710	5412	3892	2786
23	9866	7671	3795	2928
24	10068	7275	4275	3782
25	10068	7390	4489	3556
26	9068	3128	5584	4084
27	7015	1837	2657	4731
28	11406	3622	2739	4201
29	11375	3712	2128	4101
30	12763	3997	1987	3986
31	13097	5089	2128	4089
32	14614	4893	6008	3689
33	12926	3982	5982	3561
34	11193	3431	5817	2689
35	12545	3632	5967	2709
36	12691	4012	5834	2108
37	13818	4832	6026	2250
38	16712	4541	6834	2101
39	17037	4123	6978	1989
40	10852	3342	5745	3838

Week	Enterprise (P) Usage	Project (p1) Usage	Project (p2) Usage	Project (p3) Usage
41	13109	4034	5992	4314
42	11779	3522	6239	3731
43	12665	3677	6176	3946
44	12696	3968	5945	4410
45	13765	4142	6851	4609
46	14413	4821	6360	4519
47	13777	5433	6215	4704
48	13745	4650	6961	4981
49	14528	4674	8020	5899
50	15627	4652	5188	5976
51	14697	4593	4768	5901
52	12001	3609	3516	5521
53	15097	5189	2813	5432
54	12433	3711	2232	5117
55	14800	4673	1707	4591
56	14202	4516	1823	4780
57	13480	4011	2812	4211
58	6842	3104	5018	3601
59	8014	3459	6589	3882
60	13275	3989	6927	3654
61	16058	5672	7912	3010
62	16622	5782	7845	2992
63	6608	3113	5758	2423
64	9295	3490	5983	2216
65	10287	3310	6028	3704
66	16057	4872	7578	3711
67	20288	8923	8634	4989
68	25712	10012	9027	4403
69	25535	10192	9878	5778
70	20754	10322	8218	6080
71	18902	9982	8126	5989
72	20773	8543	8321	6211
73	21191	8622	8441	7301
74	21348	7505	7706	7614
75	23732	10093	7978	5770
76	23787	10409	8403	5681
77	24291	8555	9510	4915
78	23619	8949	9160	4231
79	24976	9533	6999	3901
80	25371	10911	4306	3710

APPENDIX B

FOURIER ANALYSIS REGRESSION RESULTS

SUMMARY OUTPUT **Project p1**

Regression Statistics	
Multiple R	0.268297388
R Square	0.071983488
Adjusted R Squ	0.038541452
Standard Error	2611.448053
Observations	116

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	4	58716876.7	14679219.17	2.1524852	0.079046802
Residual	111	756982364	6819660.935		
Total	115	815699240			

	Coefficients	*Standard Erro*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	6559.139019	245.323686	26.73667238	3.439E-50	6073.013773	7045.264265	6073.013773	7045.264265
X Variable 1	-328.9287938	352.416984	-0.933351141	0.3526646	-1027.266552	369.4089641	-1027.26655	369.4089641
X Variable 2	-557.8336744	340.411416	-1.63870437	0.1041063	-1232.381598	116.7142495	-1232.3816	116.7142495
X Variable 3	631.7592528	345.255423	1.829831512	0.0699588	-52.38739509	1315.905901	-52.3873951	1315.905901
X Variable 4	-281.9303506	345.177384	-0.816769474	0.4158097	-965.9223593	402.0616581	-965.922359	402.0616581

SUMMARY OUTPUT **Project p2**

Regression Statistics	
Multiple R	0.410179883
R Square	0.168247537
Adjusted R Squ	0.138274475
Standard Error	2207.168867
Observations	116

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	4	109382720	27345680.1	5.6132916	0.000375057
Residual	111	540746979	4871594.407		
Total	115	650129700			

	Coefficients	*Standard Erro*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	5705.863278	207.34504	27.51868705	2.125E-51	5294.995251	6116.731304	5294.995251	6116.731304
X Variable 1	21.1320642	297.859187	0.070946491	0.943568	-569.0957812	611.3599096	-569.095781	611.3599096
X Variable 2	-772.2760185	287.712205	-2.684196236	0.0083842	-1342.396942	-202.155095	-1342.39694	-202.155095
X Variable 3	-159.883745	291.80631	-0.547910513	0.5848535	-738.1174111	418.349921	-738.117411	418.349921
X Variable 4	-1076.106403	291.740353	-3.688575795	0.0003509	-1654.20937	-498.003437	-1654.20937	-498.003437

