FLEET REPLACEMENT MODELING

Final Report

SPR 670

Oregon Department of Transportation

FLEET REPLACEMENT MODELING

Final Report

SPR 670

by

David S. Kim and J. David Porter
Phillip Kriett, Wilson Mbugua and Tristan Wagner
School of Mechanical, Industrial and Manufacturing Engineering
Oregon State University

for

Oregon Department of Transportation
Research Section
200 Hawthorne Ave. SE, Suite B-240
Salem OR 97301-5192

and

Federal Highway Administration
400 Seventh Street, SW
Washington, DC 20590-0003

July 2009

Technical Report Documentation Page

1. Report No.

FHWA-OR-RD-10-01

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle

Fleet Replacement Modeling

5. Report Date

July 2009

6. Performing Organization Code

7. Author(s)

David S. Kim & J. David Porter
Phillip Kriett, Wilson Mbugua & Tristan Wagner
School of Mechanical, Industrial and Manufacturing Engineering
Oregon State University
Corvallis, OR 97331

8. Performing Organization Report No.

9. Performing Organization Name and Address

School of Mechanical, Industrial and Manufacturing Engineering
Oregon State University
Corvallis, OR 97331

10. Work Unit No. (TRAIS)

11. Contract or Grant No.

12. Sponsoring Agency Name and Address

Oregon Department of Transportation
Research Section
200 Hawthorne Ave. SE, Suite B-240
Salem, OR 97301-5192

and

Federal Highway Administration
400 Seventh Street, SW
Washington, DC 20590-0003

13. Type of Report and Period Covered

Final Report

14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This project focused on two interrelated areas in equipment replacement modeling for fleets. The first area was research-oriented and addressed a fundamental assumption in engineering economic replacement modeling that all assets providing a similar service are equally utilized. It is shown that if assets providing a similar service are not equally utilized, than overall operational costs of a fleet increases. The second area addressed the need of the Oregon Department of Transportation (ODOT) Fleet Services Section for a modern, user-friendly, well constructed and documented fleet condition model. The state-of-the-art in equipment replacement modeling and the research on effective prioritization measures computable from existing data is discussed. A new model is developed utilizing the highest value of (Age/Age standard)+(Usage/Use standard) as a measure to prioritize equipment. The model is described and it is shown through simulation that the prioritization measure utilized performs better than several other suggested measures. This project was conducted jointly with the ODOT Fleet Services Section and the ODOT Research Section.

17. Key Words

FLEET, VEHCILES, FLEET REPLACEMENT, FLEET MODELING, FLEET LIFE-SPAN, FLEET LIFE-CYCLE, VEHICLE LIFE-CYCLE, FLEET DECOMMISIONING, REPLACEMENT FORMULA

18. Distribution Statement

Copies available from NTIS, and online at
http://www.oregon.gov/ODOT/TD/TP_RES/

19. Security Classification (of this report)

Unclassified

20. Security Classification (of this page)

Unclassified

21. No. of Pages

177

22. Price

SI* (MODERN METRIC) CONVERSION FACTORS

APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
in	inches	25.4	millimeters	mm
ft	feet	0.305	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km
		AREA		
in^2	square inches	645.2	millimeters squared	mm^2
ft^2	square feet	0.093	meters squared	m^2
yd^2	square yards	0.836	meters squared	m^2
ac	acres	0.405	hectares	ha
mi^2	square miles	2.59	kilometers squared	km^2
		VOLUME		
fl oz	fluid ounces	29.57	milliliters	ml
gal	gallons	3.785	liters	L
ft^3	cubic feet	0.028	meters cubed	m^3
yd^3	cubic yards	0.765	meters cubed	m^3
NOTE: Volumes greater than 1000 L shall be shown in m^3.				
		MASS		
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams	Mg
		TEMPERATURE (exact)		
°F	Fahrenheit	(F-32)/1.8	Celsius	°C

APPROXIMATE CONVERSIONS FROM SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
mm	millimeters	0.039	inches	in
m	meters	3.28	feet	ft
m	meters	1.09	yards	yd
km	kilometers	0.621	miles	mi
		AREA		
mm^2	millimeters squared	0.0016	square inches	in^2
m^2	meters squared	10.764	square feet	ft^2
m^2	meters squared	1.196	square yards	yd^2
ha	hectares	2.47	acres	ac
km^2	kilometers squared	0.386	square miles	mi^2
		VOLUME		
ml	milliliters	0.034	fluid ounces	fl oz
L	liters	0.264	gallons	gal
m^3	meters cubed	35.315	cubic feet	ft^3
m^3	meters cubed	1.308	cubic yards	yd^3
		MASS		
g	grams	0.035	ounces	oz
kg	kilograms	2.205	pounds	lb
Mg	megagrams	1.102	short tons (2000 lb)	T
		TEMPERATURE (exact)		
°C	Celsius	1.8C+32	Fahrenheit	°F

*SI is the symbol for the International System of Measurement

ACKNOWLEDGEMENTS

The authors would like to thank Barnie Jones of ODOT Research for supporting this project, and Jon Lazarus of ODOT Research for his help with running this project and keeping us on time. We would also like to thank both Ed Hinman and Bill Ward at Fleet Services and Richard Munford from the Transportation Development Division for working with us. This project could not have been completed without their help and cooperation.

DISCLAIMER

This document is disseminated under the sponsorship of the Oregon Department of Transportation and the United States Department of Transportation in the interest of information exchange. The State of Oregon and the United States Government assume no liability of its contents or use thereof.

The contents of this report reflect the view of the authors who are solely responsible for the facts and accuracy of the material presented. The contents do not necessarily reflect the official views of the Oregon Department of Transportation or the United States Department of Transportation.

The State of Oregon and the United States Government do not endorse products of manufacturers. Trademarks or manufacturers' names appear herein only because they are considered essential to the object of this document.

This report does not constitute a standard, specification, or regulation.

FLEET REPLACEMENT MODELING

TABLE OF CONTENTS

APPENDICES

List of Figures

List of Tables

EXECUTIVE SUMMARY

This project focused on two interrelated areas in equipment replacement modeling for fleets. The first area was research-oriented and addressed fundamental assumptions in engineering economic replacement modeling. The second area addressed the need of the Oregon Department of Transportation (ODOT) Fleet Services Section for a modern, user-friendly, well constructed and documented fleet condition model. This project was conducted jointly with the ODOT Fleet Services Section and the ODOT Research Section. The main components of this project were as follows:

- Perform an assessment of the state-of-the-art in fleet replacement models in use by other DOTs, government agencies, and private industry.
- Evaluate equipment replacement prioritization models that utilize available equipment cost and usage data.
- Conduct a theoretical analysis of equipment replacement that explicitly considers the interdependencies between replacement decisions and equipment utilization.
- Design, develop, and test a new and improved fleet condition model with replacement prioritization reporting capabilities.

The project findings that correspond to each of the major components listed above are summarized as follows:

- Replacement practices at ODOT were in line with other DOTs throughout the nation, and a survey of the academic literature revealed little applicable work that addressed fleet replacement with interdependencies between replacement decisions and equipment utilization as found in ODOT data.
- An evaluation using simulation showed that replacing the oldest equipment in a class first was the most cost effective replacement prioritization model.
- If equipment in a fleet is utilized such that newer equipment is generally preferred over older equipment, thus resulting in decreased equipment usage with age, then
  - fleet operating costs will be higher than if all equipment is equally utilized; and
  - the age standard for the fleet should be younger than if equipment in the fleet is equally utilized.
- A new software tool was developed that includes the following features:
  - It implements a formal Entity Relationship model;
  - It produces several reports automatically;
  - Recurrent procedures are dynamic;
  - It consolidates replacement rankings with fleet plan data; and
  - It implements an error-proof, user-friendly graphical user interface.

1.0 INTRODUCTION

1.1 PROJECT MOTIVATION

All state departments of transportation (DOTs) maintain large fleets of equipment. This equipment represents a substantial investment and is a vital set of resources that is used to maintain roads and highways. An important and difficult challenge of managing such a large amount of equipment is deciding when to replace existing equipment. Such decisions have a clearly documented economic impact, and they also affect the ability of the fleet to provide required equipment when needed. In particular, the Oregon Department of Transportation (ODOT) Fleet Services Section provides management of ODOT's fleet, which consists of over 5,000 pieces of active equipment worth approximately \$340 - \$390 million. This equipment includes a variety of small and large trucks, cars, and heavy machinery such as graders, bulldozers, and many types of tractors.

Prior to this study, ODOT Fleet Services was using a relatively simple, internally developed, condition (replacement) model for prioritizing which equipment to replace, and for allocating replacement funds to equipment crews throughout Oregon. The validity of the condition model was questionable both within Fleet Services and with the various equipment crews that receive and evaluate the model output.

1.2 PROJECT ORGANIZATION

This project focused on two interrelated topics in equipment replacement modeling for fleets. The first topic was research-oriented and addressed fundamental assumptions in engineering economic replacement modeling. The second topic addressed the need of the ODOT Fleet Services Section for a modern, user friendly, well constructed and documented fleet condition model. This second topic was application-oriented and was addressed considering real-world constraints related to data collection and internal management processes

1.2.1 Research-Oriented Tasks

Data from ODOT Fleet Services clearly showed that for many large equipment classes, newer equipment was being utilized more than older equipment. As an example of how this may occur, it may be common for users of passenger vehicles in a fleet to request newer vehicles to drive when they are available. This decreasing utilization of older equipment was occurring as the overall service provided by the fleet stayed constant. The end effect of this was that replacement decisions not only affected the specific equipment being replaced, but also the utilization of other equipment in the same class (assuming the replacement is new). Furthermore, it was known that reduced utilization of a single piece of equipment as it ages extended the equipment's economic life. This research examined how these facts affected equipment replacement models that could be utilized by ODOT, and also how these facts affected equipment replacement decisions in general.

The research-oriented portion of this project had three main components. The first component was an assessment of the state-of-the-art in fleet replacement models in use by other DOTs, government agencies, and private industry. This assessment also included a literature survey of engineering economic replacement models.

The second component was to evaluate simple models (i.e., models that could be incorporated into new fleet condition model software) that utilize available equipment cost and usage data to prioritize equipment for replacement. The measure used to evaluate these models was the cost of operating a fleet of equipment over a long period of time, and simulation models were developed to conduct the evaluation. By statistically representing the use and costs of equipment in specific equipment classes in ODOT's fleet, these simulation models indirectly accounted for the interdependencies between replacement decisions and equipment utilization that occurs in such fleets.

The third component was a theoretical analysis of equipment replacement that explicitly considered the interdependencies between replacement decisions and equipment utilization. As described earlier, this is a replacement situation where assets within a class are utilized less as they get older, but the overall service provided by the fleet is constant. This is the replacement situation faced by ODOT and many other DOTs. This was a new type of engineering economic replacement problem that was analyzed with a combination of mathematical analysis and simulation.

1.2.2 Application-Oriented Tasks

The objective of the applications aspect of this project was to provide ODOT Fleet Services with a reliable, accurate, user-friendly, and valid fleet condition model to assist them in better managing equipment replacement decisions.

ODOT Fleet Services previously used a replacement model developed in 1998 (Randhawa and Douglas). This model had validity issues and required extensive amounts of manual data manipulation and manual data analysis. More recently, ODOT Fleet Services has utilized a replacement model developed internally in the form of a database application. The development platform was Microsoft Access 2003. Although a significant improvement when compared to its predecessor, ODOT Fleet Services has found that the validity of the model is questionable both within Fleet Services and with the various users of the model output. Furthermore, the database application has no underlying formal model and thus it is very difficult to use and maintain. Providing ODOT Fleet Services with an up-to-date fleet condition model will facilitate faster and better equipment replacement decisions.

There were two main components involved with providing ODOT Fleet Services with a reliable, accurate, user friendly, and valid fleet condition model (see right side of Figure 1.1). The first was to perform an assessment of the state-of-the-art in fleet replacement models in use by other DOTs, government agencies, and private industry. The intent of this work was to benchmark ODOT equipment replacement procedures with other similar organizations. This also served as the basis for determining current best practices, as well as what could effectively be implemented under various budget limitations. An assessment of data availability at ODOT Fleet Services was also conducted, and this was matched to the requirements of replacement models in the research literature.

The second component was the design and development of a new and improved fleet condition model that addressed the shortcomings of the prior system. The new model was implemented as a Microsoft Access database application. The main steps in this phase included the design of a robust relational database model, development of several database queries and macros, the production of a series of reports critical to support decisions made by ODOT Fleet Services personnel, and user testing and validation of the condition model. A simplistic, stand-alone application format was chosen to allow ODOT to incorporate the coding in any future software developments.

1.3 BENEFITS

Figure 1.1 is a diagram showing the different components of this project and how they were integrated. The accomplishment of the research and applications-oriented tasks stated will assist ODOT and other DOTs nationwide to better assess and manage equipment needs. The creation of a more effective equipment replacement system will be of tremendous benefit both in potential labor and equipment dollar savings. Additionally, it will be possible to identify the limitations of current research when considering the real-world characteristics and availability of data.

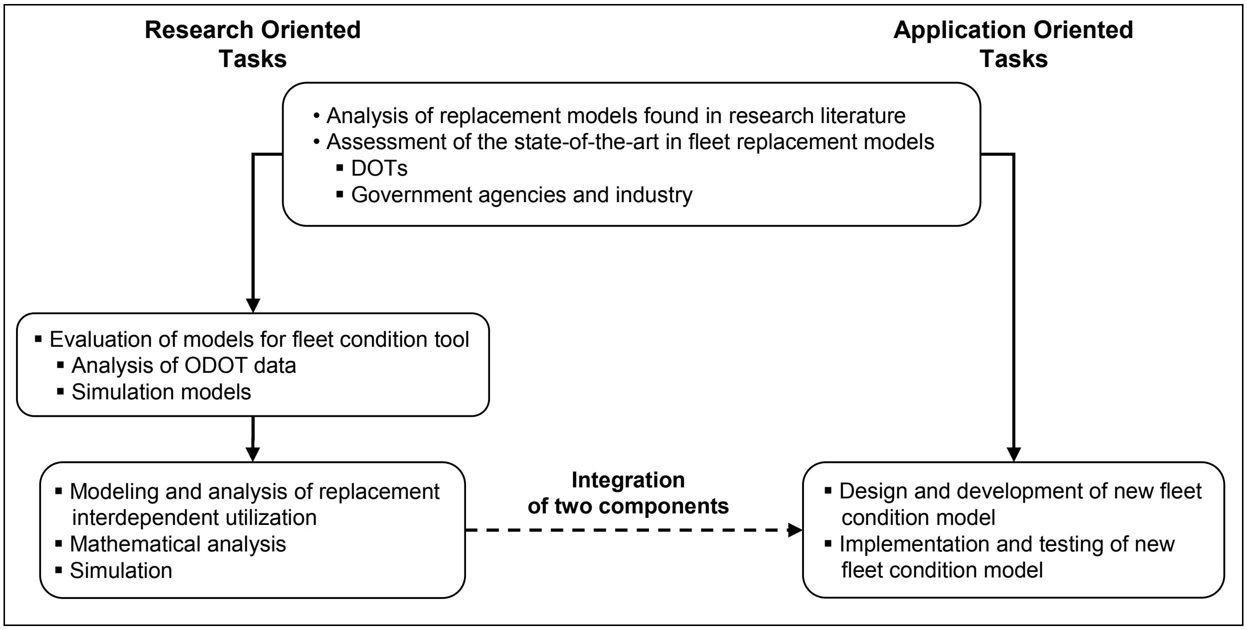

Research Oriented Tasks
Application Oriented Tasks
• Analysis of replacement models found in research literature
• Assessment of the state-of-the-art in fleet replacement models
▪ DOTs
▪ Government agencies and industry
▪ Evaluation of models for fleet condition tool
▪ Analysis of ODOT data
▪ Simulation models
▪ Modeling and analysis of replacement interdependent utilization
▪ Mathematical analysis
▪ Simulation
Integration of two components
▪ Design and development of new fleet condition model
▪ Implementation and testing of new fleet condition model

Figure 1.1: Research and application methodologies

2.0 LITERATURE REVIEW

The purpose of this assessment was to identify the state-of-the-art in equipment replacement models in both practice and in the published research literature.

Fleet managers and researchers have recognized the problem of equipment replacement for a long time, and they have developed a variety of strategies to address it. In order to complete a comprehensive overview of developed approaches, published models and studies were reviewed and a survey was completed to answer how replacement problems are managed in practice at various state DOTs. This approach revealed a difference between theory and practice.

2.1 PUBLISHED MODELS AND STUDIES

This assessment focused on equipment replacement studies and research that are applicable or motivated by replacement for vehicle fleets. The main question that was addressed was how to identify replacement candidates among fleet members so that total fleet costs are minimized in the long run. Table 2.1 at the end of this section is a summary of the literature reviewed.

2.1.1 Approaches that Compute an "Economic Life"

An intuitive method for identifying replacement candidates is to define a replacement standard such as an equipment age standard. Assets that exceed the age standard are candidates for replacement. A ranking can then be implemented that sorts equipment units by how much they exceed the standard.

One of the most popular approaches to derive an age standard is the application of single asset replacement analysis to compute an "economic life," which is also known as life cycle cost analysis (LCCA). LCCA is extensively covered in the engineering economics literature. Eilon et al. (*1966*) considered acquisition cost, resale value and maintenance cost in order to derive the minimum average costs per equipment year and the corresponding optimal equipment age policy for a fleet of fork lift trucks.

Chee (*1975*) analyzed the fleet of Ontario Hydro using LCCA and generated optimal equipment age policies for different equipment classes. Because LCCA gives only one replacement criterion – namely the economic life – for a whole equipment class, Chee proposed to also consider repair costs for individual equipment units. As a result, repair cost limits are computed in addition to an economic life. If a fleet member stays within the repair cost limits for each year, it is replaced only after reaching the economic life of its class.

Weissmann et al. (*2003*) applied LCCA to individual pieces of equipment in the Texas DOT fleet. Their results indicated that this approach combined with a multi-attribute ranking is more cost efficient than utilizing a single age standard. This multi-attribute ranking considers economic life, operation costs, repair costs and usage in order to assign replacement priorities to equipment units.

Ayres and Waizeneker (*1978*) normalized annual maintenance costs by mileage and current acquisition costs, and then used this inflation-independent parameter for LCCA. The normalization is assumed to fix the problem of differences in complexity and function of equipment units. Thus, the method can make replacement decisions fleet-wide – ignoring the fact that a fleet consists of different equipment classes.

2.1.2 Approaches that Utilize a Repair Cost Limit

Another popular replacement criterion utilized was repair costs. Some literature provides evidence that repair cost limit policies have some advantages over lifetime limit policies. Drinkwater and Hastings (*1967*) analyzed data for army vehicles. They derived age dependent frequencies for repair visits per year and distributions for repair costs per visit. They used this information in a combination of dynamic programming and Monte Carlo simulation in order to determine optimal repair cost limits. They found that their repair cost limit policy leads to financial savings when compared to an LCCA-based economic age policy, and also when compared to an experience-based repair cost limit policy (which was previously applied on the army fleet).

A similar result comes from Love et al. (*1982*) who worked with fleet data from Postal Canada and compared economic age policies with repair cost limit policies. They derived economic ages analytically and repair cost limits were generated in a Markov simulation. Applied to the Postal Canada fleet, the repair cost limit policy was superior to the economic age policy.

Instead of using repair cost limits for repairs that have occurred, Hastings (*1969*) derived repair cost limits for estimates of future repair costs. He assumed that before any repair measure was conducted, fleet members were run through an inspection and repair costs were estimated. The actual repair was only undertaken if estimated costs were smaller than the derived repair cost limit.

Another approach which comes from Nakagawa and Osaki (*1974*) does not focus on repair costs, but on repair time. Their policy defined a limit for the time a broken unit of equipment spends in repair measures. The repair time limit was derived by minimizing expected costs per unit time over an infinite time span.

2.1.3 Comprehensive Cost Minimization Models

Other approaches expand the problem of optimal replacement to the problem of optimal buy, operate and sell policies. Simms et al. (*1984*) detailed data from an urban transit bus fleet. Equipment units in this fleet were operated at different levels and performed different tasks as a function of age or cumulative mileage, subject to varying capacity constraints. Moreover, newer equipment units had different acquisition and operating cost structures than older less sophisticated fleet members. By applying a combination of dynamic programming and linear optimization, an optimal buy, operate and sell policy was derived for the investigated fleet.

Similar to Simms, Hartman (*1999*) was looking for the minimum cost replacement schedule and associated utilization levels for a multi-asset case – emphasizing that utilization is a decision variable and not a parameter. The author examined the problem of simultaneous determination of asset utilization levels as well as replacement schedules, while the total costs of assets that operated in parallel were minimized. A linear program that considered dependency of operating costs on utilization levels and dependency of utilization levels on a deterministic demand solved the problem.

In later research, Hartman faced the same problem, but asset utilization levels had to meet a stochastic demand (*Hartman 2004*). In a simplified case with two equipment units and parallel operation of both assets, the author determined the optimal replacement schedules and utilization levels for both individual vehicles by applying dynamic programming. Both Simms and Hartman faced complex equipment replacement, operating and scheduling problems in vehicle fleets. They did not promote particular replacement criteria but presented optimization methodologies that led to cost efficient results for a specific fleet.

2.1.4 Models that Explicitly Consider Decreasing Utilization with Age

None of the prior work reviewed specifically considered decreasing utilization levels of assets as they age. At ODOT, equipment utilization has been decreasing with equipment age, but constant utilization has been a widely spread assumption made in the replacement models literature. Simms et al. (*1984*) derived an optimal buy, operate and sell policy for an urban transit bus fleet whose members operated at different levels depending on equipment age. They reduced the problem to two levels of utilization: young buses were operated at a constantly high level meeting the base demand, while utilization was constantly low for buses older than ten years because they were only used when needed to meet peak demand. Unlike the replacement decision at ODOT however, they assumed utilization was controllable.

Redmer (*2005*) derived the optimal lifetime limit or economic life for a freight transportation fleet, which showed decreasing utilization as equipment grew older and constant utilization levels within age classes. The basis of his model was the LCCA approach from Eilon et al. (*1966*), which assumed constant utilization, and thus, was not directly applicable to the fleet considered. Eilon et al. considered analyzed costs per unit time. Redmer concluded that Eilon's model provided lifetime limits approaching infinity when the fleet data showed decreasing utilization with age. Instead of using costs per unit time, Redmer modified Eilon's LCCA approach so that costs were given per kilometer. As a result, discounted costs of ownership per kilometer were minimized over replacement age and a feasible, cost minimizing economic life was provided.

The second study underlining the importance of decreasing utilization levels over equipment age was published by Buddhakulsomsiri and Parthanadee (*2006*). Their model was adopted from Hartman (*1999*). A major difference was that in Hartman's model, utilization was defined as a decision variable, whereas in Buddhakulsomsiri and Parthanadee's study it was assumed that utilization per age class was constant, and thus utilization was a model parameter. Their assumptions about utilization levels were identical to the assumptions made by Redmer. In addition, Buddhakulsomsiri and Parthanadee explained that decreasing utilization might follow from a dependent use pattern: “Given that the various vehicles are available to provide the same service or perform the same function, it is the newer ones that are generally preferred.” Buddhakulsomsiri and Parthanadee’s study was motivated by the situation at ODOT. Eventually, by minimizing the total costs of purchasing, selling, owning, and operating equipment units over a finite planning horizon Buddhakulsomsiri and Parthanadee provided a fleet specific and cost minimal buy, operate, and sell policy.