SUMMARY OUTPUT **Project p3**

Regression Statistics	
Multiple R	0.214358384
R Square	0.045949517
Adjusted R Squ	0.011569319
Standard Error	1883.470397
Observations	116

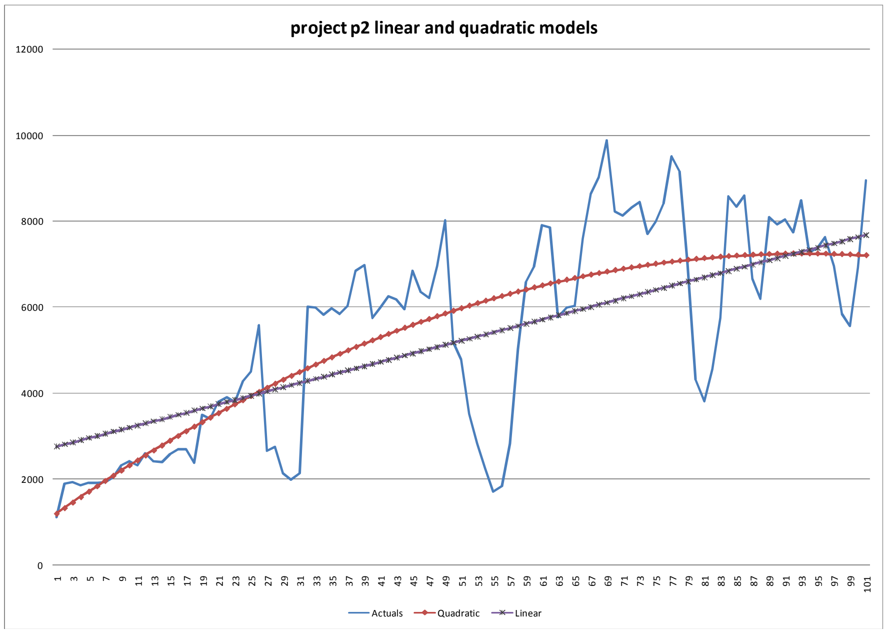
ANOVA

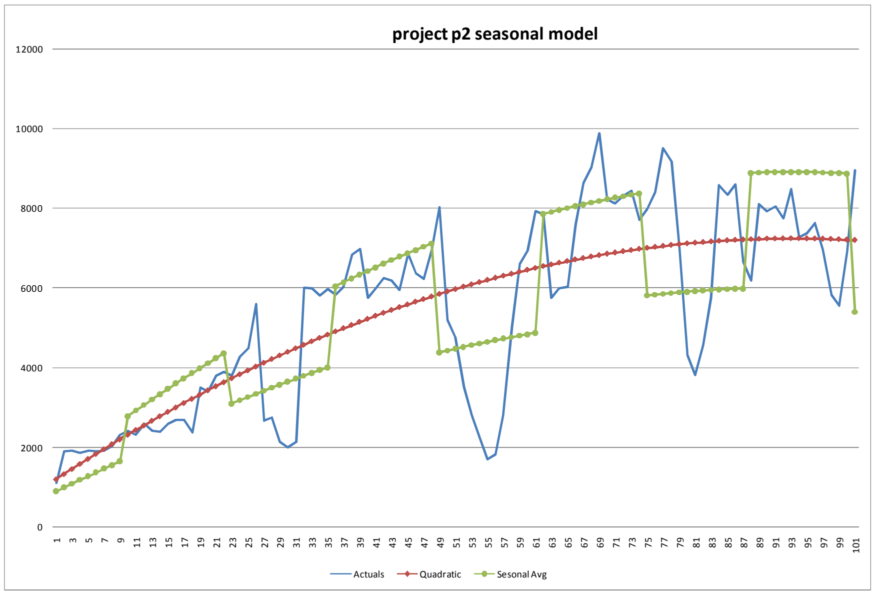
	df	*SS*	*MS*	*F*	*Significance F*
Regression	4	18964883.1	4741220.771	1.3365111	0.260893251
Residual	111	393768142	3547460.736		
Total	115	412733025			