2.1.5 Models for Equipment Replacement in Fleets

Problems related to equipment replacement in fleets were analyzed by Khasnabis et al. (*2003*), Davenport et al. (*2005*) and Rees et al. (*1982*). Khasnabis et al. assumed that future demand for fleet services and the expected costs of replacement, rehabilitation and remanufacturing were known. The authors showed that the optimal capital allocation for the dual purpose of purchasing new equipment units and rebuilding existing ones within the constraint of a fixed budget could be obtained with linear programming. For a fleet of cutaway passenger vans Davenport et al. created a fleet condition forecast model. By using a regression model they found that the parameters equipment age, total mileage, miles per year on unpaved roads, lift equipment, and percentage of population older than age 65 were the best equipment condition predictors. Rees et al. made a replacement demand forecast by simulating the steady process of deterioration and equipment breakdown within a Markov type network.

Table 2.1: Literature review summary

Source	Purpose	Fleet	Model	Utilization
Davenport, N.S., Anderson, M.D. and Farrington, P.A. (2005): Development and application of a vehicle procurement model for rural fleet asset management	Fleet condition forecast	Cutaway passenger vans	Regression model	Considered
Khasnabis, S., Bartus, J. and Ellis, R.D. (2003): Asset management framework for state departments of transportation to meet transit fleet requirements	Find the optimal capital allocation for the dual purpose of purchasing new assets and rebuilding existing ones within the constraint of a fixed budget	Medium sized buses	Linear optimization	Not considered
Rees, L.P., Clayton, E.R. and Taylor, B.W.III (1982): Network simulation model for police patrol vehicle maintenance and replacement analysis	Replacement demand forecast	Police patrol fleet	Markov type simulation	Implicitly constant
Love, C.E., Rodger, R. and Blazenko, G. (1982): Repair limit policies for vehicle replacement	Find the optimal life time limit and find the optimal repair cost limit. Compare financial consequences.	Postal Canada fleet	LCCA: Nonlinear programming; Repair cost limit: Markov decision process	Implicitly constant
Venkatakrishnan, K.S. and Venmathi, S. (1989): Optimal replacement time of equipment via simulation for truncated failure distributions	Comparison of life time limit derivation: LCCA vs. Monte Carlo Simulation	Asset independent	LCCA: Nonlinear programming; Repair cost limit: Monte Carlo simulation	Not considered
Ayres, R.M. and Waizeneker, J. (1978): A practical approach to vehicle replacement	Find the optimal life time limit	Transportation fleet	Nonlinear programming	Implicitly constant
Chee, P.C.F. (1975): A practical vehicle replacement policy for Ontario Hydro	Find the optimal life time limit and repair cost limit	Ontario Hydro fleet	No mathematical model	Constant
Eilon, S., King, J. R. and Hutchinson D. E. (1966): A Study in Equipment Replacement	Find the optimal life time limit	Fork lift trucks	Nonlinear programming	Implicitly constant
Redmer, A. (2005): Vehicle replacement planning in freight transportation companies	Find the optimal life time limit	Freight transportation fleet	Linear programming	Decreasing as assets grow older / Constant for assets of same age
Weissmann, J., Weissmann, A.J. and Gona, S. (2003): Computerized equipment replacement methodology	Find the optimal life time limit and identify a valid multi attribute asset ranking	Texas DOT fleet	Nonlinear programming	Implicitly constant
Drinkwater, R.W. and Hastings, N.A.J. (1967): An economic replacement model	Find the optimal repair cost limit	Non armored army fleet	Dynamic programming and Monte Carlo simulation	Not considered
Hastings, N.A.J. (1969): The Repair Limit Replacement Method	Find the optimal repair cost limit	Single asset case	Dynamic programming	Not considered
Nakagawa, T. and Osaki, S. (1974): The Optimum Repair Limit Replacement Policies	Find the optimal repair time limit	Single asset case	Nonlinear programming	Not considered
Buddhakulsomsiri, J., Parthanadee, P. (2006): Parallel replacement problem for a fleet with dependent use	Find the optimal replacement schedule	Not specified real fleet	Linear programming	Decreasing as assets grow older / Constant for assets of same age
Hartman, J. C. (1999): A general procedure for incorporating asset utilization decisions into replacement analysis	Find the optimal replacement schedule and associated utilization levels	Multi asset case	Linear programming	Function of deterministic demand
Hartman, J. C. (2004): Multiple asset replacement analysis under variable utilization and stochastic demand	Find the optimal replacement schedule and associated utilization levels	Two asset case	Stochastic dynamic programming	Function of stochastic demand
Simms, B.W., Lamarre, B.G. and Jardine, A.K.K. (1984): Optimal buy, operate and sell policies for fleets of vehicles	Find the optimal replacement schedule and associated utilizations levels	Urban transit bus fleet	Dynamic and linear programming	High for young assets / Low for old assets

2.2 MODELS USED IN PRACTICE

The literature review showed that equipment replacement problems occurring in vehicle fleets are addressed in many ways. While examining published and proposed methodologies the question arose as to which policy or policies were actually used by agencies in daily business? To answer this question, a telephone survey was conducted with a selection of U.S. state DOTs. The DOTs were asked what kind of replacement methodology they were currently using. Table 2.1 shows a summary of the Survey results. In most cases replacement priorities were computed by replacement priority rankings, which used a varied selection of ranking criteria.

A conversation with the Division Chief of the Washington State Department of Transportation (WSDOT) revealed that WSDOT started with an LCCA model similar to that used by a commercial long distance fleet transportation company when they initially set up their replacement methodology. LCCA models assume that annualized maintenance and operating costs do not decrease while equipment is growing older. According to the Division Chief, however, this requirement was not fulfilled by WSDOT fleet. The experiences at WSDOT coincided with the concerns about applying LCCA to the ODOT fleet, which showed decreasing utilization over equipment age. However, it was surprising that two DOTs stated that they still planned on conducting LCCA in the near future in order to improve their vehicle replacement.

Eighty-nine percent of the DOTs acknowledged that the selection of the fixed standards used in replacement decisions originated from experience and not from data. Once these standards were determined, quantitative data driven models were used. The models used in practice generated replacement priority rankings based on a set of certain ranking criteria that differed for different state DOTs. Typically, a model would utilize a measure computed as a ratio relative to a fixed standard for some given ranking criterion. For example, if mileage were used as a ranking criterion, then a measure computed for a single asset might be the asset mileage divided by a fixed mileage standard. The result of the telephone survey indicated that the criteria mileage/hours and time in service were used by 89% of those contacted. Only 56% used repair cost limits in their models.

Table 2.2: Summary of telephone survey results (conducted between 2/12/08 and 2/26/08)

Department of Transportation	Managed Fleet	Replacement Priority Ranking Criteria						Experience based Policy	Plan to apply LCCA in near Future
		Mileage/Hours	Time in Service	Operating Cost	Repair Cost	Acquisition Cost	Physical Assessment		
Alabama DOT	All equipment	x	x					x	
California DOT	All equipment	x	x		x				
Florida DOT	All equipment	x	x				x	x	x
Illinois DOT	Heavy trucks	x			x		x	x	
Michigan DOT	Heavy trucks		x				x	x	
Oregon DOT	All equipment	x	x	x	x	x		x	
Texas DOT	All equipment	x	x		x	x		x	
Virginia DOT	All equipment	x	x		x			x	x
Washington State DOT	All equipment	x	x					x	
Portion of DOTs using criteria		**89%**	**89%**	**11%**	**56%**	**22%**	**33%**	**89%**	**22%**

2.3 CONCLUSIONS

Both a review of published literature, and a telephone survey among state DOTs were conducted to assess the state-of-the-art in equipment replacement modeling. Recommended methods in literature relied heavily on assumptions made about the fleet and on conditions under which the fleet was operated. Two studies addressed explicitly a case of decreasing utilization over equipment age, as seen with many assets in state DOTs. The authors pointed out that the assumptions of the standard economic life model did not hold for a fleet with decreasing utilization.

The telephone survey revealed that state DOTs were familiar with age dependent use patterns and that they managed replacement decisions with simple models. In these models equipment units were ranked based on measures computed relative to standards for various criteria. The standards and criteria that were found to be a part of most ranking systems currently in use by state DOTs were not quantitatively justified, but instead were experience-based. For fleets showing decreasing utilization over equipment age, it was not possible to identify a dominant replacement strategy.

The literature review and telephone survey showed that there is a discrepancy between theory and practice. While published literature offers a multitude of methodologies to derive cost efficient replacement decisions – LCCA being the most prominent amongst them – a selection of nine U.S. DOTs did not apply any of these methodologies. Instead, in practice, simple asset rankings based on different criteria incorporating fixed standards which are mostly experience-based were utilized. No justifications for the methods used were known or available.

The results also showed that there was an opportunity in this project to apply quantitative engineering approaches to how ODOT determines its equipment replacement priorities.

3.0 EVALUATION OF MODELS USED IN THE FLEET CONDITION TOOL

In this section the testing and evaluation of simple replacement models that utilize only life-to-date equipment cost and usage data are presented. One of the models tested is the current model being used to prioritize equipment replacement across multiple equipment classes as found in the Oregon Department of Transportation (ODOT) fleet. The emphasis on models that utilize only life-to-date equipment cost and usage data were driven by data available in most equipment management and maintenance systems information systems in use by state DOTs.

The approach used in this research to evaluate different replacement models was simulation. Part of this procedure used probability distributions, which were fit to historical ODOT data. Prior annual usage and cost data compiled for ODOTs fleet was analyzed. Probability distributions representing equipment usage and costs over time in different equipment classes were fitted to the data. Correlations between usage and costs, and correlation over time for usage and costs were computed and represented in the simulations.

In order to reflect ODOT fleet management characteristics, the size of a simulated fleet remained constant over time, and the replacement of a fixed and limited amount of equipment occurred every two years using different replacement models to identify equipment to replace. Single simulation runs were 500 years in length, and the performance measure utilized was the average total cost per usage unit to operate the fleet. The centerpiece of the simulation was the ability to apply different replacement priority ranking methods to the same fleet of equipment under identical circumstances. This made the recorded performance measures comparable and allowed an assessment of the different replacement criteria.

The simulations modeled a fleet of equipment over time where a fixed number of assets from the fleet were replaced at regular intervals. Ten different simple replacement models or replacement priority ranking criteria were utilized in the simulations to determine which equipment had the highest priority for replacement. The total costs of operating the fleet over the simulated time, and total usage provided by the fleet was measured.

This approach required the analysis of historical usage and cost data to determine appropriate distributions and correlation values to represent equipment in the fleet.

3.1 DATA PREPARATION

ODOT Fleet Services provides management of a fleet, which consists of a variety of small and large trucks, cars, and heavy machinery such as graders, bulldozers, and many types of tractors. To help manage this variety of equipment, ODOT's fleet is separated into equipment classes. In preparation for the simulation historical data from a selection of equipment classes were analyzed. Costs over time were adjusted for inflation.

3.1.1 Organization of Historical Fleet Data in Matrix Format

The historical data analyzed included equipment acquisition cost, annual usage (mileage for the classes examined), repair costs, fixed costs, and operating costs for each piece of equipment in the fleet. The data were from the time period of July 1994 to June 2002 with a small amount of equipment having some older data available. The equipment classes analyzed are shown in Table 3.1. These equipment classes were selected because they represented a variety of larger and more expensive equipment classes, and were also the five largest equipment classes in the fleet.

For each analyzed equipment class, records were organized in such a way that mileage, repair costs, fixed costs, and operating costs were separated from each other into subsets. Within each subset, the data were sorted by equipment age.

Table 3.2 presents an example data matrix for an arbitrary equipment class. Each row in this matrix contains data for one asset, and each column in the matrix is labeled with a combination of data type and equipment age.

The five equipment classes represented in the simulations and shown in Table 3.1 were selected because they represented a large portion of the more costly assets in the ODOT fleet and were also large equipment classes. Having more individual pieces of equipment in the class meant that more data were available to analyze, giving a better representation of the equipment in the simulation.

The largest equipment class was Truck HVY DSL with around 215 individual assets. The smallest equipment class analyzed was the Sedan class with around 115 individual assets. However, even for the largest equipment classes, the number of available records for a given data item at a specific equipment age decreased each year. The decrease started at different ages for different equipment classes. For Truck HVY DSL there were at least 26 records for each age of 19 years and younger but only four or less records for trucks of age 20 and older. For Sedans the amount of records decreased to 23 through age 11, and from age 12 and older there were less than nine records per age available.

The number of available records per age depended on the length of class specific service life as well as on the total size of an equipment class. A larger equipment class showed more records per age than a smaller one. An equipment class with characteristically long service life had more records per age, particularly more records at higher ages, than an equipment class with generally short service life. This explains why class Truck HVY DSL provided more data through the age of 19 when compared to class Sedan.

Table 3.1: Equipment classes represented in the simulation

Equipment Class	Description
Sedan	Sedans
Pickup 3/4T 4X2	¾ ton pickups with two wheels rear drive
Truck LT 4X2	Light trucks with two wheels rear drive
Truck Med4X2 DSL	Medium trucks with two wheels rear drive and diesel engine
Truck HVY DSL	Heavy diesel trucks

Table 3.2: Example data matrix showing the organization of historical fleet data

	M_01	M_02	...	M_12	M_13	...	M_24	M_25	RC_01	RC_02	...	RC_12	RC_13	...	RC_24	RC_25	FC_01	FC_02	...	FC_12	FC_13	...	FC_24	FC_25	OC_01	OC_02	...	OC_12	OC_13	...	OC_24	OC_25	AC
EQ_A	R	R	...	R	-	...	-	-	R	R	...	R	-	...	-	-	R	R	...	R	-	...	-	-	R	R	...	R	-	...	-	-	R
EQ_B	R	R	...	R	-	...	-	-	R	R	...	R	-	...	-	-	R	R	...	R	-	...	-	-	R	R	...	R	-	...	-	-	R
EQ_C	-	R	...	R	R	...	-	-	-	R	...	R	R	...	-	-	-	R	...	R	R	...	-	-	-	R	...	R	R	...	-	-	R
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
EQ_X	-	-	...	R	R	...	-	-	-	-	...	R	R	...	-	-	-	-	...	R	R	...	-	-	-	-	...	R	R	...	-	-	R
EQ_Y	-	-	...	-	R	...	R	-	-	-	...	-	R	...	R	-	-	-	...	-	R	...	R	-	-	-	...	-	R	...	R	-	R
EQ_Z	-	-	...	-	R	...	R	-	-	-	...	-	R	...	R	-	-	-	...	-	R	...	R	-	-	-	...	-	R	...	R	-	R

Key
R = Recorded parameter value
EQ_x = Equipment unit x
M_x = Mileage travelled by equipment aged x
RC_x, FC_x, OC_x = Repair Cost, Fixed Cost, Operating Cost for equipment aged x
AC = Acquisition Cost)

3.1.2 Adjustment for Inflation

A data matrix generated from data provided by ODOT contained historical acquisition cost and annual mileage, repair cost, fixed cost, and operating cost data. Annual cost and mileage parameters were for individual equipment units for July 1, 1994 through June 30, 2002. To make costs comparable they were adjusted for inflation. All costs were adjusted to the price level prevalent during the ODOT accounting year July 1, 2001 to June 30, 2002 (2001/2002). Adjustments were made with four different consumer price indexes (CPI), which are published by the U.S. Department of Labor (see Table 3.3).

Table 3.3: Consumer price indexes used to adjust cost data

Cost Type	Used CPI
Repair cost	U.S. city average; Motor vehicle maintenance and repair; NSA; 1947-2008
Fix cost	U.S. city average Motor vehicle insurance; NSA; 1947-2008
Operating cost	U.S. city average; Motor fuel; NSA; 1935-2008
Acquisition Cost	U.S. city average; New vehicles; NSA, 1947-2008

NSA = Not seasonally adjusted

All four CPI's were given as monthly data. The arithmetic mean of twelve monthly CPIs (July through June) was used as the CPI for a corresponding ODOT accounting year. Price inflators for each year were computed by dividing the 2001/2002 CPI by CPIs from 1994/1995 through 2001/2002. The results are presented in Table 3.4. To adjust costs, values were multiplied by the adjustment factor for the accounting year when the data were recorded.

Table 3.4: Price adjustment factors used to adjust costs to the price level of ODOT accounting year 2001/2002

	CPI: U.S. city average; Motor vehicle maintenance and repair; NSA	**CPI: U.S. city average; Motor vehicle insurance; NSA**	**CPI: U.S. city average; Motor fuel; NSA**	**CPI: U.S. city average; New vehicles; NSA**
Accounting Year	**Price Inflator for Repair Cost**	**Price Inflator for Fix Cost**	**Price Inflator for Operating Cost**	**Price Inflator for Acquisition Cost**
2001/2002	1	1	1	1
2000/2001	1.03678	1.069995	0.854661	0.989826
1999/2000	1.072511	1.094338	0.96597	0.988035
1998/1999	1.102496	1.1002	1.240441	0.985217
1997/1998	1.135182	1.098071	1.137312	0.981789
1996/1997	1.163314	1.123913	1.060724	0.976521
1995/1996	1.198291	1.166887	1.110251	0.991739
1994/1995	1.229412	1.217249	1.119372	1.010446
…	…	…	…	…

3.2 DATA ANALYSIS

3.2.1 Correlation Analysis

Analysis was conducted to determine if significant correlation existed between mileage, repair costs, fixed costs, operating costs, and acquisition costs. Sample correlations were computed using Spearman's rank correlation. Analysis was conducted using MATLAB software. MATLAB offers a function that returns a p-by-p matrix that contains pair-wise Spearman rank correlation coefficients between each pair of columns in the n-by-p input data matrix (Table 3.2). When MATLAB computes the rank correlation coefficient between the two columns in Table 3.2, the computation only considers rows that have no missing values in both of the compared columns.

The outcome of the correlation coefficient computation is shown in Table 3.5. Table 3.5 shows an upper triangular 101-by-101 matrix containing the pair-wise Spearman's rank correlation coefficient between each pair of columns in the input matrix (Table 3.2). For example, coefficient $\rho(M\_12, M\_13)$ in Table 3.5 represents the value of Spearman's rank correlation coefficient between mileage of equipment of age 12 and mileage for equipment of age 13. The diagonal of the triangular matrix is filled with ones because each column in Table 3.2 is perfectly correlated with itself.

Table 3.5 is referred to as the *full* correlation matrix because this matrix contains all correlation coefficients computed by MATLAB. A *reduced* correlation matrix (see Table 3.6) was created and used in the simulation. To generate the reduced correlation matrix the following steps were executed:

1. Set cells which have no content or are filled with “-” to zero.
2. Many elements in Table 3.5 were eliminated to simplify random variate generation.
    a. Correlation between columns of the same data type was restricted to a time lag of one. Example: $\rho(M\_12, M\_13)$
    b. Correlation between columns of different data types was included only for the same equipment ages. Example: $\rho(M\_12, RC\_12)$
    c. Correlation between acquisition cost and any other parameter were included. Example: $\rho(M\_12, AC)$
3. Correlations of $\rho \geq 0.5$ were included in the simulation and other values were set to zero.

Table 3.5: Upper triangular p-by-p matrix containing pair-wise Spearman rank correlation coefficients (ρ) between each pair of columns in Table (referred to as the "full correlation matrix").

	M_01	M_02	…	M_12	M_13	…	M_24	M_25	RC_01	RC_02	…	RC_12	RC_13	…	RC_24	RC_25	FC_01	FC_02	…	FC_12	FC_13	…	FC_24	FC_25	OC_01	OC_02	…	OC_12	OC_13	…	OC_24	OC_25	AC
M_01	1	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
M_02		1	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
…			1	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
M_12				1	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
M_13					1	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
…						1	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
M_24							1	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
M_25								1	-	-	…	-	-	…	-	-	-	-	…	-	-	…	-	-	-	-	…	-	-	…	-	-	-
RC_01									1	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
RC_02										1	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
...											1	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
RC_12												1	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
RC_13													1	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
…														1	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
RC_24															1	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
RC_25																1	-	-	…	-	-	…	-	-	-	-	…	-	-	…	-	-	-
FC_01																	1	ρ	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
FC_02																		1	…	ρ	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
…																			1	…	…	…	…	…	…	…	…	…	…	…	…	…	…
FC_12																				1	ρ	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
FC_13																					1	…	ρ	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
…																						1	…	…	…	…	…	…	…	…	…	…	…
FC_24																							1	-	ρ	ρ	…	ρ	ρ	…	ρ	-	ρ
FC_25																								1	-	-	…	-	-	…	-	-	-
OC_01																									1	ρ	…	ρ	ρ	…	ρ	-	ρ
OC_02																										1	…	ρ	ρ	…	ρ	-	ρ
…																											1	…	…	…	…	…	…
OC_12																												1	ρ	…	ρ	-	ρ
OC_13																													1	…	ρ	-	ρ
…																														1	…	…	…
OC_24																															1	-	ρ
OC_25																																1	-
AC																																	1

These steps, applied to the full correlation matrix generated by MATLAB, yielded a reduced correlation matrix (Table 3.6). Details of particular steps are discussed below.

Step 1: The cells with no content were set to zero (shown as empty cells).

Step 2: Reduction of a correlation matrix to diagonals eliminates redundant information. To support this statement, two sets of random variates were generated, one based on a full correlation matrix and one based on the corresponding reduced correlation matrix. Correlation analysis of the generated sets of random variates yielded roughly identical full correlation matrixes for both sets of random variates.

Step 3: The criterion used to consider data as uncorrelated is that $\rho \leq 0.5$. Engineering judgment has shown that low correlations like $\rho \leq 0.5$ have no major impact on the outcome of random variate generation.

Table 3.6: Reduced correlation matrix containing selected rank correlation coefficients from Table 3.5.

	M_01	**M_02**	**…**	**M_12**	**M_13**	**…**	**M_24**	**M_25**	**RC_01**	**RC_02**	**…**	**RC_12**	**RC_13**	**…**	**RC_24**	**RC_25**	**FC_01**	**FC_02**	**…**	**FC_12**	**FC_13**	**…**	**FC_24**	**FC_25**	**OC_01**	**OC_02**	**…**	**OC_12**	**OC_13**	**…**	**OC_24**	**OC_25**	**AC**
M_01	1	ρ							ρ																ρ								ρ
M_02		1	…							ρ																ρ							ρ
…			1	…							…																…						…
M_12				1	ρ							ρ																ρ					ρ
M_13					1	…							ρ																ρ				ρ
…						1	…							…																…			…
M_24							1	ρ							ρ																ρ		ρ
M_25								1								ρ																ρ	ρ
RC_01									1	ρ															ρ								ρ
RC_02										1	…															ρ							ρ
...											1	…															…						…
RC_12												1	ρ															ρ					ρ
RC_13													1	…															ρ				ρ
…														1	…															…			…
RC_24															1	ρ															ρ		ρ
RC_25																1																ρ	ρ
FC_01																	1																
FC_02																		1															
…																			1														
FC_12																				1													
FC_13																					1												
…																						1											
FC_24																							1										
FC_25																								1									
OC_01																									1	ρ							ρ
OC_02																										1	…						ρ
…																											1	…					…
OC_12																												1	ρ				ρ
OC_13																													1	…			ρ
…																														1	…		…
OC_24																															1	ρ	ρ
OC_25																																1	ρ
AC																																	1

3.2.2 Distribution Fitting

The last step of data preparation was to find the best fit probability distributions for each considered data item for each equipment age (e.g., mileage during year five). Distribution fitting was done with Crystal Ball (version 7.3) software. The goodness of fit test used was the Kolmogorov-Smirnov test (K-S test), which ranks fitted distributions based on the largest vertical distance between an empirical distribution function of the sample, and the cumulative distribution function of a hypothesized distribution. Crystal Ball conducts K-S tests and identifies the best fit from 14 continuous distributions.

Historical fleet data were organized in an n-by-p data matrix (see Table 3.2) such that each row represented a unit of equipment and each column represented one of four types of data at a specific equipment age. Each column of this matrix formed the input for the process of distribution fitting. Values in each column represented the raw data, and for each column a continuous statistical distribution was fit to these data. Table 3.7 shows the form of the results of the probability distribution fitting.

Table 3.7: Some results of probability distribution fitting to historical fleet data shown in Table 3.2

Sample:	**M_01**	**M_02**	**…**	**M_12**	**M_13**	**…**	**M_24**	**M_25**	**RC_01**	**RC_02**	**…**	**RC_12**	**RC_13**	**…**	**RC_24**	**RC_25**	**FC_01**	**FC_02**	**…**	**FC_12**	**FC_13**	**…**	**FC_24**	**FC_25**	**OC_01**	**OC_02**	**…**	**OC_12**	**OC_13**	**…**	**OC_24**	**OC_25**	**AC**
Best fit:	BetaPERT	Gamma	…	Normal	Beta	…	Beta	Lognormal	Beta	Gamma	…	Weibull	Weibull	…	Lognormal	Lognormal	Student's t	Logistic	…	BetaPERT	Logistic	…	Beta	Lognormal	Beta	Max Extreme	…	Beta	Lognormal	…	Beta	Lognormal	
K-S statistic of best fit:	0.0483	0.0445	…	0.0516		0.0616	0.0708	…	0.0679		0.1399	0.2111	…	0.2289		0.2537	0.1705	…	0.2037		0.0483	0.0445	…	0.0516		0.0616	0.0708	…	0.0679		0.1399	0.2111	

For many ages of equipment, data were not available for fitting. This was particularly true for an equipment class like sedans, where the average service life was small compared to heavy diesel trucks. A forecast for the mean and standard deviation of a data item for years where data were not available was made by continuing a plot of existing means and standard deviations smoothly by hand. Figure 3.1 shows such forecasts (indicated by arrows) for year 25 mileage when data were available for years 1 through 24. The forecasted values for the mean and standard deviation were used as parameters for a lognormal distribution.

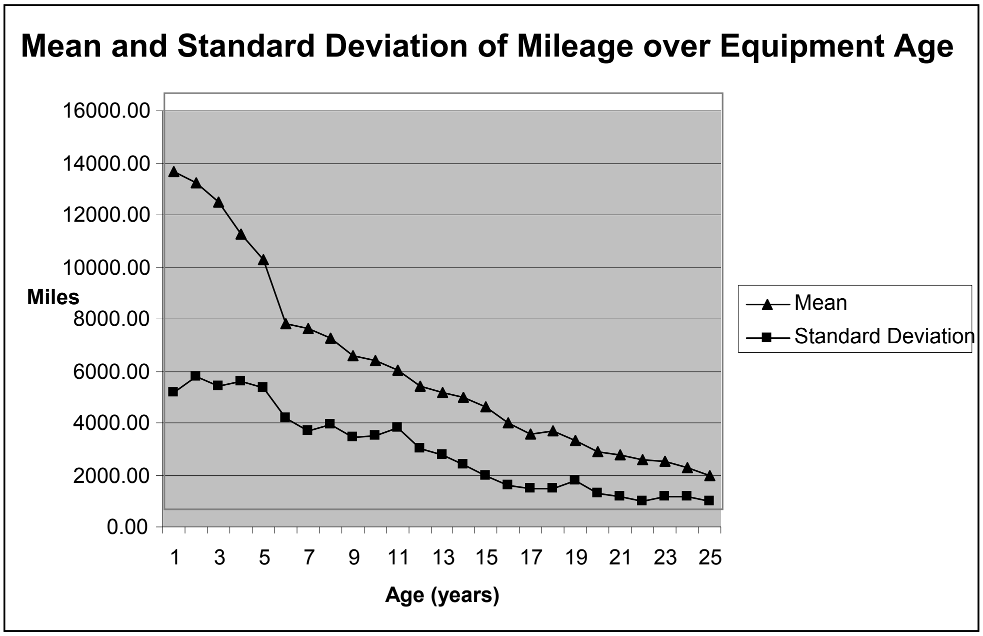


Figure 3.1: Mean and standard deviation of mileage over equipment age and a hand drawn forecast for age 25 years.

Although this forecasting method seemed unconventional, it generated more "common sense" results than forecasting methods like single and double moving average or exponential smoothing, which were all tried. For example, the Sedan equipment class had historical data only for equipment ages one through eleven. In this case, a 14-year forecast had to be made for the mean and standard deviation of mileage cost data. Over such a long period, moving average and exponential smoothing forecasts produced unreasonable forecasts. Single moving average and single exponential smoothing forecasts resulted in a constant value which did not reflect negative trends. Double moving average and double exponential smoothing methods were likely to forecast values below zero over higher equipment ages. Existing historical data showed decreasing trends with age. Thus, double moving average and double exponential smoothing forecasts tended to continue the negative trend without turning convex at higher equipment ages to avoid negative values.

3.3 SIMULATION

The simulation was coded in Visual Basic. The objective was to simulate the costs produced by a single equipment class fleet over a long period of time. The results from utilizing different replacement models to prioritize equipment to replace were then compared.

3.3.1 Generation of Equipment Data

One major component of the simulation was the data for individual pieces of equipment for each year that item was present in the fleet. This was accomplished by generating data for 1,000 pieces of equipment over a 25-year life and storing this information in an equipment table, as shown in Table 3.8. For a single piece of equipment this required the generation of 101 random variates from the 101 distributions presented in Table 3.7, and the correlations in Table 3.6. A row in Table 3.8 contains annual mileage, annual inflation adjusted repair costs, fixed costs, and operating costs as well as an inflation adjusted acquisition cost for a single piece of equipment.

Table 3.8: Data generated from fitted probability distribution and computed correlations (G = generated data)

	M_01	**M_02**	**…**	**M_12**	**M_13**	**…**	**M_24**	**M_25**	**RC_01**	**RC_02**	**…**	**RC_12**	**RC_13**	**…**	**RC_24**	**RC_25**	**FC_01**	**FC_02**	**…**	**FC_12**	**FC_13**	**…**	**FC_24**	**FC_25**	**OC_01**	**OC_02**	**…**	**OC_12**	**OC_13**	**…**	**OC_24**	**OC_25**	**AC**
EQ_000	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G
EQ_002	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
EQ_998	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G
EQ_999	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G

Crystal Ball v7.3 was used to generate the 1000 rows of Table 3.8. To verify that the generation of random variates was successful, sample means and sample standard deviations were compared to the means and standard deviations of the fitted distributions. No unusual or unexpected results occurred.

3.3.2 Simulation of Equipment Replacement

The 1000 pieces of equipment (vehicles) with sample data generated (Table 3.8) was used as a supply of equipment for the simulation. When a vehicle was replaced, it was replaced with a randomly selected piece of equipment from Table 3.8.

The simulation was conceptually organized as a three-dimensional matrix. The first dimension of this matrix was time (years). The second dimension was for individual pieces of equipment. The third dimension of the simulation matrix was data type (e.g., repair costs). As each year was incremented, the appropriate equipment data were added to the matrix.

The simulation started in year zero. An initial selection of equipment units was chosen randomly from Table 3.8. The data for mileage, repair costs, fix costs, operating costs, and acquisition cost were copied into the matrix. Equipment replacement was accomplished every other year where a fixed number of equipment units were replaced with new equipment from Table 3.8.

At the beginning of each simulation run, a warm up period of 300 years was run before recording performance measures. Performance was then recorded for a simulated time of 200 years. Although most equipment replaced was replaced at an age less than 25 years, if an equipment unit reached the age of 25, the simulation algorithm assigned a very high replacement priority to it, so that it was replaced first. It was assumed that equipment was no longer functional after 25 years.

For a given replacement method, five simulation replications were conducted. The random number generator used five different seeds for these five replicates. The seeds were chosen arbitrarily to be equal to the replication number. This pattern was applied without exception on every tested simulation configuration.

3.3.3 Replacement Methods

The centerpiece of this simulation approach was the ability to test different replacement methods (Table 3.9) in order to identify their impact on different performance measures – particularly cost-related performance measures. The selection of replacement methods was based on the literature review, telephone survey, and brainstorming. Each of the tested replacement methods provided a unique rule to rank active members of the simulated fleet. The top-ranked equipment unit had the highest priority to be replaced and the lowest-ranked equipment unit had the lowest replacement priority. The rules were used to compute rankings from data stored in the simulation matrix. Each time replacement occurred the candidates with the highest ranking according to replacement criteria used were replaced first.

Replacement methods 7 and 8 (shown in Table 3.9) required age standards and used standards for the specific equipment class. For the five equipment classes analyzed in this study, the appropriate age and use standards were taken from Table 3.10. The values in Table 3.10 were in use by ODOT Fleet Services. Denominators in replacement methods 7 and 8 were introduced for normalization. They transformed values given in years, miles, or monetary units to dimensionless ratios. Hence, normalization facilitated creating ranking methods that include multiple criteria. Specific values for age and use standards reflected the age and accumulated mileage expected to be provided by an average equipment unit.

3.3.4 Experimental Design

The simulation method described was used to conduct a designed experiment to test the effectiveness of different replacement criteria. The response in the experiments used was the cost per mile over 200 simulated years. Cost per mile was used as a performance measure to normalize for differences in total mileage generated in different simulation replications, since costs are very highly correlated with total mileage. In addition to replacement method and equipment class, three additional factors were included in the experimental design, as shown in Table 3.11. Replacement budget was included to test for differences in the replacement methods as more or less vehicles per replacement period could be replaced. Correlation refers to the inclusion or exclusion of computed correlations in the simulation. No effect of correlation would imply that future simulations could be executed without including this factor, which adds a significant amount of data analysis to the procedure. Fleet size was included to test for differences in the replacement methods for different size equipment classes.

Table 3.9: Models and ranking criteria evaluated

No.	Replacement Priority Ranking Method	Explanation
1	Random selection	Randomly select equipment for replacement.
2	Replace oldest first	Equipment units are ranked based on the time they have spent as active fleet members.
3	Replace highest total mileage first	Active equipment units are ranked based on the accumulated mileage they have traveled during the time they have spent as active fleet members.
4	Replace highest life total cost first	Active equipment units are ranked based on the accumulated total costs they have produced during the time they have spent as active fleet members.
5	Replace highest repair cost delta first	Active equipment units are ranked based on the highest positive difference between repair cost produced during the current year and the preceding year.
6	Replace highest total cost delta first	Active equipment units are ranked based on the highest positive difference between total cost produced during the current year and the preceding year of simulation.
7	Replace highest value of $\left(\frac{age}{age\ std}+\frac{life\ usage}{use\ std}\right)$ first	Active equipment units are ranked based on the value of the presented equation. The denominators age standard and use standard depend on the equipment class (see Table 3.10).
8	Replace highest value of $\left(\frac{age}{age\ std}+\frac{life\ usage}{use\ std}+\frac{life\ total\ cost}{acquisition\ cost}\right)$ first	Active equipment units are ranked based on the presented equation. Currently, ODOT Fleet Services uses this model. For age and use standards refer to Table 3.10.
9	Replace highest total life repair + operating cost first	Active equipment units are ranked based on the accumulated total repair and operating costs they have produced during the time they have spent as active fleet members.
10	Replace highest total life repair + operating cost per mile first	Active equipment units are ranked based on the accumulated total repair and operating costs per mile of usage they have produced during the time they have spent as active fleet members.

Table 3.10: Current ODOT age standards and use standards.

Equipment Class	Age Standard (Years)	Use Standard (Miles)
Sedan	8	100000
Pickup 3/4T 4X2	8	125000
Truck LT 4X2	8	125000
Truck Med 4X2 DSL	12	250000
Truck HVY DSL	15	300000

Table 3.11: Experimental design summary

Factor	Equipment Class	Replacement Budget	Correlation	Fleet Size	Replacement Method
Levels	Sedan	8 year avg. age for the fleet (larger budget)	Reduced correlation	25	Method 1
	Pickup 3/4T 4X2	18 year avg. age for the fleet (smaller budget)	No correlation	125	Method 2
	Truck LT 4X2				Method 3
	Truck Med 4X2 DSL				Method 4
	Truck HVY DSL				Method 5
					Method 6
					Method 7
					Method 8
					Method 9
					Method 10
Number of Levels	5	2	2	2	10
Total Number of Treatment Combinations	$5*2*2*2*10=400$				
Total Number of Simulation Runs	$400*5=2000$				

3.4 RESULTS

For a single equipment class, the experimental design described was a 2^3 x 10 factorial experiment with five replications per treatment combination. The procedure for analyzing the simulation results for an equipment class consisted of the following steps.

1. Conduct an ANOVA on the experimental results as a full factorial experiment. Produce various interaction charts.
2. Identify if the factor correlation is significant (this was significant for all equipment classes).
3. Conduct an analysis of residuals through and examination of various residual plots and normal probability plots.
4. Eliminate insignificant factors and only consider data with correlations included (since this represents reality). This resulted in a 2x10 factorial experiment with 10 replications per treatment combination (fleet size was an insignificant factor for all classes).
5. Conduct an ANOVA on this reduced experiment.
6. Generate interaction plots.
7. Conduct residual analysis.
8. Perform multiple range tests to determine differences in replacement methods.

These steps are shown below for Heavy Diesel trucks (as an example) and were applied to all equipment classes studied.

3.4.1 2^3 x 10 Factorial Design of Experiment – Heavy Diesel Trucks

Step 1 is to conduct an ANOVA as shown in Table 3.12. This is a standard ANOVA table with p-values produced that allow a determination of significance at any level. The p-value is the probability of observing the value of the test statistic assuming the null hypothesis is true. A p-value greater than 0.05 would indicate that an effect is not significant at a 0.05 level of type I error.

The results in the ANOVA table indicate that the replacement method, correlation, and replacement age (budget), have significant effects but the fleet size does not. All two-factor and three-factor interactions with these three factors are also significant. Samples of the interaction plots are shown in Figures 3.2 – 3.4. Significant interactions indicate that the effect of a particular factor is different at varying levels of another factor (or factors).

Validation of normality assumptions and constant variance assumptions in the ANOVA was conducted by generating various residual plots as shown in Figures 3.5 – 3.8. Residual plots are commonly used in practice to assess the validity of normality assumptions. A residual is computed as the difference between on observed value at a particular treatment combination, and the predicted response for that particular treatment combination. The predicted response is determined by the statistical model assumed for the ANOVA and the estimates of its parameters computed from the data.

Normal probability plots of residuals should show a clear linear pattern. Plots of residuals versus different factor levels should show an equal spread for each factor level. A plot of residuals versus fitted values should show a plot with no observable pattern.

Table 3.12: Truck HVY DSL: Analysis of variance for total cost per mile; 2^3 x 10 factorial design of experiment (all F-ratios are based on the residual mean square error)

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Main Effects					
A:Repl Method	0.0824148	9	0.0091572	517.40	0.00
B:Correlation	0.000104176	1	0.000104176	5.89	0.0158
C:Repl Age	0.908421	1	0.908421	51327.69	0.00
D:Fleet Size	0.00000138518	1	0.00000138518	0.08	0.7798
Interactions					
AB	0.0156668	9	0.00174075	98.36	0.00
AC	0.0419223	9	0.00465803	263.19	0.00
AD	0.000024578	9	0.00000273089	0.15	0.9978
BC	0.00198621	1	0.00198621	112.22	0.00
BD	0.000000367279	1	0.000000367279	0.02	0.8855
CD	0.000471156	1	0.000471156	26.62	0.00
ABC	0.00134365	9	0.000149295	8.44	0.00
ABD	0.0000367455	9	0.00000408284	0.23	0.9899
ACD	0.0000774167	9	0.00000860186	0.49	0.8838
BCD	0.00000985644	1	0.00000985644	0.56	0.4561
ABCD	0.0000507217	9	0.00000563575	0.32	0.9687
Residual	0.00566351	320	0.0000176985		
Total (Corrected)	1.05819	399			

3.4.1.1 Interaction plots

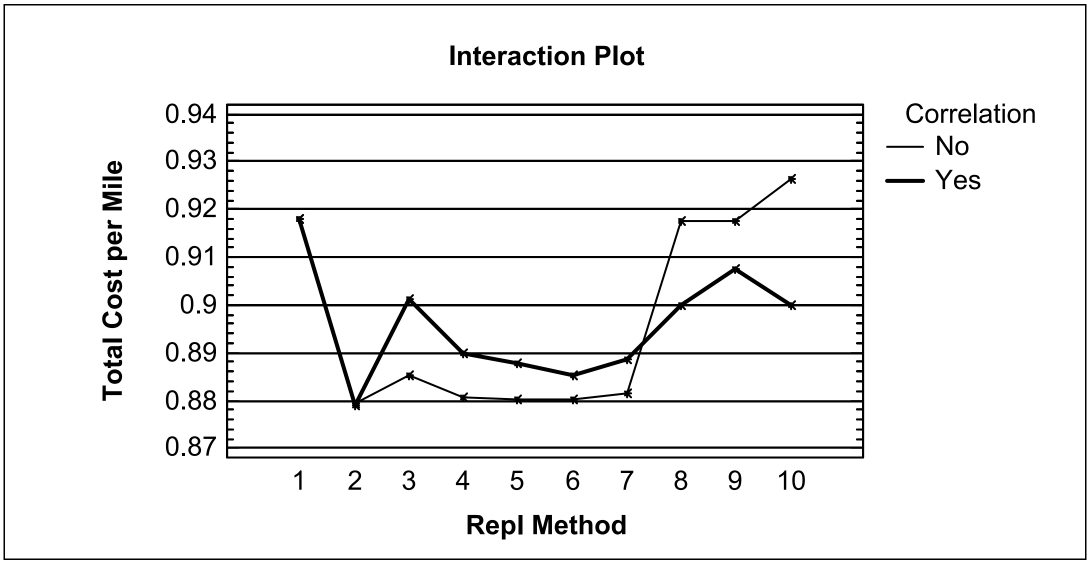


Figure 3.2: Correlation/Replacement method interaction plot showing significant interaction. Method numbers are on the x-axis and are referenced in Table 3.9

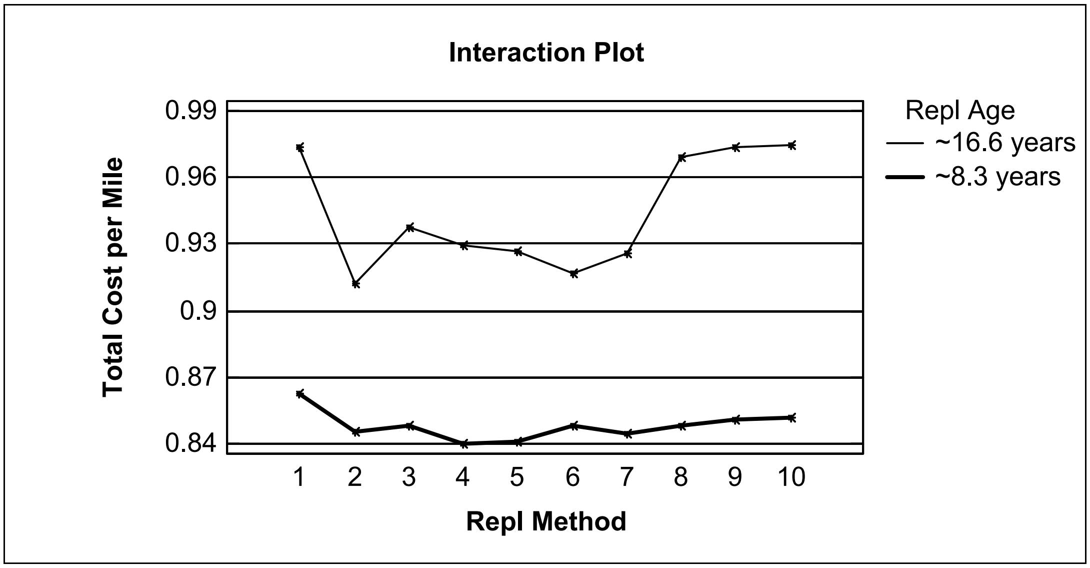


Figure 3.3: Replacement age (budget)/Replacement method interaction plot showing significant interaction.

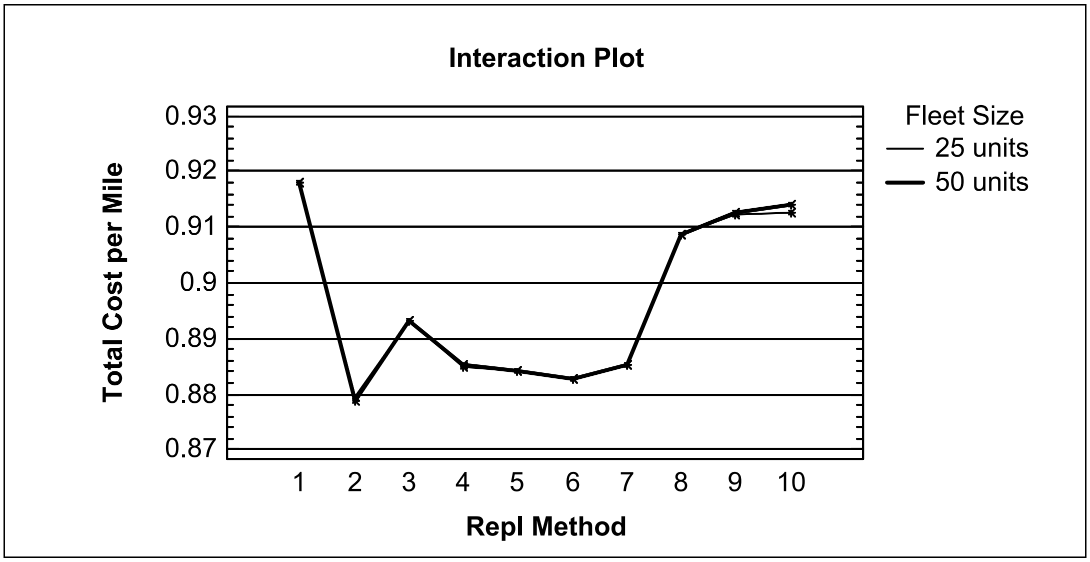


Figure 3.4: Fleet size/Replacement method interaction plot showing no significant interaction.