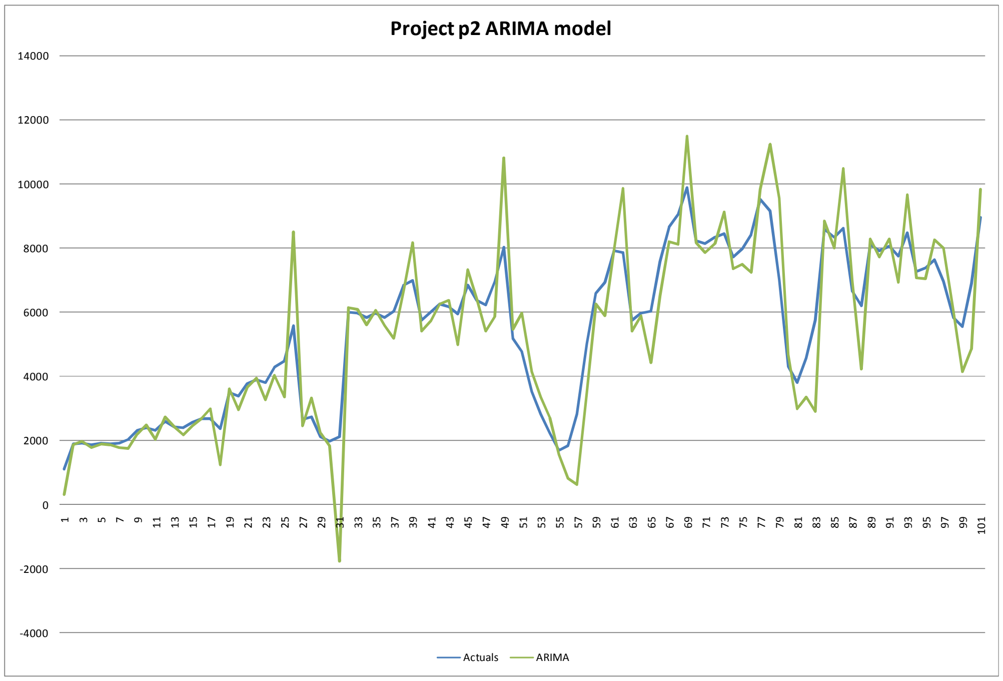
	Coefficients	*Standard Erro*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	4554.013147	176.936279	25.73815378	1.314E-48	4203.40211	4904.624184	4203.40211	4904.624184
X Variable 1	531.8007738	254.175822	2.092255546	0.0386955	28.13442956	1035.467118	28.13442956	1035.467118
X Variable 2	-252.6406011	245.516974	-1.029014805	0.3057098	-739.1488602	233.8676579	-739.14886	233.8676579
X Variable 3	-36.39664818	249.010647	-0.146165028	0.8840562	-529.8278527	457.0345563	-529.827853	457.0345563
X Variable 4	55.64912719	248.954362	0.22353144	0.8235328	-437.670546	548.9688003	-437.670546	548.9688003