3.4.1.2 Model adequacy checking

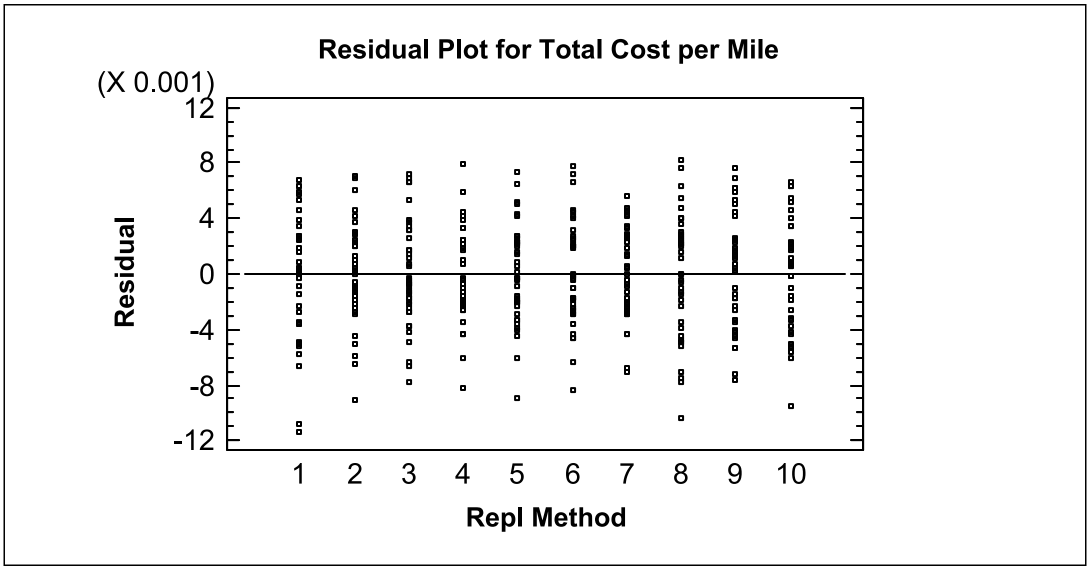


Figure 3.5: Residuals vs. replacement method plot showing no evidence of non-constant variance for different replacement methods.

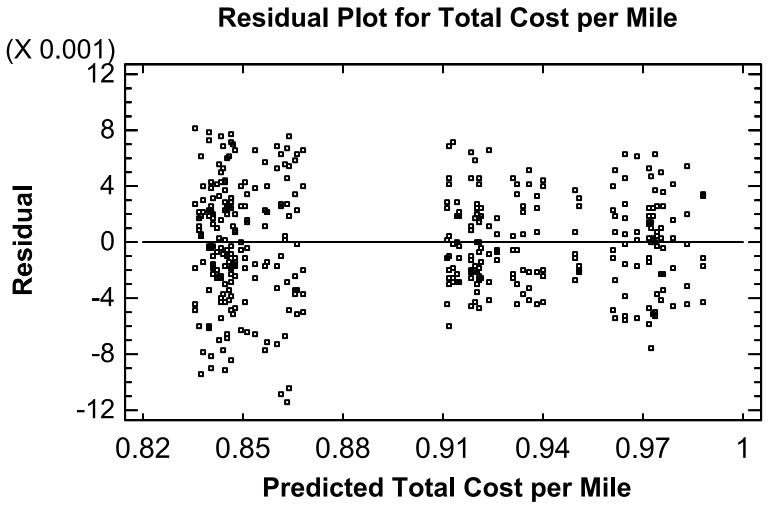


Figure 3.6: Residuals vs. fitted value plot showing no evidence of non-constant variance for different predicted values.

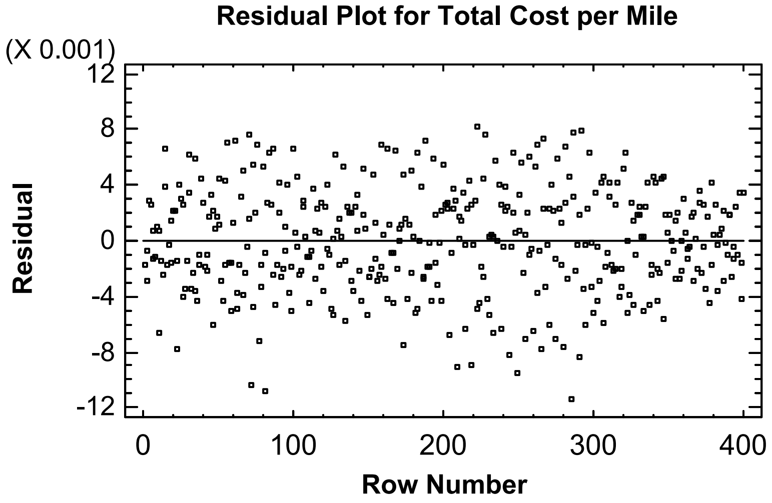


Figure 3.7: Residuals vs. experiment order plot showing no evidence of non-constant variance over time.

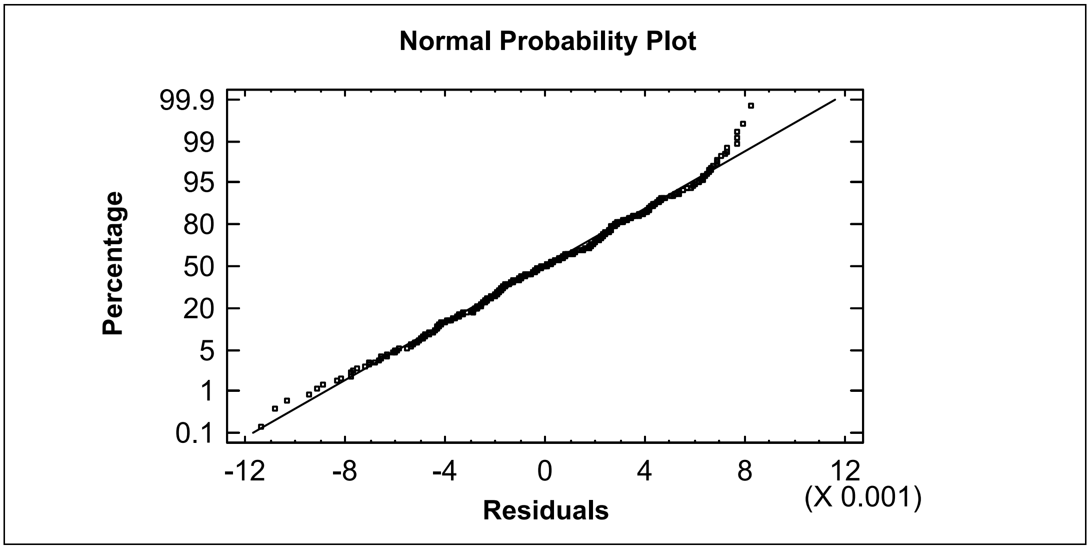


Figure 3.8: Normal probability plot of residuals showing evidence of a normally distributed error term.

The next step in the analysis was to remove non-significant factors and repeat the analysis. Since correlation among the cost data was a significant factor, only the results utilizing correlated data were subsequently used.

3.4.2 2^1 x 10 Factorial Design of Experiment – Heavy Diesel Trucks

In this analysis the following steps were executed with the experimental results.

- Rows with NO correlation deleted
- Fleet size factor ignored (merged)

The ANOVA table (Table 3.13) indicates both the replacement method and replacement age (budget) are significant at a 95% confidence level as is their interaction (all have p-values less than 0.05). The single interaction plot and residual plots are shown in Figures 3.9 – 3.11.

Table 3.13: Truck HVY DSL: Analysis of variance for total cost per mile – 2^1 x 10 factorial design of experiment (Type III sums of squares; all F-ratios are based on the residual mean square error.)

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Main Effects					
A:Repl Method	0.0250037	9	0.00277819	181.84	0.00
B:Repl Age	0.497681	1	0.497681	32575.23	0.00
Interaction					
AB	0.021947	9	0.00243856	159.61	0.00
Residual	0.00275002	180	0.0000152779		
Total (Corrected)	0.547381	199			

3.4.2.1 Interaction plots

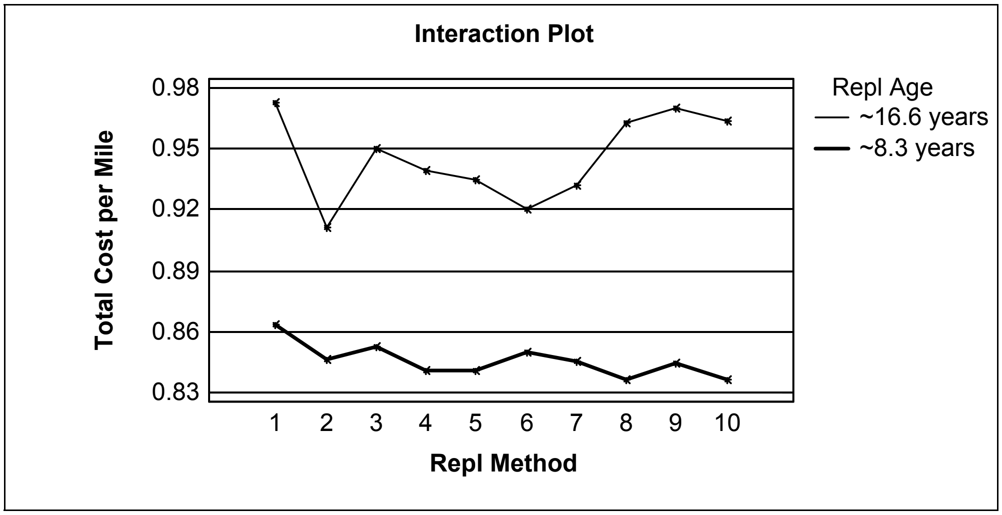


Figure 3.9: Replacement age (budget)/Replacement method interaction plot showing interaction determined to be significant in the ANOVA.

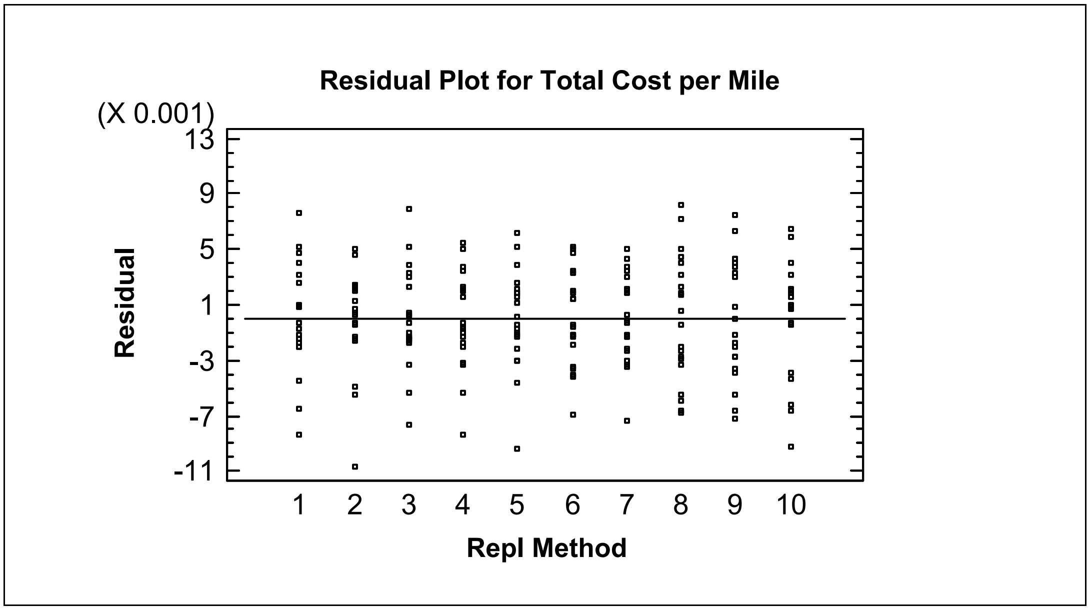


Figure 3.10: Residuals vs. replacement method plot showing no evidence of non-constant variance for different replacement methods.

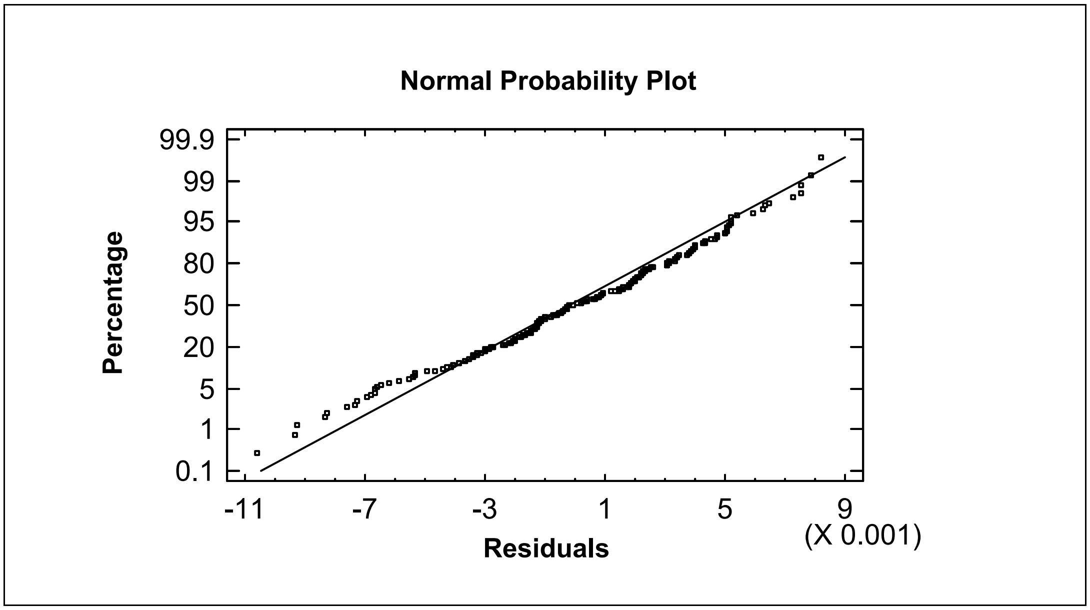


Figure 3.11: Normal probability plot of residuals showing evidence of a normally distributed error term.

3.4.2.2 Multiple comparison tests of replacement methods (analyzed separately for each replacement age)

Multiple comparison tests were conducted to determine significantly different replacement methods. Tukey's multiple range test was applied. Multiple comparison tests are used to make conclusions about more than comparison at an overall confidence level. Tukey's test is applicable to all paired comparisons of different treatment level effects. For comparison of different replacement methods, These tests were applied separately for different replacement ages (budgets) due the presence of significant interaction. The results are shown in Figures 3.12 – 3.13.

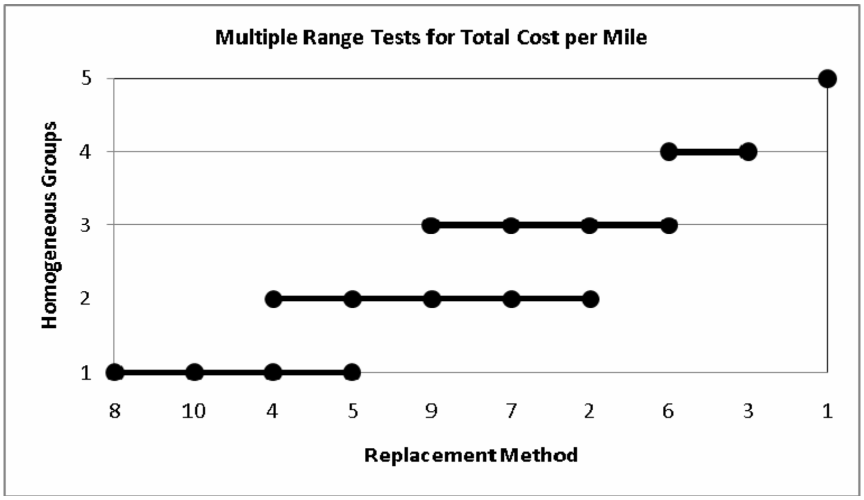


Figure 3.12: Truck HVY DSL: Tukey's multiple range tests for replacement methods for replacement age 8.3. (Replacement methods are sorted from most cost effective (left) to most expensive (right). Not significantly different replacement methods are connected by a line. The overall significance level is 95%.

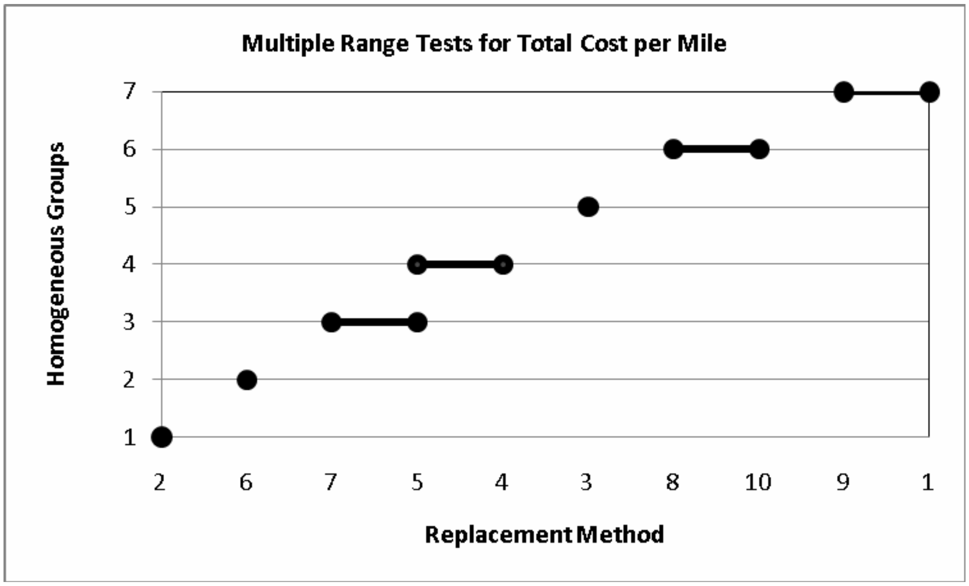


Figure 3.13: Truck HVY DSL: Tukey's multiple range tests for replacement methods for replacement age 16.6. (Replacement methods are sorted from most cost effective (left) to most expensive (right). Not significantly different replacement methods are connected by a line. The overall significance level is 95%

3.4.3 Summary of Results

Since the Replacement Budget (Replacement Age) is significant, and the interaction between Replacement Method and Replacement Age is significant for all equipment classes, the multiple range tests were conducted separately for each Replacement Age. In Tables 3.14 and 3.15 the replacement methods are ranked based on mean total cost per mile. For each equipment class the homogeneous groups based on Tukey's Test resulting in the lowest total cost per mile are shaded grey. The results show that different replacement methods perform differently in different equipment classes and that some equipment classes are more sensitive to the replacement method utilized.

Table 3.14: Replacement ranking for replacement age 8.3 (large budget). The replacement method associated with each number is shown in Table 3.9.

Rank Position	Sedan		Pickup 3/4T 4X2		Truck LT 4X2		Truck MED 4X2 DSL		Truck HVY DSL	
	Repl. Method	Mean	Repl. Method	Mean	Repl. Method	Mean	Repl. Method	Mean	Repl. Method	Mean
1	5	0.159024	2	0.280649	5	0.447442	2	0.930611	8	0.836895
2	4	0.15929	7	0.282965	4	0.447845	6	0.938446	10	0.836993
3	10	0.159488	6	0.28307	8	0.449301	10	0.940776	4	0.84066
4	2	0.159766	5	0.283707	7	0.450003	8	0.942633	5	0.840811
5	7	0.160158	4	0.284296	6	0.451343	9	0.945263	9	0.844684
6	6	0.16029	3	0.28816	2	0.451775	7	0.962612	7	0.84577
7	3	0.162047	8	0.294582	9	0.453836	1	0.968393	2	0.846201
8	8	0.164933	10	0.295352	3	0.454942	5	0.969443	6	0.849888
9	1	0.165134	9	0.298158	10	0.457524	4	0.970302	3	0.852362
10	9	0.166272	1	0.30015	1	0.468842	3	0.983592	1	0.86398

Table 3.15: Replacement ranking for replacement age 16.6 (small budget)

Rank Position	Sedan		Pickup 3/4T 4X2		Truck LT 4X2		Truck MED 4X2 DSL		Truck HVY DSL	
	Repl. Method	Mean	Repl. Method	Mean	Repl. Method	Mean	Repl. Method	Mean	Repl. Method	Mean
1	2	0.186015	2	0.338837	8	0.517714	10	0.968155	2	0.91147
2	5	0.186124	5	0.339869	9	0.519548	8	0.980852	6	0.920462
3	6	0.186358	6	0.339969	10	0.525752	9	0.981539	7	0.93162
4	7	0.186486	4	0.340048	3	0.530269	1	0.984195	5	0.934821
5	4	0.186516	7	0.340055	1	0.531522	4	1.00744	4	0.938896
6	3	0.188267	3	0.340968	4	0.532199	3	1.01109	3	0.950193
7	10	0.189496	8	0.346263	5	0.533952	5	1.01134	8	0.963016
8	8	0.191887	10	0.347184	7	0.539078	7	1.01535	10	0.963057
9	1	0.191941	1	0.347414	6	0.542597	2	1.01957	9	0.970022
10	9	0.192192	9	0.348107	2	0.543707	6	1.01959	1	0.972363

Of the ten criteria/models tested, the overall two most cost-effective criteria for prioritizing equipment within a class for replacement are:

1. Replace the oldest equipment first, Method #2 from Table 3.9; and
2. Method #5, #7, and #8 from Table 3.9 all perform about the same.

Based on discussions with Fleet Services personnel, it is recommended that Method #7, and/or #8 be used as a simple model to prioritize equipment across equipment classes. The software tool developed can utilize either method. Based on intuition and experience, Fleet Services personnel were not comfortable with methods #2 and #5.

Also, since these models/criteria will be used to prioritize equipment replacement across multiple equipment classes, as found in the ODOT fleet, methods #7 and #8 are recommended since both methods compute values that are relative to fixed standards for a particular equipment class.

4.0 ANALYSIS OF FLEET REPLACEMENT AND DECISIONS WHEN ASSET UTILIZATION DECREASES WITH AGE

The economic life of an asset ends when continued use of the asset would cost more than purchasing a replacement. Economic replacement analysis can be used to determine the optimal age of replacement and to choose the best replacement alternative.

The measure frequently used to determine optimal replacement age is equivalent annual cost (EAC), which may also be referred to as a rent or payment. This measure is obtained by adding all costs over the lifetime of an asset, discounted to the time of purchase, and dividing by the number of years in service.