APPENDIX C

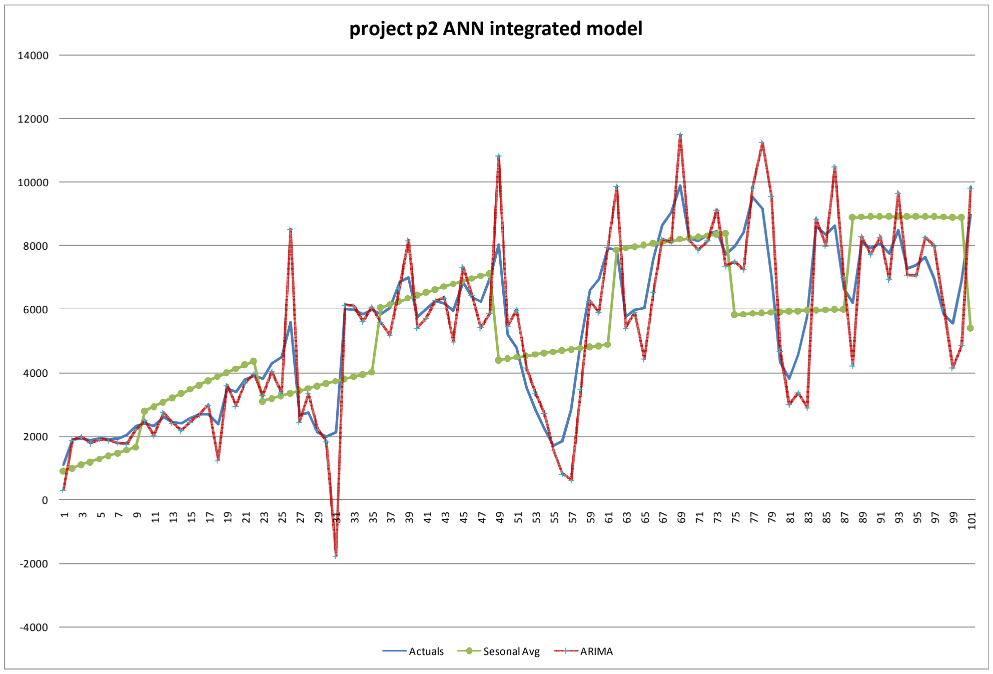
FORECASTING MODELS FOR PROJECTS P2 AND P3


project p2 linear and quadratic models
12000
10000
8000
6000
4000
2000
0
Actuals
Quadratic
Linear

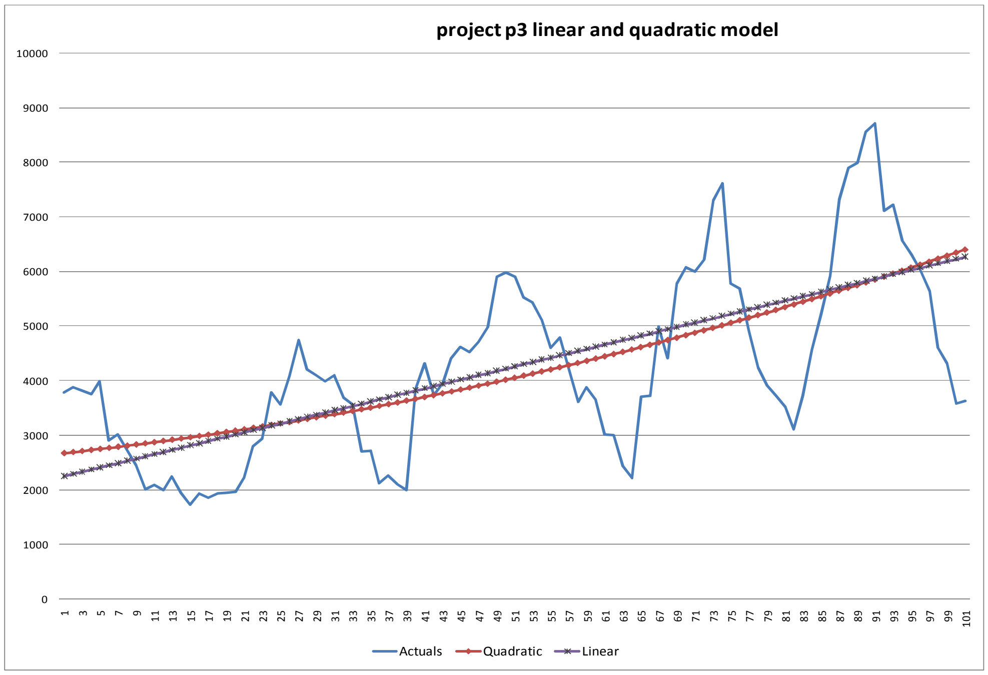

project p2 seasonal model
12000
10000
8000
6000
4000
2000
0
Actuals
Quadratic
Sesonal Avg

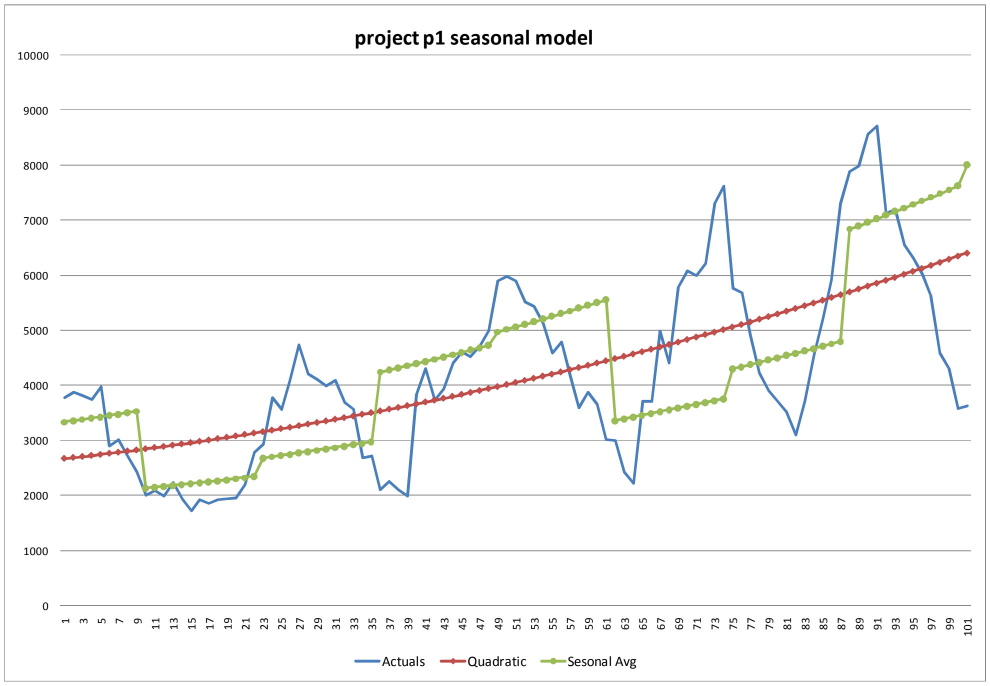


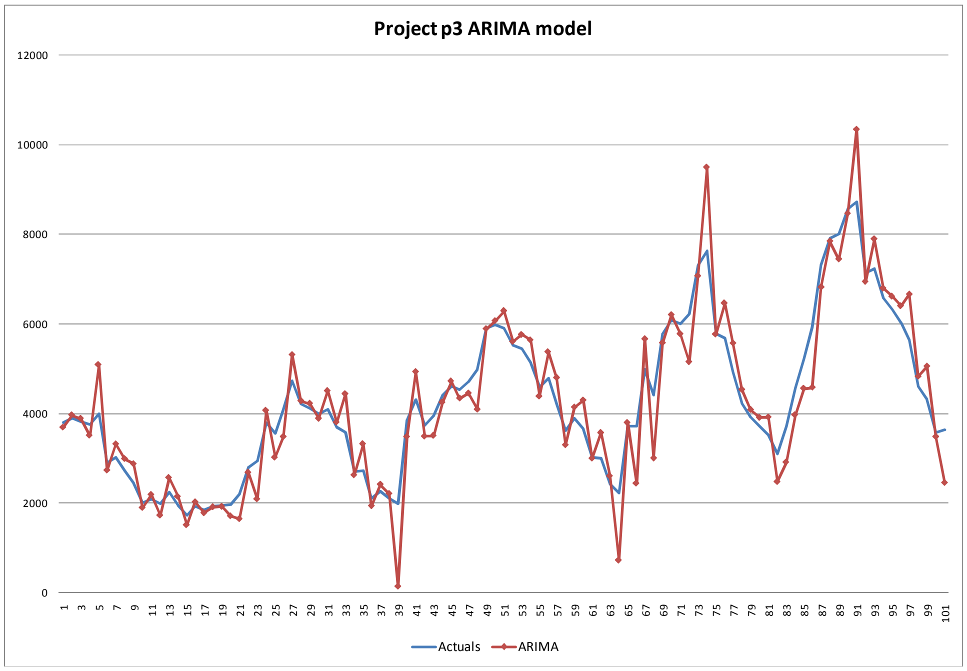
Project p2 ARIMA model
14000
12000
10000
8000
6000
4000
2000
0
-2000
-4000
Actuals
ARIMA