One application of economic replacement analysis is in the management of a motor vehicle pool such as that performed by the Oregon Department of Transportation (ODOT) Fleet Services Section. As vehicles age, a decision must be made whether to continue maintaining the existing vehicle or to purchase a new vehicle. Replacement analysis can also be applied to production equipment used in a factory and many other situations.

Countless improvements have extended economic replacement models over the years, but many build upon a relatively simple basic model. This model analyzes the case of a single asset. It assumes that annual operating and maintenance (O&M) costs increase as the asset ages; this results in an upward curve for that portion of the EAC. On the other hand, the portion of the EAC made up by the initial capital investment is ever-decreasing, as upfront costs get spread over a longer period of time. Adding these two components together (to get the total EAC) results in a U-shaped curve – making the minimum relatively easily to find (Figure 4.1).

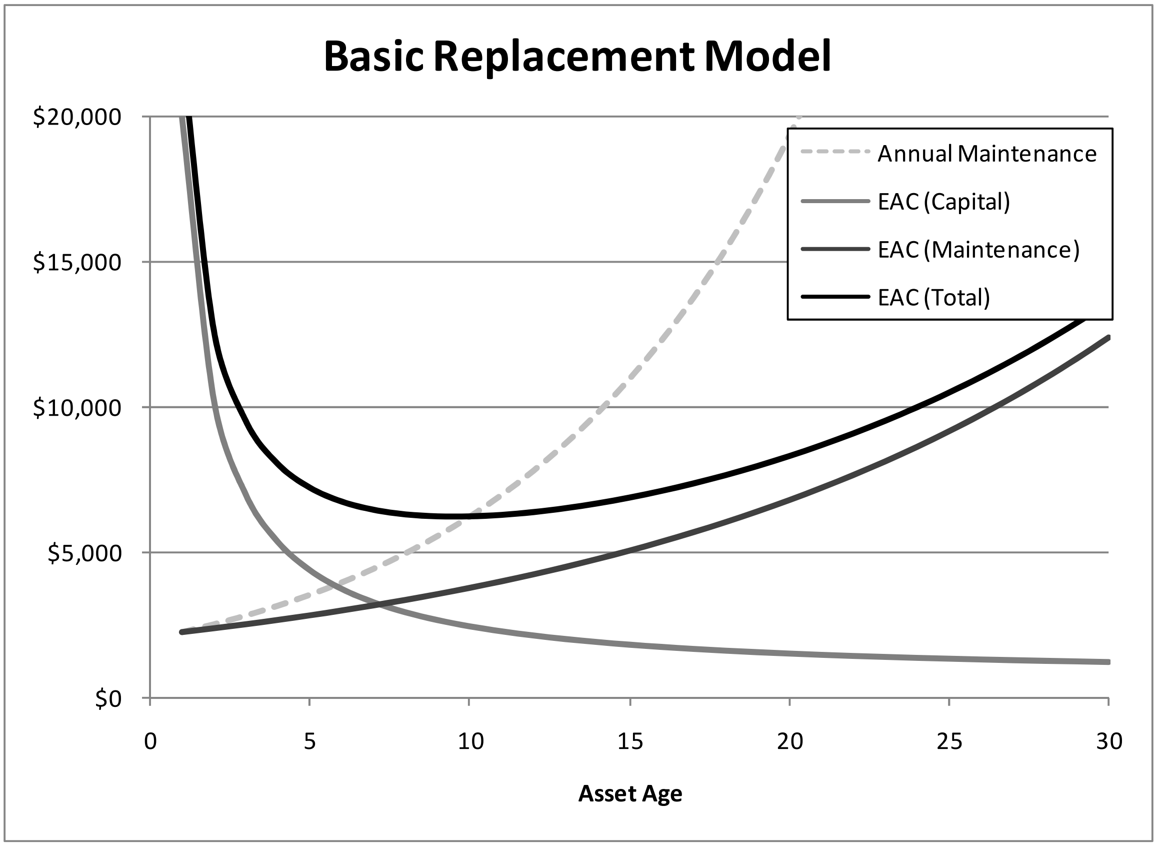


Figure 4.1: Basic replacement model

An examination of data from the ODOT fleet turns up many cases where maintenance costs do not show a smooth increase over time, however. ODOT breaks maintenance costs into three categories: operating costs (fuel and motor oil), fixed costs (insurance), and repair costs. Figure 4.2 plots average operating, fixed, and repair costs with respect to age, after adjustment using relevant consumer price indices (*Kriett 2008*). The data includes costs for over 300 heavy diesel trucks.

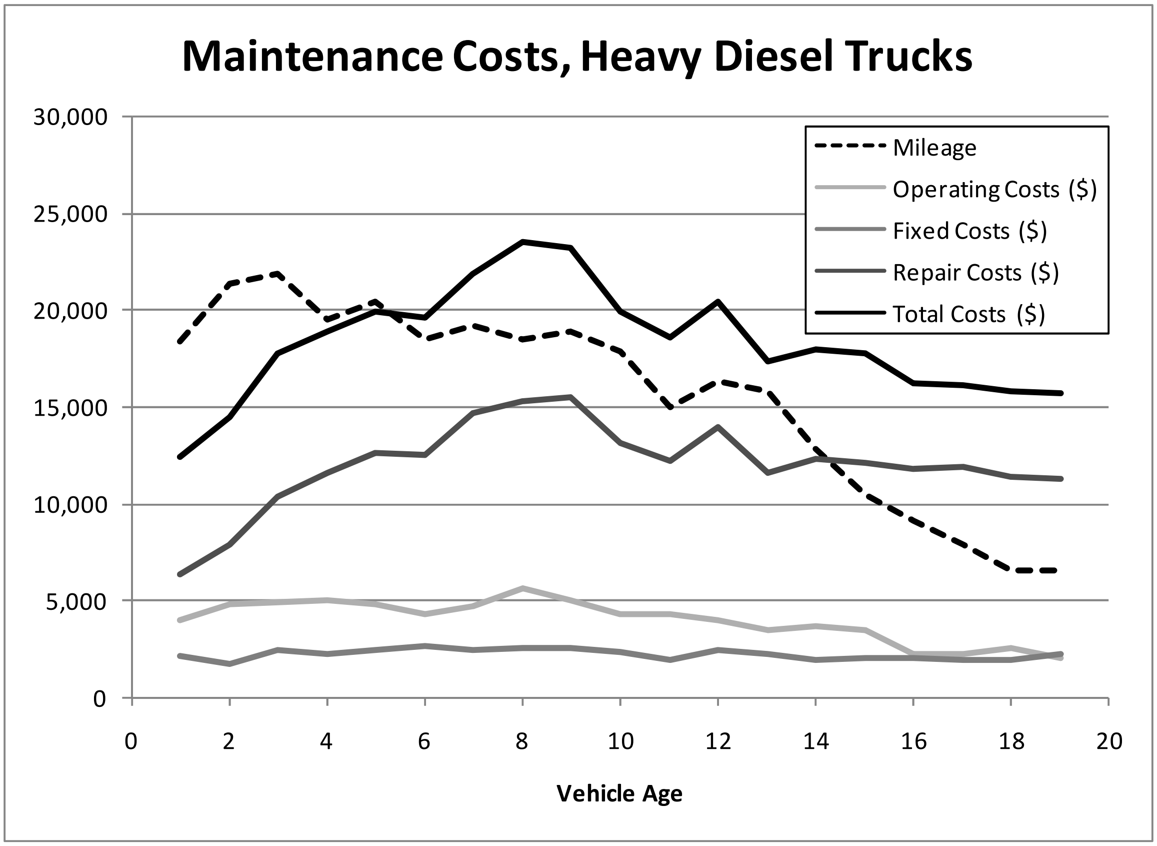


Figure 4.2: Average maintenance costs, heavy diesel trucks

In this case, the total cost curve for the "average vehicle" peaks, then *declines* with age. If this data were used to compute an economic life for vehicles in this fleet (an age standard) the O&M component of the EAC curve would not steadily increase, and the total EAC curve would not be U-shaped. This implies that an asset has economic immortality!

It is important to keep in mind that Figure 4.2 refers to pool-wide average data. Data for individual assets shows a wide degree of variation. Nevertheless, the same general trend can be observed when all of the data are considered. Figure 4.3 shows an example for operating costs.

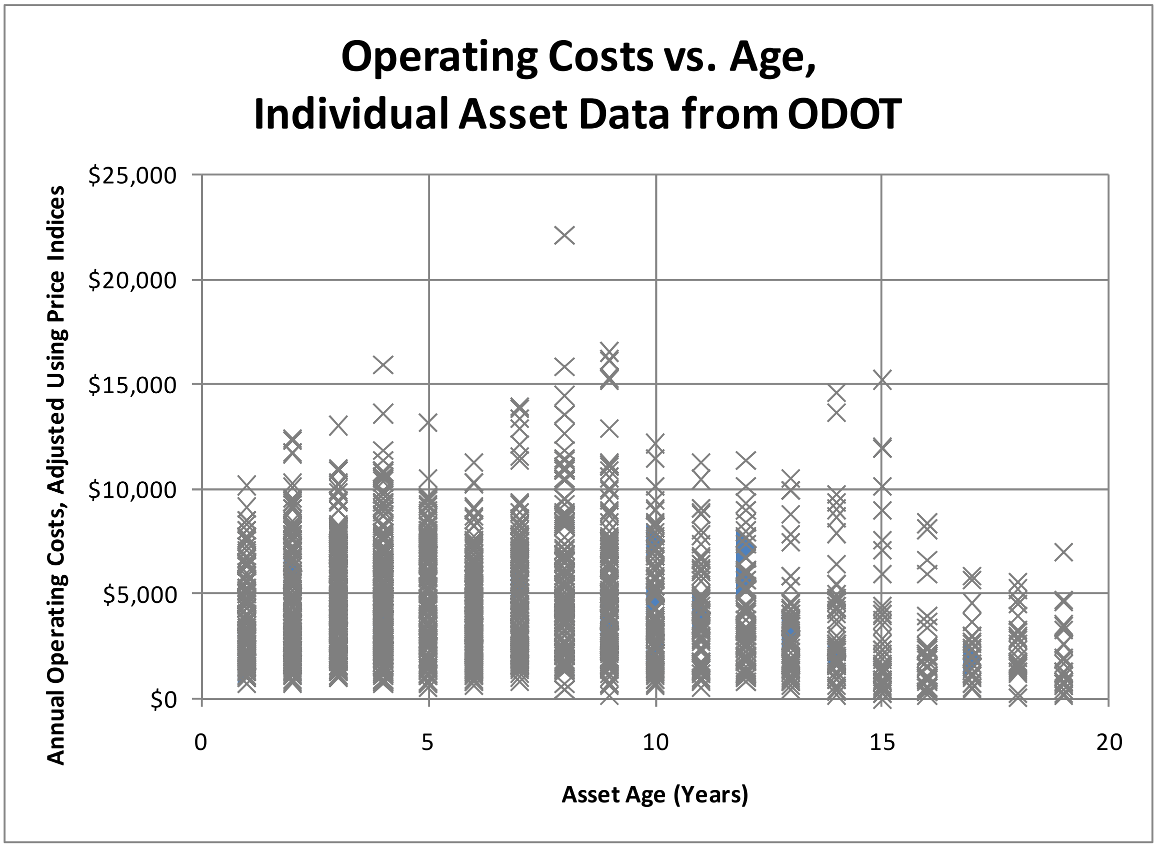


Figure 4.3: Operating costs vs. age, individual asset data from ODOT

In the average asset case, a relatively steady decreasing trend can also be observed with regard to annual mileage driven, analogous to a declining utilization of the asset over time. At ODOT, work crew preference for using the newest vehicle available may be a practical explanation for the trend observed. Dependence of O&M costs on utilization could explain their eventual decline. However, in a multiple asset situation with constant total demand, reduced utilization of one asset would need to be offset by increased utilization of another asset in order to maintain the desired level of service, implying an interdependence of assets with respect to utilization. This illustrates a significant difference between using a single versus a parallel asset replacement model to represent such a situation. The above implication of unlimited economic life would be contradicted, since all assets could not simultaneously remain in the low-utilization phase of life.

This study investigated situations in which multiple assets provide capacity that exceeds demand. If jobs arrive at random intervals, excess capacity is desirable in order to limit the number of jobs queued or turned away. In this type of situation, there is also flexibility in how work is assigned amongst assets. The research question addressed in this study was, “how do different policies of allocating work to multiple assets affect the optimal asset replacement age and the total cost of operation?”

Two primary policies for allocating work across multiple assets were investigated. The first approach, or the "random" approach, assigns a newly arrived to job to any available asset (with equal probability of choosing each). The second approach, or "newest first" approach, is similar to what has been observed at ODOT. Under this policy, a new job is always assigned to the newest available asset. Ties result in random allocation, with each of the newest available assets assigned an equal probability (other assets are assigned a probability of zero).

4.1 RELATED RESEARCH

Replacement models that treat utilization as a variable do exist. Bethuyne (*1998*) looked at replacement based not only on age, but on output rate. If an asset is utilized at a constant rate that is less than its full capacity, its lifetime can be prolonged, resulting in a lower equivalent annual cost (EAC). The same results can be obtained by utilizing an asset at its full capacity until the end of its economic life, then continuing to use it at a reduced level. A key consideration here is that reduced utilization of a single asset does not provide the same level of service. The author did not explore the impact of these effects on a multiple-asset problem.

Hartman (*2001*) defined a single-asset model with discrete variable asset utilization. Utilization was probabilistic and depended upon utilization in the previous period (resulting in a Markov chain-like relationship). Cumulative usage was also taken into account when calculating costs.

As appears to be the case for ODOT's heavy diesel trucks, it is important to consider the interactions between multiple assets in a replacement model. Several papers have been published investigating these types of parallel replacement problems. In such cases, multiple assets share the responsibility of meeting demand. Multiple replacement decisions are complicated by a variety of interdependent relationships between assets.

Karabakal et al. (*1994*) used a parallel replacement model to study the interdependence of replacement decisions under capital rationing constraints, but did not investigate other aspects of the multiple-asset problem. Capital constraints are relevant considering that many replacement decisions are made within the context of a limited budget.

Hartman (*1999*; *2004*) studied multiple-asset replacement problems with a focus on optimally allocating usage amongst assets. Hartman (*1999*) developed a multiple-asset integer programming model treating utilization as a decision variable to be optimized across multiple assets. Hartman (*2004*) more recently applied a dynamic programming approach, including an O&M cost function that assumed costs were non-decreasing in utilization level (u), age (i), and cumulative utilization (j). Much of the paper was limited to a two-asset case. Optimization problems become increasingly difficult to solve as the number of integer variables grow, particularly if each additional variable is reflected across multiple dimensions. Due to this "curse of dimensionality," the author's dynamic programming approach encountered difficulties when applied to a greater number of assets.

4.2 ANALYSIS OF COSTS FOR DIFFERING WORK ALLOCATION POLICIES

First, the question of how maintenance costs vary under different work allocation policies was addressed. To facilitate the analysis, all costs were assumed to fall into one or a combination of the following cost categories: constant, age-dependent, and/or usage-dependent. A constant annual cost was defined to remain the same from year to year, regardless of the age of the asset or how much it has been used. A vehicle-related example could be insurance costs. In this sense, a constant annual cost varies from a typical fixed cost. Fixed costs might be better included as part of the asset's capital costs. Age-dependent annual costs were defined to depend only upon the age of the asset. It may be difficult to separate out costs that are strictly age-dependent in most practical cases, as a usage-related component is likely present as well.

Usage-dependent costs were split into two categories. The first category was defined to include costs that were dependent only upon annual usage. An example might be fuel costs for a vehicle, which are directly related to how much that vehicle was driven in a given year. The second category was cumulative usage-dependent costs. These costs were defined to depend upon how much an asset had been used over its entire life to date. Repair costs for a vehicle could fall into this category, but it is important to note that many real costs likely cover a territory that includes more than one cost category (repair costs possibly included). Figure 4.4 gives a visual representation of the relationships between annual cost categories.

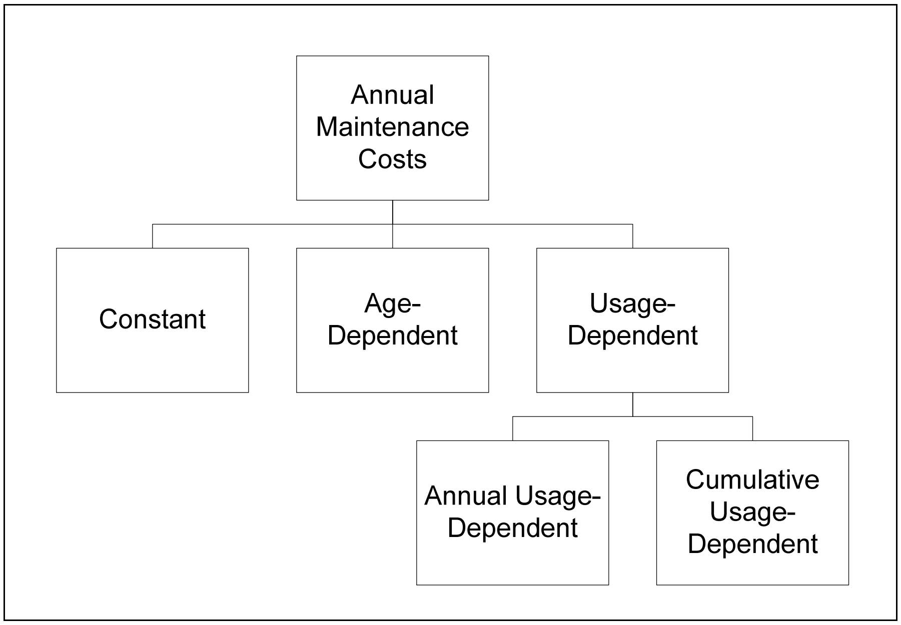


Figure 4.4: Annual maintenance cost categories

Costs were studied under a fixed age replacement policy. Under this type of policy, an asset that reached the maximum (replacement) age was replaced regardless of other factors (e.g., how much it had been used over its lifetime). The maximum age was the same under the two work allocation policies investigated.

4.2.1 Usage versus Age under Varying Job Allocation Policies

Two distinct patterns for the usage versus age curve of individual assets may be observed upon consideration of the two job allocation policies described above (i.e., random or newest first). In the random job allocation case, utilization remains the same from year to year with minor fluctuations due to random effects. An asset has the same chance of being used at the beginning of its life as at the end of its life. Hence, a graph of annual usage versus age for an asset will approximate a straight line with a slope of zero.

In the newest first job allocation case, an asset is generally assigned more jobs towards the beginning of its life and less towards the end of its life. Hence, its utilization is a non-increasing function with respect to age, again with minor fluctuations due to random effects. The exact shape of the curve depends upon the composition of the entire pool of assets. In the extreme case where all of the assets are the same age, a duplication of the random allocation curve is observed, since there are no meaningful age distinctions. Also, in cases where capacity is closely matched to demand, there is not as significant of a decrease in the curve (since the oldest asset still gets considerable use). Bearing in mind these exceptions, a general idea of the difference between the curves can be observed in Figure 4.5.

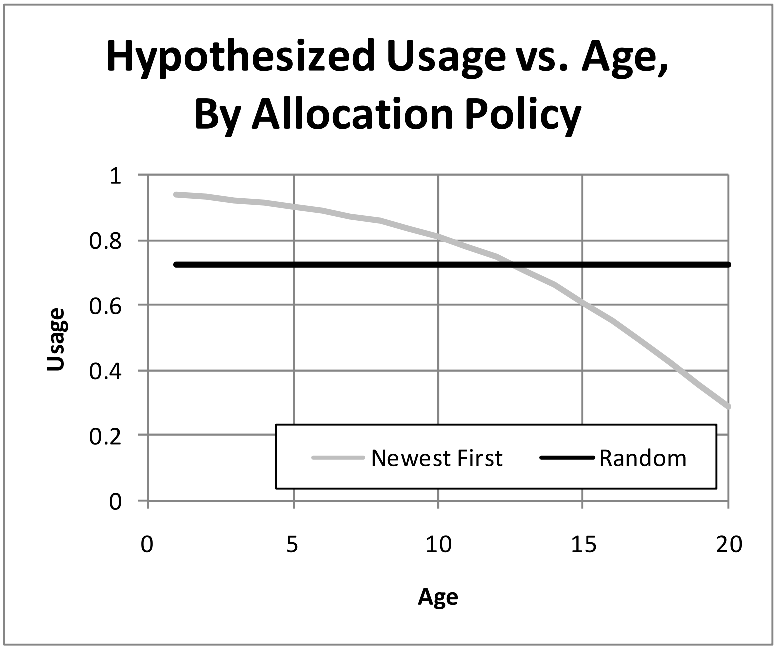


Figure 4.5: Hypothesized usage vs. age, by allocation policy

4.2.2 Constant, Age-Dependent, and Annual Usage-Dependent Costs

The usage-versus-age curves from Figure 4.5 were used to predict how different job allocation policies affected each of the identified cost categories.

Firstly, constant annual costs are not affected by a different lifetime usage distribution. Neither are costs that are strictly age-dependent. If usage plays no factor in determining the cost, there is no reason why distributing usage differently over a lifetime of unchanged length should give different results.

On the other hand, costs that are dependent only on annual usage follow a predictable pattern based upon the shape of the function used to determine costs. Hartman (*2004*) discussed a two-asset case where maximum per-asset utilization in a period was defined to be ū and total demand for the period was defined as d_t. The author considered several cost curves as a function of annual usage, assuming smooth, non-decreasing curves as in Figure 4.6.

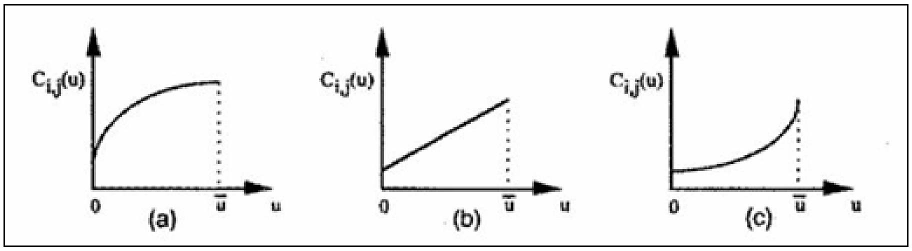


Figure 4.6: Cost as a function of annual usage (*Hartman 2004*)

In case (a), with marginally decreasing costs, costs were minimized by using one asset at full capacity (ū) and the second asset only enough to meet demand (d_t – ū). One method of implementing this could be the newest first policy, assuming that age classes are adequately distributed. The newest first policy would clearly be preferable to the random policy in this case. Case (b), on the other hand, resulted in a situation where costs could be minimized through any linear combination of asset utilization, since the curve was linear. Pool-wide usage adds to meet total demand, which can be found on the curve directly. In this situation, neither the random nor the newest first job allocation policy has an advantage. Finally, in (c), marginally increasing costs demanded that each asset be used at the minimum level possible (both assets are used at the level of $d_t/2$). This is equivalent to the random allocation case, which results in lower total costs than the newest first policy for case (c).