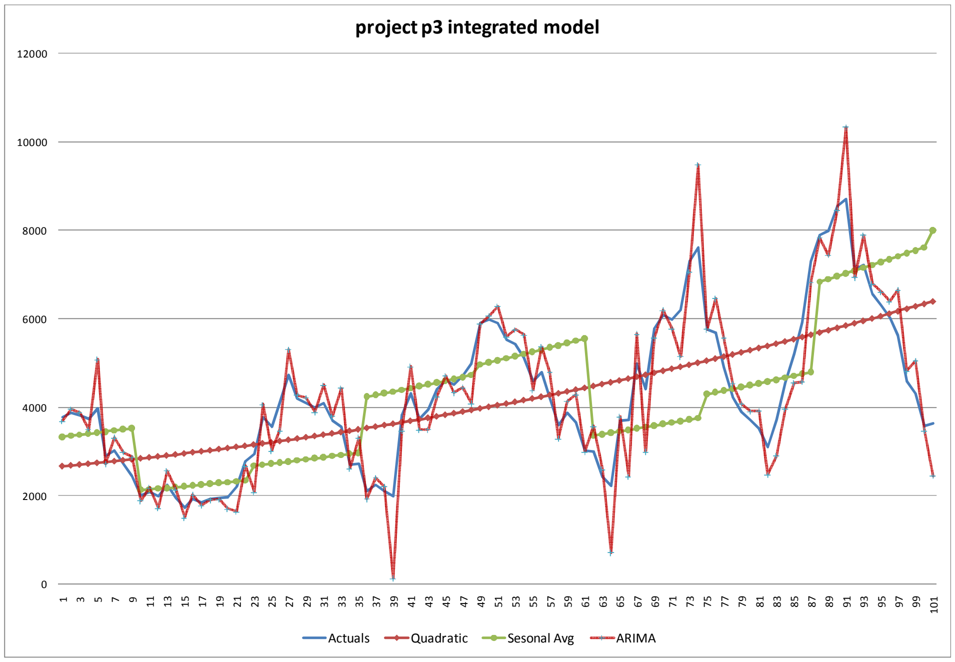


project p2 ANN integrated model
14000
12000
10000
8000
6000
4000
2000
0
-2000
-4000
Actuals
Sesonal Avg
ARIMA


project p3 linear and quadratic model
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
Actuals
Quadratic
Linear


project p1 seasonal model
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
Actuals
Quadratic
Sesonal Avg


Project p3 ARIMA model
12000
10000
8000
6000
4000
2000
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99 101
Actuals
ARIMA


project p3 integrated model
12000
10000
8000
6000
4000
2000
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99 101
Actuals
Quadratic
Sesonal Avg
ARIMA

REFERENCES

Ahituv, N., Hadass, M., & Neumann, S. (1984). A flexible approach to information system development. *MIS Quarterly*, 69-78.

Anderson, D.R., Sweeney, D.J., & Williams, T.A. (2004). Quantitative methods for business, 175-206.

Cash, J,I., McFarlan, F.W., & McKenny, J.L. (1988). *Corporate information systems management: The issues facing senior executives* (2nd ed.). Homewood, IL: Dow Jones-Irwin.

Chatfield, C. (1998). Time series forecasting with neural networks. *IEEE*, 419-427.

Dvir, D., Lipovetsky, S., Shenhar, A.J., & Tishler, A. (1998). In search of project classification: A non-universal approach to project success factors. *Research Policy*, *27*, 915-935.

Faraway, J., & Chatfield, C. (1998). Time series forecasting with neural networks: A comparative study using the airline data. *Applied Statistics, 47*, 231-250.

Hopp, W.J., & Spearman, M.J.(2001). *Factory physics* (2nd ed., pp. 626-631). New York: McGraw-Hill Higher Education.

Kennington, J., & Whitler, J. (1999). An efficient decomposition algorithm to optimize spare capacity in a telecommunications network. *INFORMS Journal of Computing*, *11*(2), 149-160.

Krishnan, V. (1996). Managing the simultaneous execution of coupled phases in concurrent product development. *IEE Transactions on Engineering Management, 43*(2), 210-217*.*

Krishnan, V., & Bhattacharya, S. (2002). Technology selection and commitment in new product development: The role of uncertainty and design flexibility. *Management Science, 48*(3), 313-327.

Krishnan, V., & Gupta, S. (2001). Appropriateness and impact of platform-based product development. *Management Science, 47*(1), 52-68.

Kuvulmaz, J., & Usanmaz, S. (2005). Time series forecasting by means of linear and nonlinear models. *4th Mexican International Conference on Artificial Intelligence*, *V3789,* 504-513.

Laguna, M. (1998). Applying robust optimization to capacity expansion of one location in telecommunications with demand. *Management Science, 44*(11), S101-110.

Lee, L., & Chun, H.W. (1992). An ANN approach to spare capacity planning. *IEEE*, 891-895.

McClelland , J., Rumelhart D., & PDP Research Group. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition(Vol. 1: Foundations)*. Cambridge, MA: MIT Press.

Meimban, III, J., Morris, J., & Govett, R. (1992). The evolution of wood-fired cogeneration investment using monte-carlo simulation. *The Engineering Economist, 37*(2), 115-125.

Papagiannaki, K., Taft, N., & Diot, C. (2005). Long-term forecasting of internet backbone traffic. *IEE Transactions on Neural Networks, 16*(5), 1110-1124.

Park, K., & Kim, S. (2002). A capacity planning model of unreliable multimedia service systems. *The Journal of Systems and Software, 63,* 669-76.

Ragsdale, C. (2007). Spreadsheet modeling and decision analysis. *A practical introduction to management science*(5th ed., pp. 485-536). Mason: Thomson Southwestern.

Rumelhart, D., Hinton, G., & Williams, R. (1988). Learning internal representations by error propagation. In J. Anderson & E. Rosendield (Eds.).*Neurocomputing* (pp. 675-695). Cambridge, MA: MIT Press.

Saito, M., & Kakemoto, Y. (2004). Demand forecasting by the neural network with Fourier transform. *Proceedings of International Joint Conference on Neural Networks, Vol4*, 2759-2763.

Shenhar, A.J. (1993). From low- to high-tech project management. *R&D Management, 23*(3), 199-214.

Shenhar, A.J. (1997). The new taxonomy of systems: Toward an adaptive systems engineering framework. *IEEE Transactions on Man and Cybernetics—Part A: Systems and Humans, 27*(2), 137-145.

Shenhar, A.J., Dvir, D., & Shulman, Y. (1995). A two-dimensional taxonomy of products and innovations. *Journal of Engineering Technology Management, 12*, 175-200.

Stallings, W. (1997). *Data and computer communications* (5th ed., pp. 34-70). New Jersey: Prentice Hall.

Temponi, C., & Malhotra, R. (2002) *.* Project management challenges of high technology product development. *Proceeding of the 2002 IEEE International Engineering Management Conference,* (pp 80-88) Cambridge: IEEE Press.

Yang, Y.H., Haddad, K., & Chow, C.W. (2001). Capacity planning using monte carlo simulation: An illustrative application of commonly available PC software. *Managerial Finance 27*(5), 33-54.

Wheelwright, S.C., & Clark, K.B. (1992, March/April). Creating project plans to focus. *Harvard Business Review*, 70-82.

BIOGRAPHICAL INFORMATION

Rajiv Malhotra is an Engineering Manager at Advanced Micro Devices in Austin, Texas, in the area of Management of Technology. He has over 25 years of experience in the semiconductor industry and as a consultant in the Information Technology (IT) industry at various levels of Management and Technology Development. His technology experience is in the Electronic Design Automation (EDA) Software Development, microchip Product Development Methodology, Project Management, Deployment of Technology and Software Engineering. He has management experience in software Product Life Cycle Management, Negotiation and Management of global EDA software contracts, Management of R&D budgets, Management of microchip Product Development R&D teams and IT Management and consulting for small businesses. Rajiv Malhotra has a B.S. in Electrical Engineering, M.S. in Industrial Engineering, an M.S. in Computer Engineering and a PhD in Industrial Engineering. His research interests are in the area of Management of Technology, Management of Knowledge, Business and Technology Forecasting, and the Financial Valuation of engineering software and services in the high technology industry. His research work has been presented at several professional conferences including the IEEE Engineering Management Conferences, the International Conference on Industrial Engineering and Systems Management, and the annual AMD Engineering Conferences. Rajiv is also an active member of several professional organizations.