4.2.3 Cumulative Usage-Dependent Costs

Costs that depend on cumulative usage introduced new complexity. Insight was gained by translating the usage versus age curve (Figure 4.5) into a graph of cumulative usage versus age, resulting in a graph similar to Figure 4.7. The new graph was generated by summing the annual usage for each year up to age t. If continuous values were used, an integral could have been taken instead.

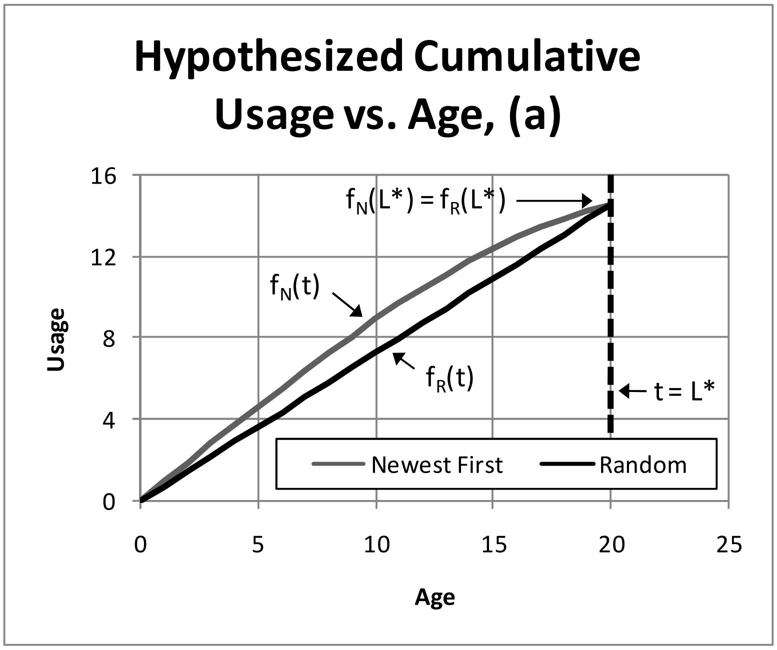


Figure 4.7: Hypothesized cumulative usage vs. age for an individual asset (a): $f_N(L^*) = f_R(L^*)$

In Figure 4.7, $f_R(t)$ gives the cumulative utilization at time t under the random allocation policy and $f_N(t)$ gives the cumulative utilization at time t under the newest first allocation policy. L* was defined as the age at which an asset is replaced. It was observed that **fN(t) ≥ fR(t) for 0 ≥ t ≥ L*.** By assuming that annual cost as a function of cumulative usage, $C_{CU}(f(t))$, is non-decreasing, it was shown that **C_{CU}(fN(t)) ≥ C_{CU}(fR(t)) for 0 ≥ t ≥ L*.** Summing all annual $C_{CU}(f(t))$ values under the newest first allocation policy results in greater or equal cost over the lifetime of the asset, since each term in the summation is greater than its equivalent under a random allocation policy.

Figure 4.7 assumes a case where cumulative usage at the end of the asset's life, t = L*, is equal under either job allocation policy. This might be true if the initial pool of assets is evenly distributed by age, but is unlikely if asset ages are unevenly distributed. Consider a case where a large group of assets are of nearly the same age. Some assets (i.e., the youngest among the large group) will have above-average cumulative usage at the end of their life, as represented in Figure 4.8.

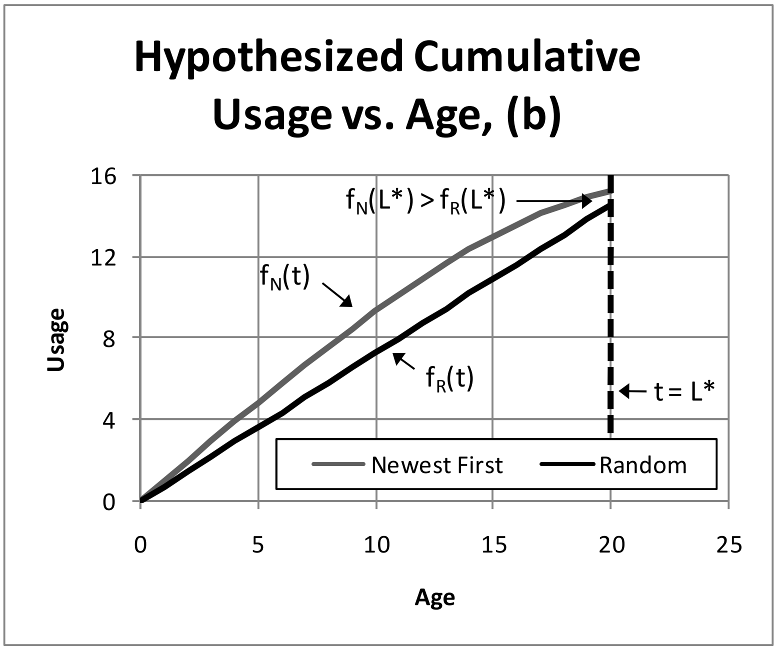


Figure 4.8: Hypothesized cumulative usage vs. age for an individual asset (b): $f_N(L^*) > f_R(L^*)$

This case does not conflict with the above conclusions. In fact, the difference in costs between allocation methods is magnified. To the contrary, Figure 4.9 demonstrates a case where an asset may have below average cumulative usage at the end of its life. This might occur if the asset is amongst the oldest of a large group of assets of nearly equal age.

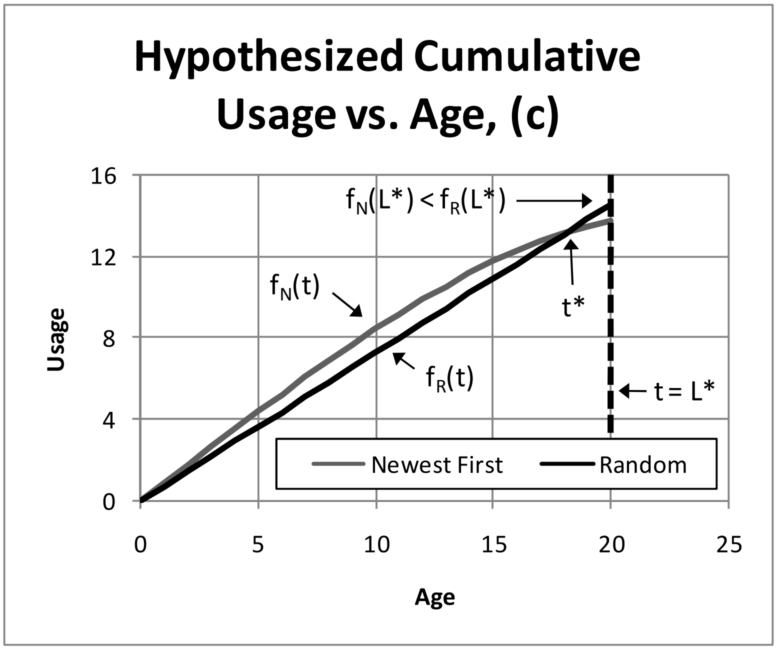


Figure 4.9: Hypothesized cumulative usage vs. age for an individual asset (c): $f_N(L^*) < f_R(L^*)$

In this case, **$fN(t) \geq fR(t)$ for $0 \geq t \geq t^*$, if t^* is the point at which $fN(t)$ and $fR(t)$ intersect. After t^*, however, the situation is reversed, with $fR(t) \geq fN(t)$ for $t^* \geq t \geq L^*$.** Using the logic described before, **$C_{CU}(fN(t)) \geq C_{CU}(fR(t))$ for $0 \geq t \geq t^*$, but $C_{CU}(fR(t)) \geq C_{CU}(fN(t))$ for $t^* \geq t \geq L^*$.** A clear-cut determination cannot be made as to which allocation policy will result in the minimal cost over the lifetime of the asset.

However, a conclusion was reached when costs were considered on a pool-wide basis. Since all assets are replaced at the same age, regardless of where they are in the age distribution of the initial pool, ongoing replacement results in a cycle with length equal to the replacement age, L*. If a snapshot of the pool is taken at any time t_1 after the initial cycle, and again at $t_1 + L^*$, the age distribution of the assets within the pool is identical at both points in time. Due to this cycling, the interval between those two snapshots encompasses what is *equivalent* to exactly one full lifetime for each asset, whether or not the lifetime for a particular asset starts at t_1.

If it is assumed that annual pool-wide demand remains constant, summing the cumulative usage at the end of all assets' lifetimes will equal the total demand over the interval. Hence, the pool-wide cumulative utilization versus age curves for the newest first and random allocation policies have equal end points; that is, **$\sum fN(L^*) = \sum fR(L^*)$.** Each $f_N(t)$ and $f_R(t)$ curve is non-decreasing, so $\sum f_N(t)$ and $\sum f_R(t)$ are also non-decreasing. If all assumptions hold true, then **$\sum fN(t) \geq \sum fR(t)$ for $0 \geq t \geq L^*$ and $C_{CU}(\sum fN(t)) \geq C_{CU}(\sum fR(t))$ for $0 \geq t \geq L^*$.** Expressed in general terms, the newest first allocation policy results in costs greater than or equal to costs under the random allocation policy – regardless of the initial age distribution of the assets.

4.2.4 Hypotheses

Simulation was used to test hypotheses relating to the research question. Assumptions and reasoning are described in detail above. The specific hypotheses tested were the following:

1. Pool-wide average annual asset usage under the random allocation policy will remain constant with respect to age.
2. Pool-wide average annual asset usage under the newest first allocation policy will decrease with respect to age.
3. Pool-wide average age-dependent costs will be equal under both allocation policies.
4. Pool-wide average annual usage-dependent costs will be equal under both allocation policies if the cost function is linear with respect to annual usage.
5. Pool-wide average annual usage-dependent costs will be greater than or equal to costs under the random allocation policy if the cost function has negative concavity with respect to annual usage.
6. Pool-wide average annual usage-dependent costs will be greater than or equal to costs under the newest first allocation policy if the cost function has positive concavity with respect to annual usage.
7. Pool-wide average annual cumulative usage for an asset under the random policy will increase linearly with respect to age.

8. Pool-wide average annual cumulative usage for an asset with respect to age under the newest first policy will be greater than or equal to pool-wide average annual cumulative usage for an asset with respect to age under the random policy.
9. Pool-wide average annual cumulative usage-dependent costs for an asset with respect to age under the newest first policy will be greater than or equal to pool-wide average annual cumulative usage-dependent costs for an asset with respect to age under the random policy if the cost function is non-decreasing with respect to cumulative usage.

4.2.5 Comparison of Work Allocation Policies While Varying Replacement Age

The behavior of pool-wide costs was contrasted between the two job allocation policies when a single, fixed replacement age was applied across the entire pool. In the basic economic replacement model, however, EAC is graphed against many potential replacement ages for an individual asset, as illustrated in Figure 4.1. The minimum of this curve identifies the optimal replacement age (i.e. the age that will result in the lowest total cost).

How do pool-wide costs for the two job allocation policies vary with respect to replacement age? Due to the complexities of these calculations, simulation was used to answer this question.

4.3 METHOD

4.3.1 Reasons for Simulation

Statistical simulation provided a convenient method to compare different job allocation policies. Although it might have been possible to calculate expected utilizations for assets in a priority-based system (such as assigning jobs to the newest available asset) using queuing models, the calculations would have quickly become very laborious for a pool of any significant size. In a statistical simulation, the fairly simple logic behind different allocation policies was programmed to apply, based on the asset pool's current state, upon the arrival of a new job. This also resulted in utilization that was measured based on jobs of varying length. Much of the existing literature relied upon measuring asset utilization in discrete increments. The expected effects for different job allocation policies on the total cost of maintaining a pool of assets were hypothesized. Simulation provided one manner in which these hypotheses could be tested.

In the context of actual cost data, simulation could also be applied to determine how *significant* cost differences are between the allocation policies. Take a practical example: it was hypothesized that a newest first selection policy can frequently result in higher costs, and observations have indicated that this is how vehicles are commonly assigned within the ODOT pool (*Kriett 2008*). If ODOT wanted to change its assignment policy to random, would the expected savings be worth the considerable effort involved? Making significant procedural changes in a large organization has a cost as well (e.g., the time supervisors must spend enforcing the new policy). Kobbacy and Nicol (*1994*) used a statistical simulation to conduct sensitivity analysis for a single asset replacement problem. In this study, statistical simulation was applied to a multiple asset case.

4.3.2 Simulation Structure

The statistical simulation was programmed using Visual Basic 2008 Express Edition. Simulation parameters were pulled from a Microsoft Access database. Simulation output data were exported to the same database for further analysis. The core functionality of the simulation was based upon maintaining a list of three classes of events: year end, replacements, and other. These event classes are defined in the next section.

4.3.2.1 Year End Event

For each asset, usage and costs from the previous year were recorded in the simulation. Cumulative usage for each asset was also tracked continuously. Any number of annual costs could be specified as parameters, and they were all calculated and recorded separately. Parameters for each cost included the following:

- Cost type (constant, age-dependent, annual usage-dependent, or cumulative usage-dependent)
- Cost concavity for non-constant costs (zero/linear, positive, or negative)
- Exponential growth rate for non-linear costs

All non-linear costs used the formulas below, where r was the exponential growth rate and x was the independent variable (e.g., age, annual usage, cumulative usage, etc.).

Positive concavity: **erx / (er - 1) - (er - 1)-1**
Negative concavity: **e-rx / (e-r - 1) - (e-r - 1)-1**

These formulas were adjusted to result in a cost of 0 when x = 0 and a cost of 1 when x = 1. If the cost being represented had an intercept (i.e., the cost did not equal zero when x = 0), the intercept could be represented using a separate constant cost. A higher exponential growth rate results in a curve that deviates more from the linear cost function. An example with an exponential growth rate of three (r = 3) can be observed in Figure 4.10.

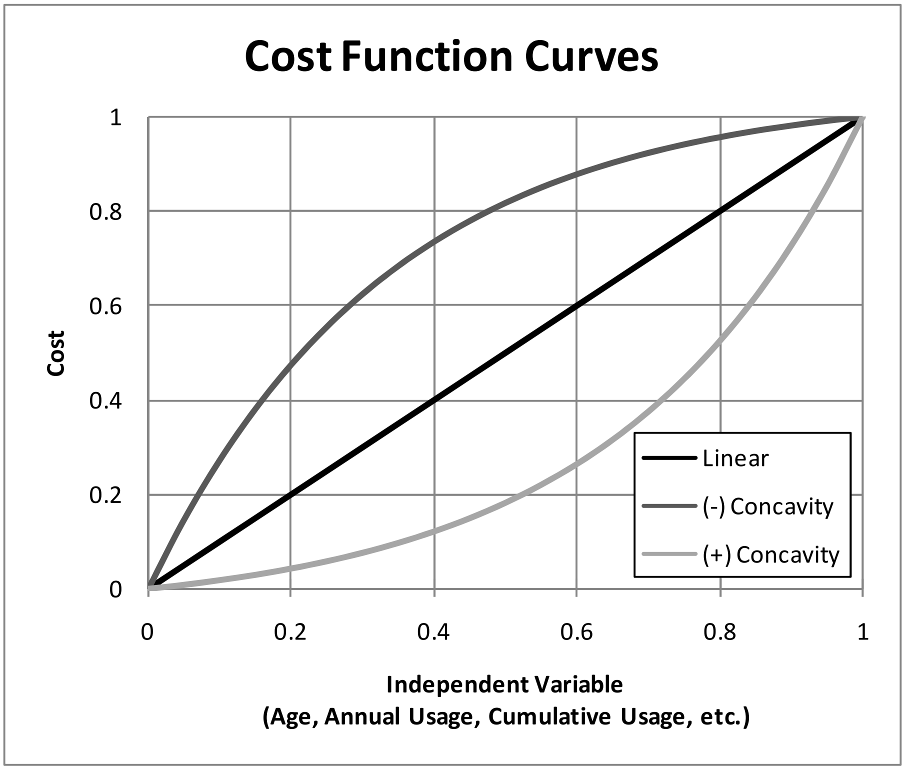


Figure 4.10: Cost function curves for r = 3

4.3.2.2 Replacement Event

All assets were evaluated for replacement based upon specified criteria. Optionally specified criteria included the following:

- Replacement cycle frequency, in years
- Maximum age (requires replacement)
- Maximum cumulative usage (requires replacement)
- Minimum age for replacement
- Minimum cumulative usage for replacement
- Maximum replacement quantity (could be superseded by replacements for maximum age or cumulative usage)
- Weighting assigned to age in selecting replacements (applied to standard deviations from pool mean)
- Weighting assigned to cumulative usage in selecting replacements (applied to standard deviations from pool mean)

4.3.2.3 Other Events

A simulation end event was used to mark the end of the simulation. The occurrence of this type of event cleared any events remaining in the queue. Output data were then recorded to the database for analysis.

Jobs arrived in the form of job arrival events. Jobs had exponentially distributed arrival times with the mean rate of arrival specified as a parameter. There was no job queue; if all assets were busy and new job arrived, the job was rejected. The rate of rejection was recorded. When a job arrived, it was assigned using one of two allocation schemes: newest first or random. The allocation scheme was specified as a parameter.

When a job was assigned, a corresponding job departure event would be added to the queue. Jobs had exponentially distributed service times with the mean service rate specified as a parameter. Upon occurrence of a job departure event, the relevant asset was again made available for use. Departure events for assets that had since been retired were ignored.

4.3.3 Experiment 1: Annual Usage and Cumulative Usage Costs

4.3.3.1 Purpose

The purpose of the first experiment was to test the relationships between annual and cumulative usage-dependent costs for the two job allocation policies, under fixed-age replacement.

4.3.3.2 Construction

A maximum replacement age was set at 20 years. All other replacement parameters were turned off to result in a simple, fixed-age replacement scenario. Six costs were calculated for each replication:

- Linear annual usage-dependent cost
- Negative concavity annual usage-dependent cost (exponential growth rate of 3)
- Positive concavity annual usage-dependent cost (exponential growth rate of 3)
- Linear cumulative usage-dependent cost
- Negative concavity cumulative usage-dependent cost (exponential growth rate of 3)
- Positive concavity cumulative usage-dependent cost (exponential growth rate of 3)

The initial pool of assets was constructed based on ODOT data for heavy diesel trucks. A discrete distribution for asset age was assembled based on the current state of the ODOT fleet (Figure 4.11). Asset ages were randomly chosen from this distribution to assemble an initial pool of 20 assets (Figure 4.12). Each asset was assigned an initial cumulative utilization based on the average ODOT value for a heavy diesel truck of that age.

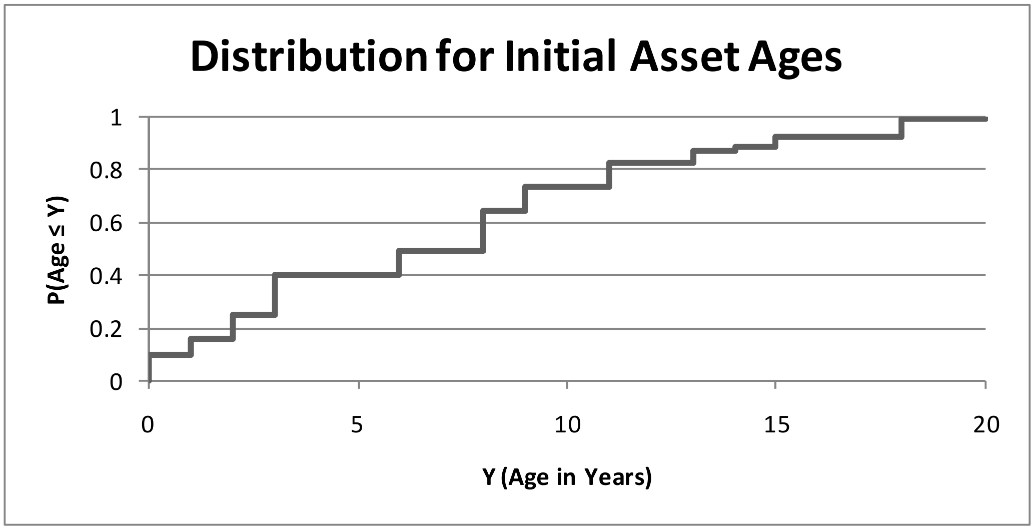


Figure 4.11: Distribution from which initial asset pool ages were selected

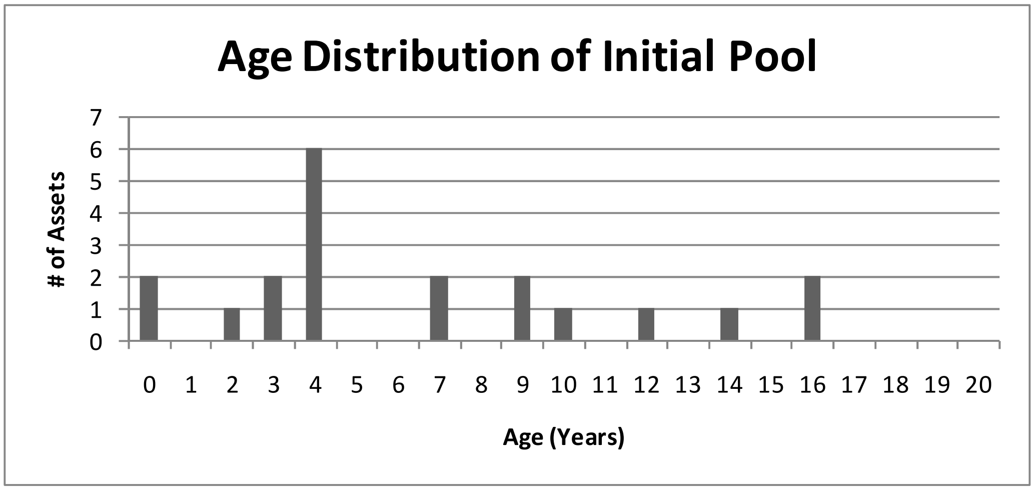


Figure 4.12: Age makeup of initial asset pool used in Experiment 1

An asset's annual usage under the newest first job allocation policy is dependent upon its age rank, or priority class, relative to the entire pool (with minor variation due to random job arrivals). If a particular asset is the newest of all assets in the pool, it will have the top priority for assignment to a new job (it will have the highest rank). Likewise, the oldest asset in the pool will have the lowest priority for assignment to a new job. Since all assets within the pool are replaced with the same frequency, a plot of annual usage versus age for an asset is identical to the plot for its replacement. For example, consider Table 4.1.

The asset with an initial age of two is the oldest asset in the pool in year 2. Five years later (the replacement age in this example), in year 7, it is again the oldest asset in the pool.

Table 4.1: Age of assets within a pool over time under fixed-age replacement

		Year									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
(Initial Age)	(0)	1	2	3	4	5	1	2	3	4	5
	(2)	3	4	5	1	2	3	4	5	1	2
	(4)	5	1	2	3	4	5	1	2	3	4

Because of this effect, a time interval of length equal to the replacement age represents a cycle of asset states that is repeated indefinitely. Therefore, it was not necessary to generate more than one cycle of data in order to analyze costs that were dependent upon annual usage.

However, analysis of cumulative usage-dependent costs required at least two cycles of data. Since cumulative usage is simply a summation of all annual usage values up to that point, it also stays the same from cycle to cycle. However, values in the first cycle differ since each asset in the initial pool is assigned an initial cumulative usage. This cumulative usage value may differ from the cumulative usage that the asset would have accumulated over a regular cycle in the simulation. Cumulative usage for all assets in the second cycle and beyond will have been generated through simulation, and hence will be the same from cycle to cycle. Any cycle after the first cycle was defined as a “stable” cycle.

Since the fixed replacement age was set to 20 years in Experiment 1, each replication was run for a length of 60 years to ensure data collection over two stable cycles as well as the initial cycle. Thirty replications were run under each job allocation scheme (newest first and random). Multiple replications were run so that the results could be averaged to reduce the impact of random job arrivals.

4.3.4 Experiment 2: Variation of Fixed Replacement Interval

4.3.4.1 Purpose

While the first experiment was designed to demonstrate basic patterns for annual usage and cumulative usage-dependent costs under two different job allocation policies, the purpose of the second experiment was to observe how these patterns changed as the replacement age was varied.

4.3.4.2 Construction

The majority of parameters from the first experiment were retained without modification. The principle variable that was altered was replacement age. The replacement age was still held constant within a given simulation run, or replication, resulting in stable cycles that could be observed. The replacement age was varied in one-year increments from 5 years to 50 years. As in the first experiment, thirty replications were run using the random job allocation policy and thirty replications were run using the newest first policy for each of the replacement ages.

For each replication, the simulation length in years was adjusted based upon the replacement age. The simulation length used was equal to three times the replacement age, so that two stable cycles could be observed.

Age-dependent and cumulative usage-dependent costs are scaled in proportion to a typical lifetime value, measured in years. Age-dependent costs for each of the curve types (linear, positive concavity, negative concavity) converge to a value of one when this typical lifetime value is reached. Likewise, cumulative usage-dependent costs for each of the curve types converge to one when the cumulative usage reaches this typical lifetime value. If an asset is used at 100% utilization, it will reach these points at the same time. In the first simulation, this typical lifetime value was the same as the replacement age, 20 years. It remained set at 20 years for the second experiment, but its effect was more significant when the replacement age was varied.

Finally, age-dependent costs were calculated in the second experiment in addition to annual usage and cumulative usage-dependent costs. Three age-dependent cost functions were included, one of each curve type. An exponential growth rate of three was specified for the non-linear curves. All other parameters were the same as described for the first experiment.

4.3.4.3 Additional calculations

In order to compare lifetime pool-wide costs for different replacement ages, equivalent annual costs (EACs) were calculated. Pool-wide costs were taken from the first stable cycle and discounted to the beginning of the cycle. Since the makeup of the initial pool was not changed while the replacement age was varied, it was possible that more replacements were made towards the beginning or end of a cycle. Discounting had the potential to magnify this effect, so multiple EACs were calculated for each simulation under observation. An EAC was calculated over a period equal in length to the replacement age beginning with each of the years in the first stable cycle, and all of these EACs were then averaged. For example, if the replacement age was 5 years, EACs would be calculated for years 5-9, 6-10, 7-11, 8-12, 9-13, and 10-14; they would then be averaged.

4.4 RESULTS

4.4.1 Experiment 1: Annual Usage and Cumulative Usage Costs

4.4.1.1 Usage Versus Age

Before analysis of data on annual usage and cumulative usage-dependent costs, asset usage was inspected to verify the hypothesized trend. A constant usage level with regard to age was expected under the random allocation policy, and a decreasing usage level with regard to age was expected under the newest first allocation policy. The pool-wide averages (for assets during stable cycles) confirmed this, as is shown below in Figure 4.13.

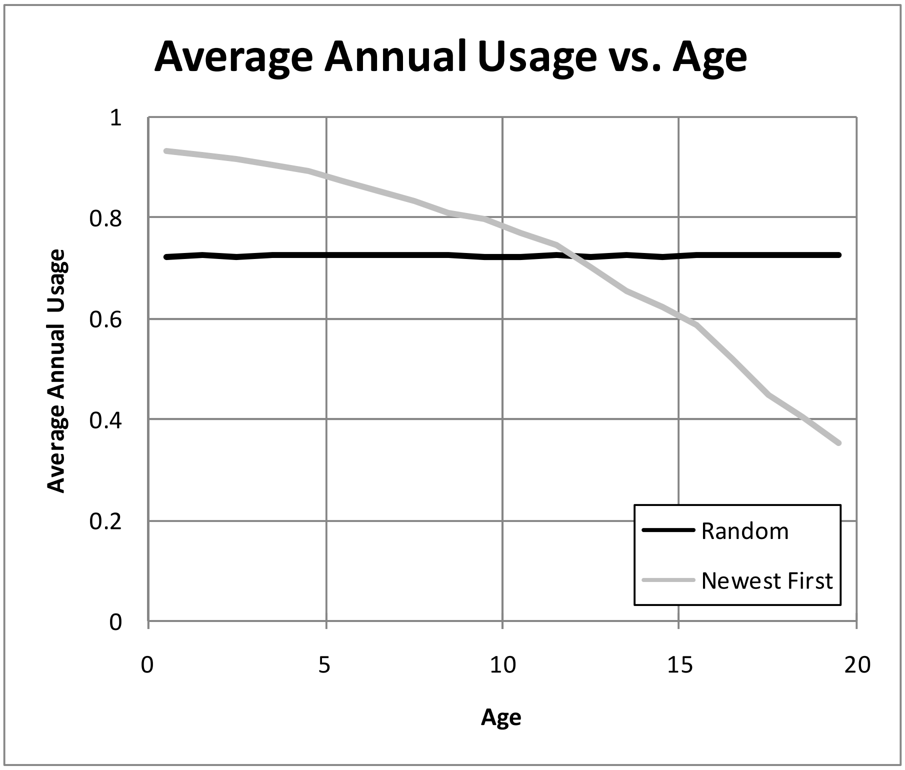


Figure 4.13: Average annual usage vs. age in Experiment 1

Further analysis of the data showed that there was considerable variation between curves for individual assets under the newest first job allocation policy (Figure 4.14), while the random job allocation policy resulted in minimal variation (Figure 4.15).

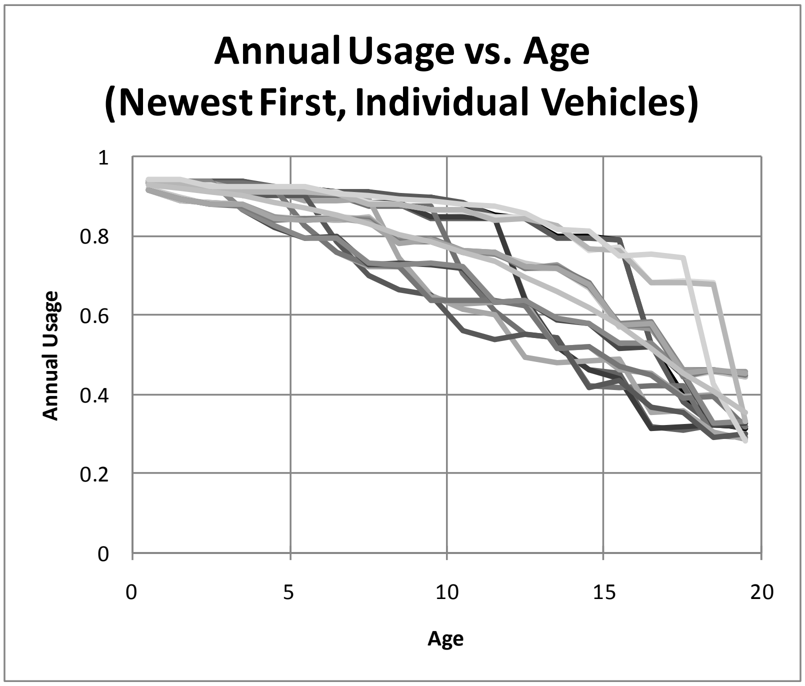


Figure 4.14: Annual usage vs. age (newest first, individual assets) in Experiment 1

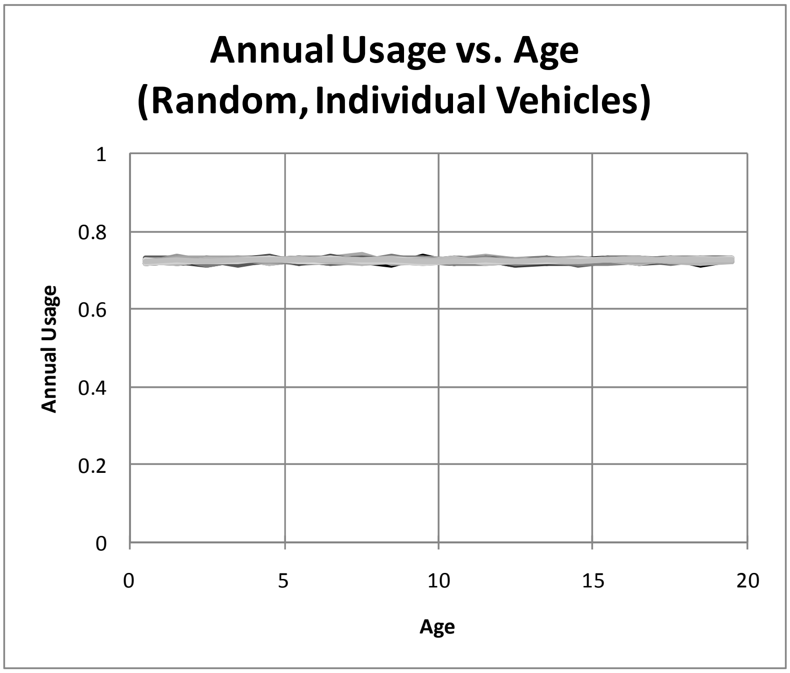


Figure 4.15: Annual usage vs. age (random, individual assets) in Experiment 1

Minor variation under the random policy was due to the random job arrival process. The more significant variation under the newest first policy was largely due to each individual asset's path through multiple age priority classes. Each asset's path is closely related to the initial makeup of the asset pool. For example, if the majority of initial assets are of approximately equal age, the replacement asset for the oldest in the group will spend a lot less time in the top age priority class than the replacement asset for the newest in the group. On the other hand, if exactly one asset is replaced each and every year, the individual asset curves are all essentially identical to the average curve.

Pool-wide average data were also plotted for cumulative usage versus age, in Figure 4.16. Again, data were only taken from the stable cycles. As hypothesized, the curve for the newest first job allocation policy did not go below the (linear) curve for the random policy at any point. It may also be noted that the two average curves arrived at the same cumulative usage value by the end of their 20-year life, despite the fact that end points for individual cumulative usage curves (Figure 4.17) fell above and below the dotted line representing cumulative usage under the random policy.

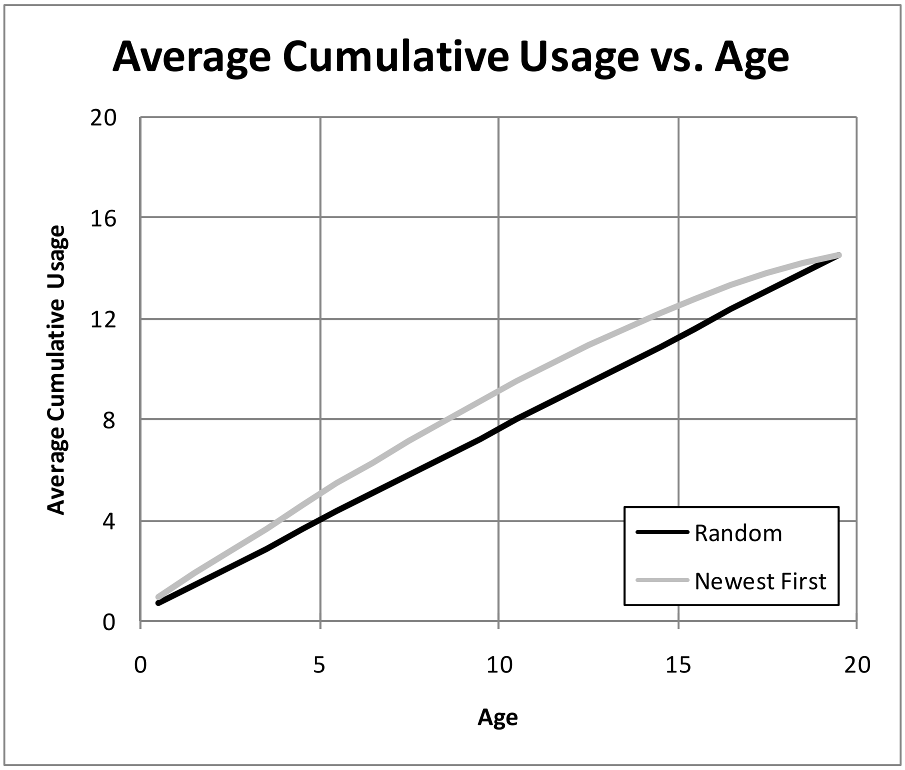


Figure 4.16: Average cumulative usage vs. age in Experiment 1

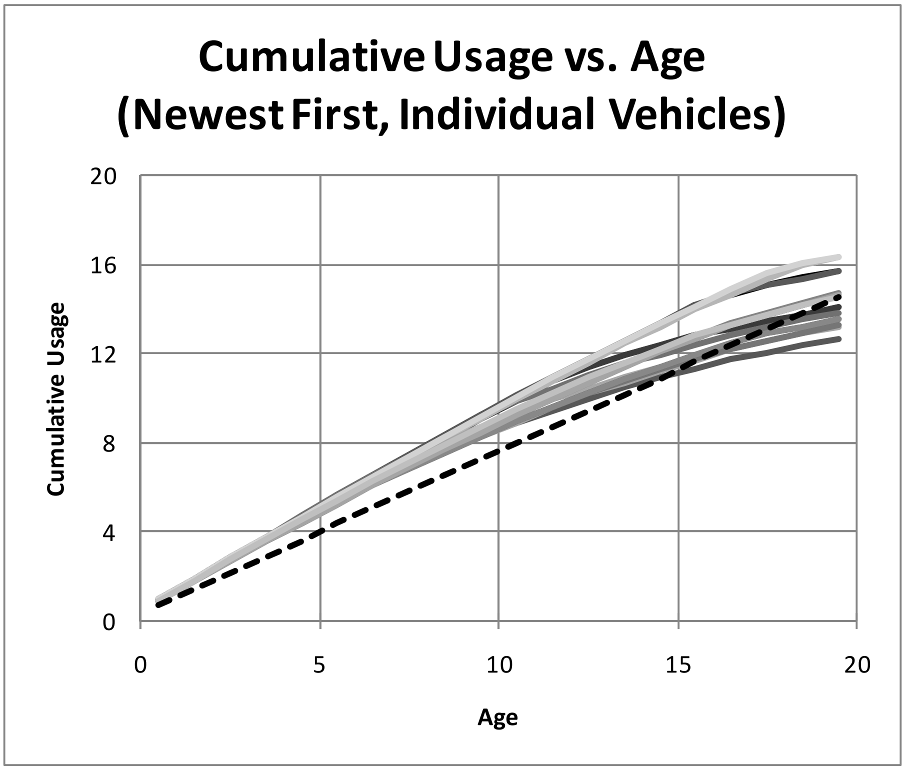


Figure 4.17: Cumulative usage vs. age (newest first, individual assets) in Experiment 1

4.4.1.2 Annual usage-dependent and cumulative usage-dependent costs

Costs were analyzed on a pool-wide level, since that is the level at which total costs are minimized for a multiple asset replacement problem. Costs were compared in relative quantitative terms; no attempt was made to assign actual dollar values to costs.

Behavior of annual usage-dependent costs in the simulation was compared to the hypothesized behavior. For each of the three types of cost curves (linear, negative concavity, and positive concavity), it was expected to be found that a particular job allocation policy minimized costs. Simulation data (see Figure 4.18) confirmed this.

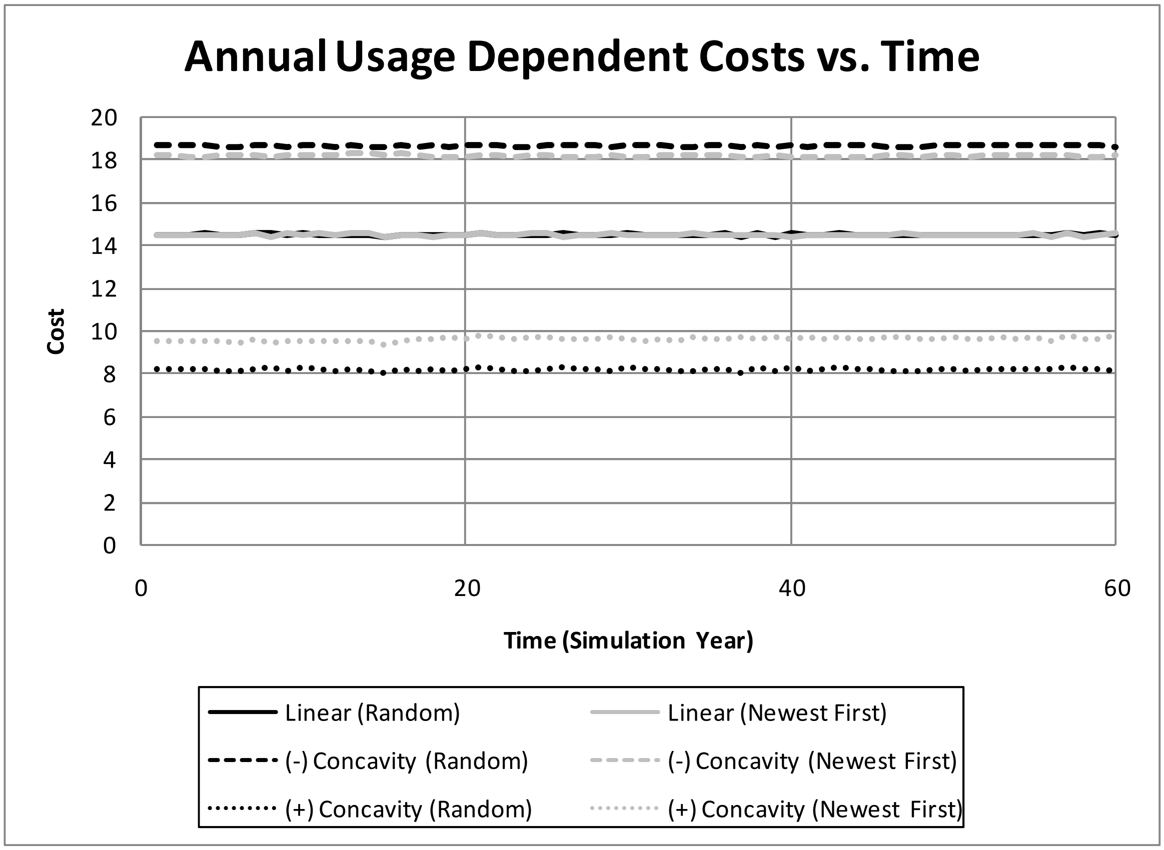


Figure 4.18: Annual usage-dependent costs vs. time in Experiment 1

No significant difference between newest first and random policies was reflected by pool-wide costs when cost as a function of annual usage was linear. When the cost function had negative concavity (decreasing marginal cost), costs were lower under the newest first policy. When the cost function had positive concavity (increasing marginal cost), costs were lower under the random policy.

Cumulative usage-dependent costs displayed a more complicated pattern. As was observed above, plotting simulation data for average cumulative usage versus age resulted in a relatively smooth curve (whether for an individual asset, or pool-wide). Pool-wide cumulative usage-dependent costs also depended upon which assets were active at a given point in time, however. Because assets were replaced, the curve of pool-wide cumulative usage for active assets over time had many sharp breaks (Figure 4.19).

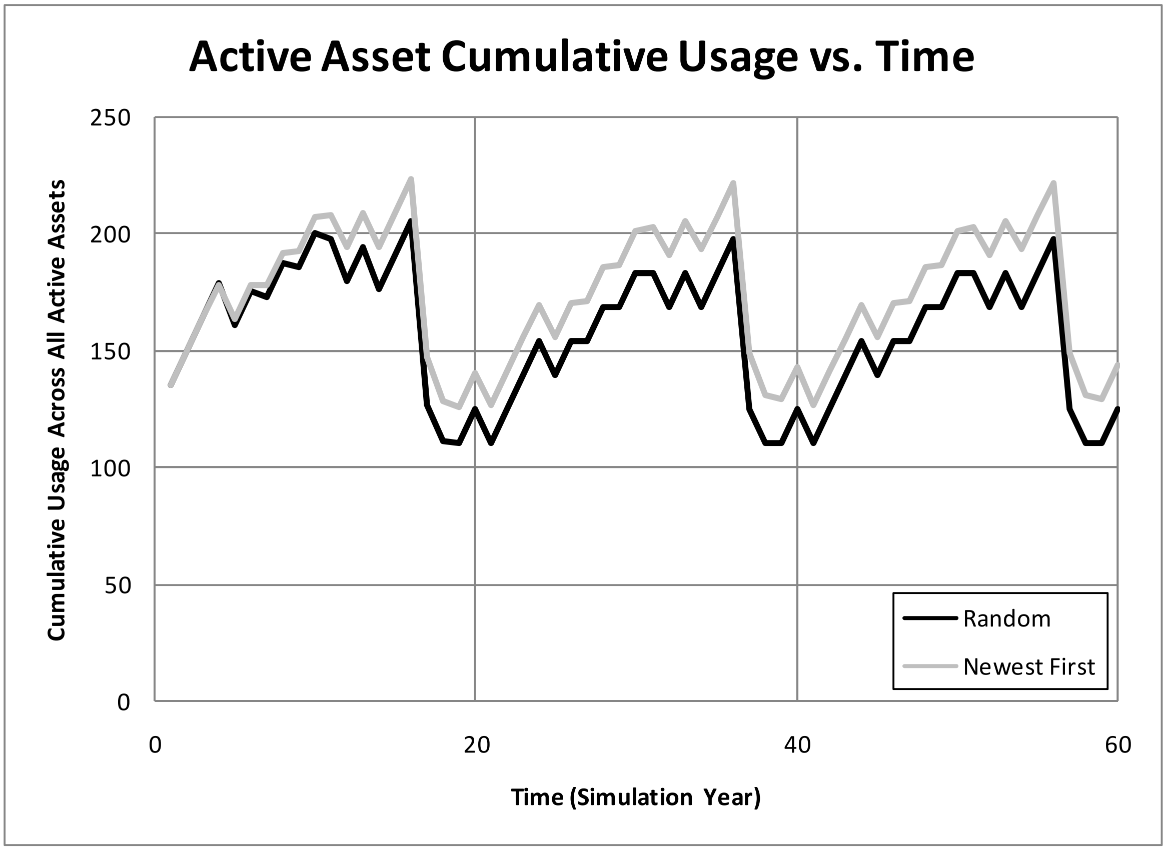


Figure 4.19: Cumulative usage across all active assets vs. time in Experiment 1

In some years, a general pool-wide increase in cumulative usage was counteracted by replacement of an asset having high cumulative usage with one having zero, resulting in a net pool-wide drop. Hence, the curve was also very dependent upon the makeup of the initial pool, since that determined when replacements would occur.

In order to verify that the difference between the curves under random and newest first policies remains the same from stable cycle to stable cycle, a small side experiment (Experiment 1.i) was conducted with all of the same parameters except for an increase in simulation length. Instead of observing three cycles, eleven cycles were observed, resulting in Figure 4.20.

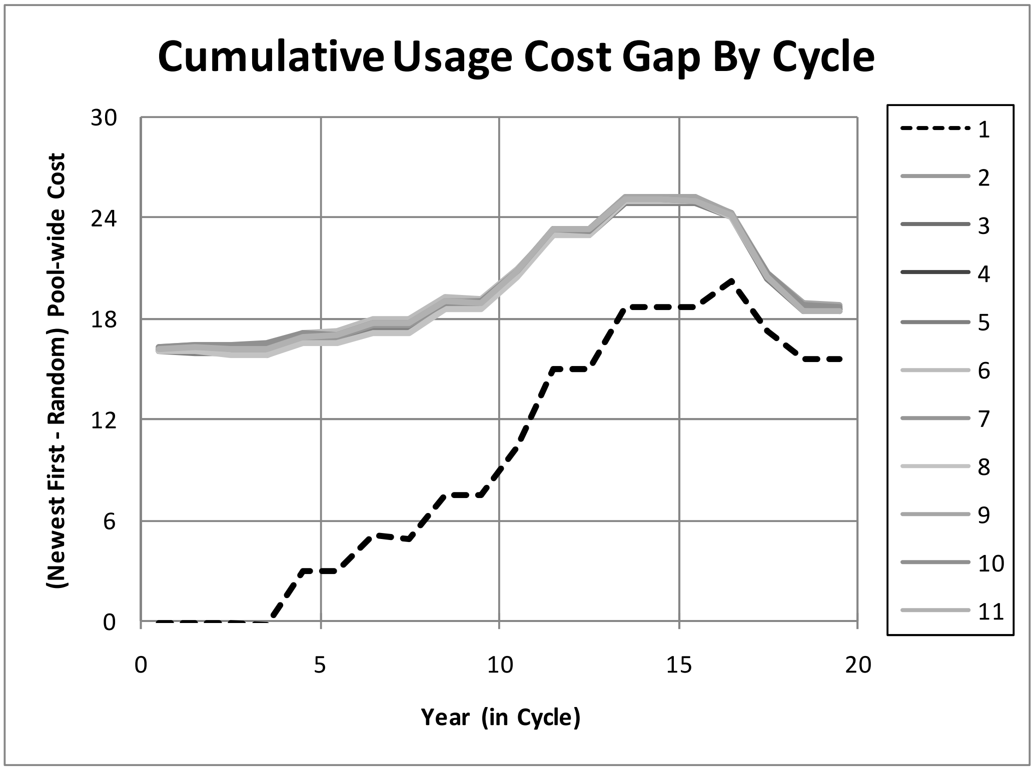


Figure 4.20: Pool-wide cumulative usage cost difference (newest first – random) vs. year in Experiment 1.i

The first cycle was clearly different from the others, as was expected. The initial pool of assets had usage values assigned, and their history of usage (and therefore, cumulative utilization) varied from the other cycles. The remaining cycles displayed only minor variation in the difference of pool-wide cumulative usage under the two policies. This minor variation was most likely due to random job arrivals and service times.

Although three different cumulative usage-dependent cost functions were applied, they did not produce the same variety of results as did the annual usage-dependent costs. The newest first policy resulted in higher costs incurred throughout the stable portion of the simulation (Figure 4.21).

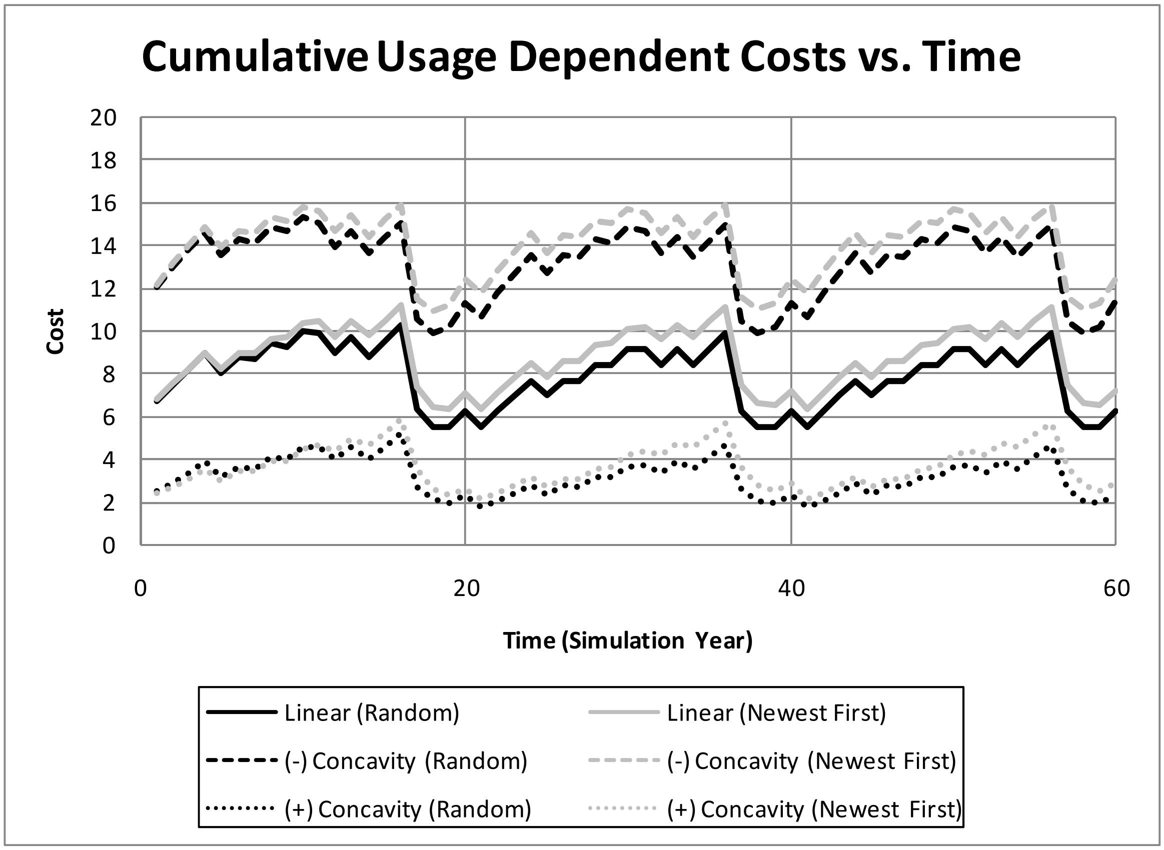


Figure 4.21: Cumulative usage-dependent costs vs. time in Experiment 1

The various cumulative usage-dependent cost functions affected the size of the cost difference between the two policies. The cost difference also peaked at different points within the cycle based on the cost function. Nevertheless, these effects were relatively minor. The only other possible significant difference between the cost functions was found in the first cycle, where the cost function with positive concavity (marginally increasing cost) resulted in several years where the newest first policy was cheaper than the random policy. Even in this case, over the course of the entire first cycle, the newest first policy remained more costly.

4.4.2 Experiment 2: Variation of Fixed Replacement Interval

4.4.2.1 Costs prior to discounting

After observing different types of costs for one fixed replacement age, questions arise regarding how varying the replacement age will affect these costs relative to each other. First, undiscounted costs (costs discounted at a rate of 0%) were observed for age, annual usage, and cumulative usage-dependent costs.

As the average asset age across the pool increased with a longer replacement age, age-dependent costs increased in a manner similar to how they might increase for an individual asset. Cost curves followed the shape of their cost functions (Figure 4.22).

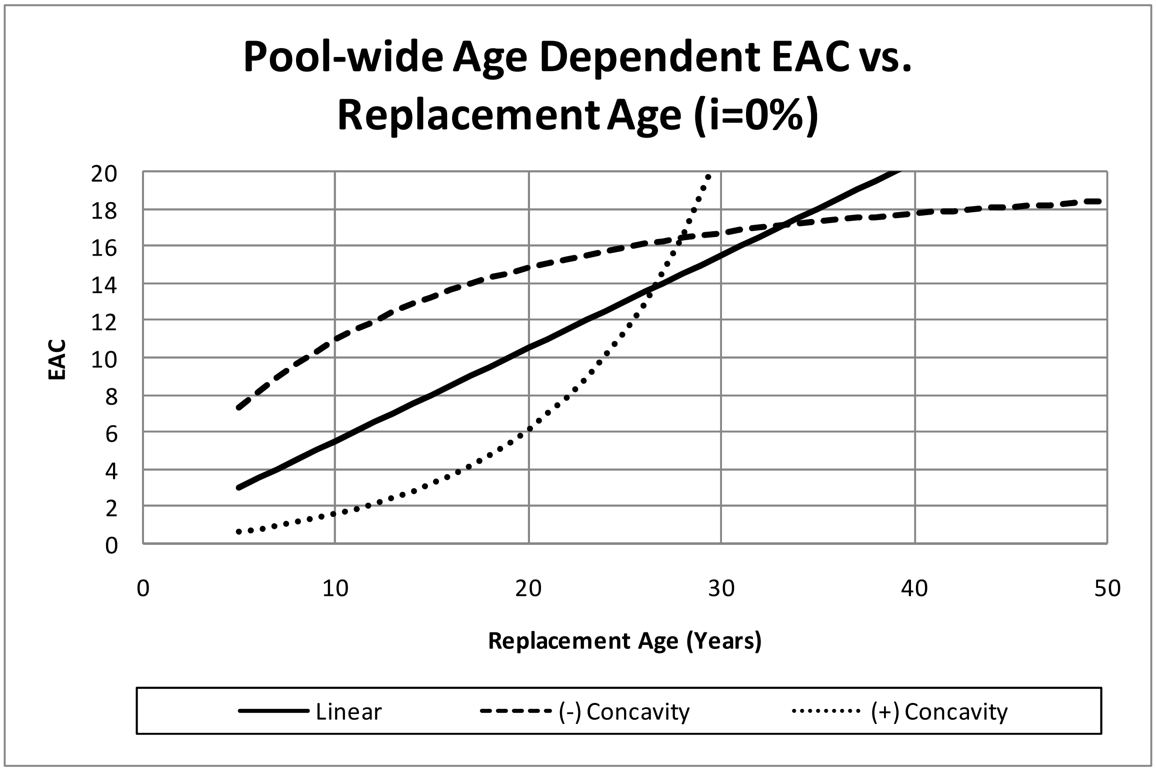


Figure 4.22: Pool-wide age-dependent EAC vs. replacement age (i=0%) in Experiment 2

Age-dependent cost curves were identical under random and newest first job allocation policies. This was logical; the primary effect of different job allocation policies is on usage, leaving age unaffected.

Annual usage-dependent costs experienced some small shifts as replacement age was varied but were generally quite constant (Figure 4.23). These small shifts were most noticeable for lesser replacement ages. Varying replacement age did not contradict the hypotheses regarding costs generated through analysis for a single replacement age. Newest first remained the more cost-effective policy for a cost function of negative concavity, and random remained the more cost-effective policy for a cost function of positive concavity. The policies remained equally beneficial where the cost function was linear.

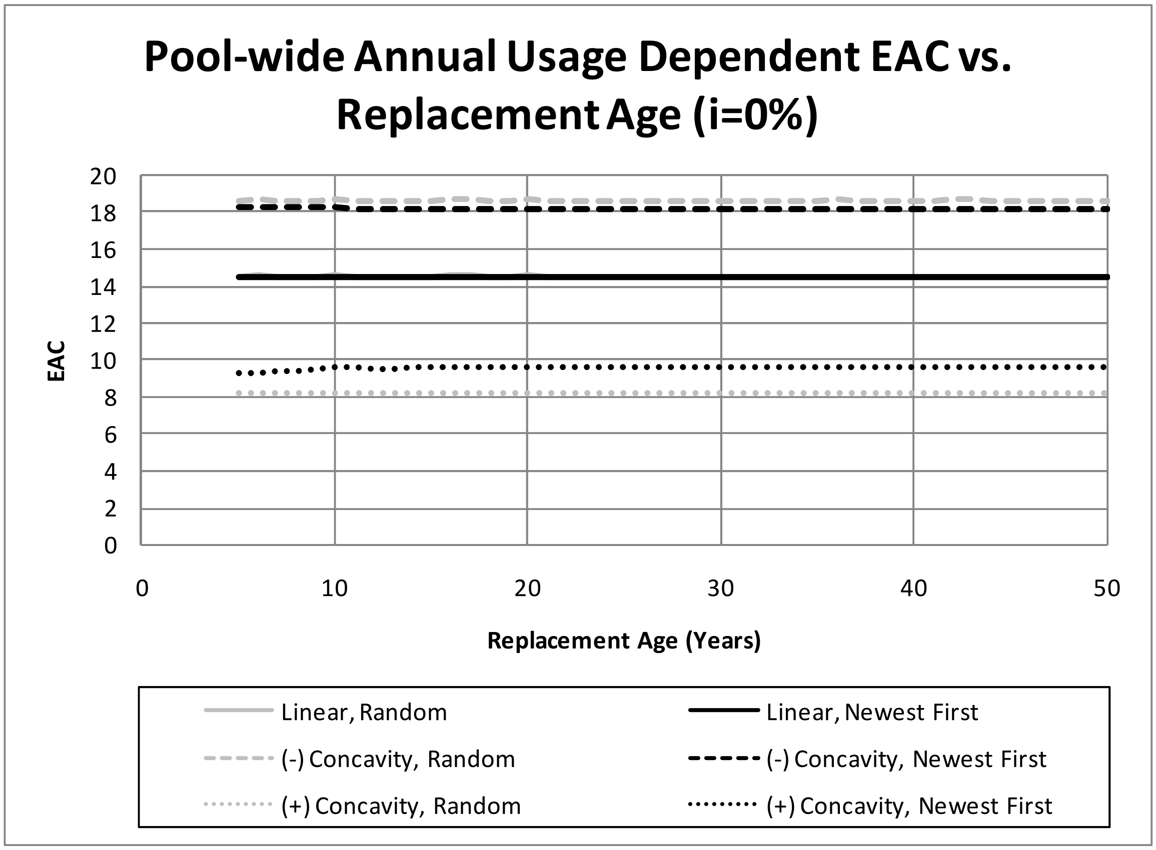


Figure 4.23: Pool-wide annual usage-dependent EAC vs. replacement age (i=0%) in Experiment 2

Annual pool-wide demand remains constant regardless of the replacement age, which provides insight into why the annual usage-dependent curves remained relatively constant. One possible explanation for the small shifts in the curves may be due to the fact that the same initial pool was used for all simulations. This effect will be further discussed in Section 4.4.2.2.

In contrast to annual usage-dependent costs, a clear relationship between cumulative usage costs and replacement age was observed. As an asset is kept for a longer period of time, it has the chance to accumulate additional usage. The curves in Figure 4.24 are similar in shape to their respective cost functions, as was the case for age-dependent costs.

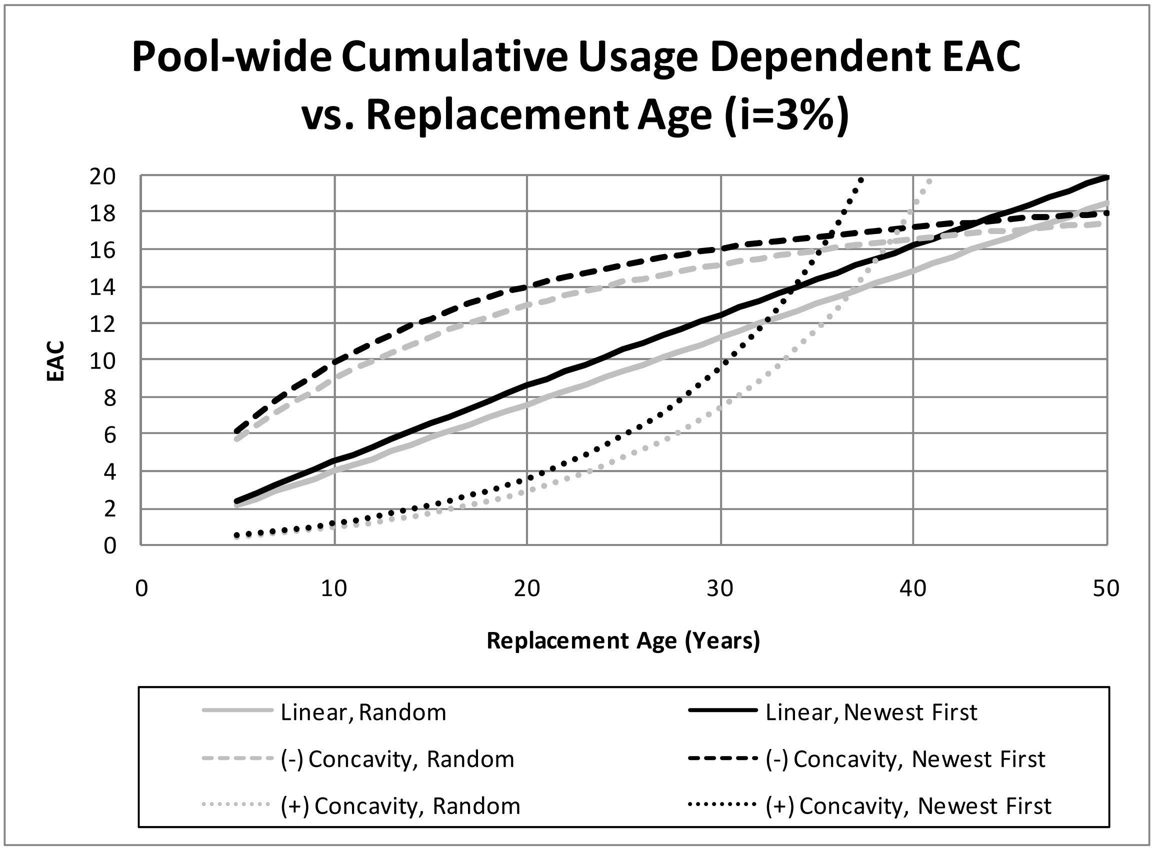


Figure 4.24: Pool-wide cumulative usage-dependent EAC vs. replacement age (i=0%) in Experiment 2

Unlike age-dependent costs, however, there was a significant difference between the two job allocation policies for cumulative usage-dependent costs. The gap between costs under the two policies widened as replacement age increased for the linear cost curve and the cost curve with positive concavity. For the cost curve with negative concavity, the cost difference between the two job allocation policies peaked around a replacement age of fifteen years in this case, then gradually declined. A longer lifetime puts an asset at a higher cumulative usage along the curve, where the marginal cost is smaller; hence the difference between policies is less pronounced for the cost curve with negative concavity.

4.4.2.2 Experiment 2.i: sensitivity to initial pool makeup

As can be observed from Figure 4.25, the initial pool of assets used in Experiment 2 resulted in distorted usage versus age curves for replacement ages greater than twenty years in length. Since no initial asset was older than twenty years, there was a long period within the fifty year cycle where no replacements were made. Without any replacements, the newest first curve began to resemble the random curve throughout this period.

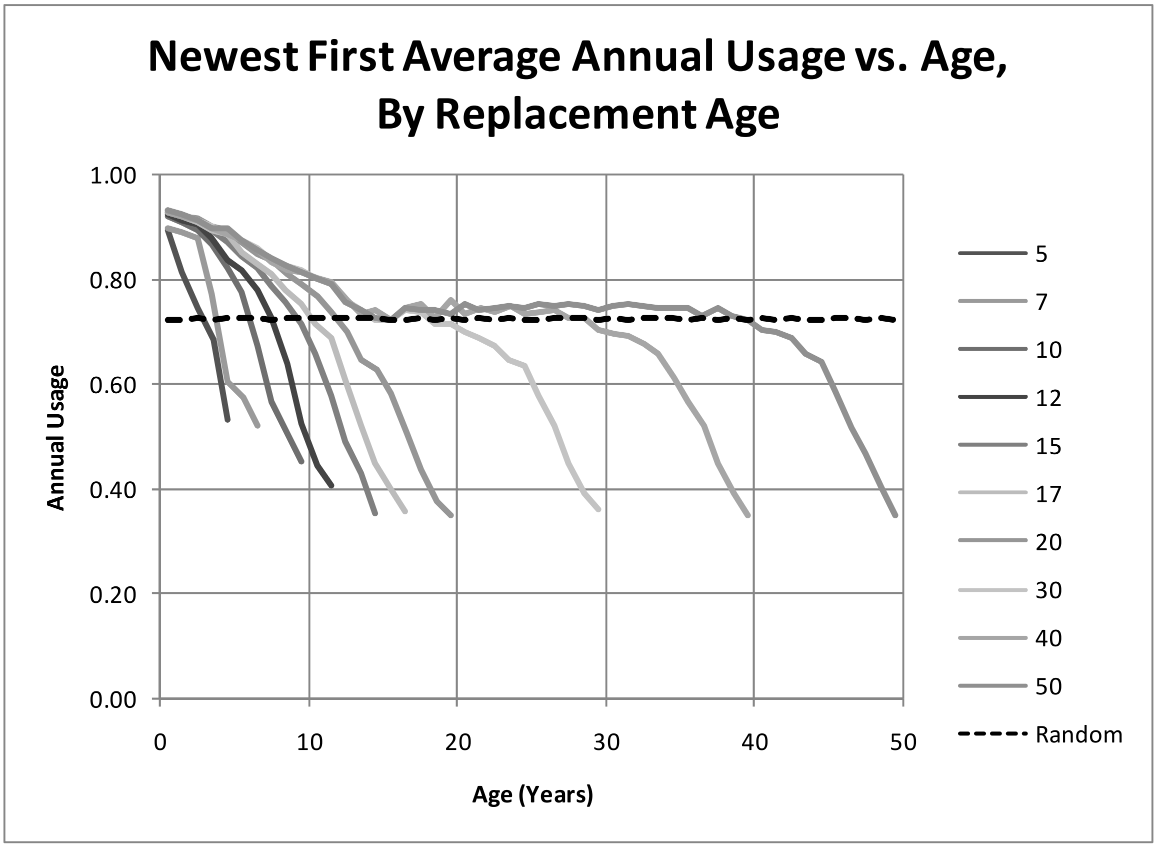


Figure 4.25: Newest first average annual usage vs. age, by replacement age in Experiment 2

A slightly modified version of the experiment (referred to here as Experiment 2.i) was performed to test the impact of this effect on costs. A new initial pool was created for each replacement age, with asset ages evenly distributed. This included fractional years in many cases. Ages with fractional years were eliminated as the initial assets were replaced, since replacements were made only once per year. For simplicity, the initial cumulative usage was set to zero for all initial assets regardless of age. Since only data from stable cycles were used, cumulative usage of the initial assets did not affect the costs under investigation. Figure 4.26 shows usage versus age curves for Experiment 2.i.

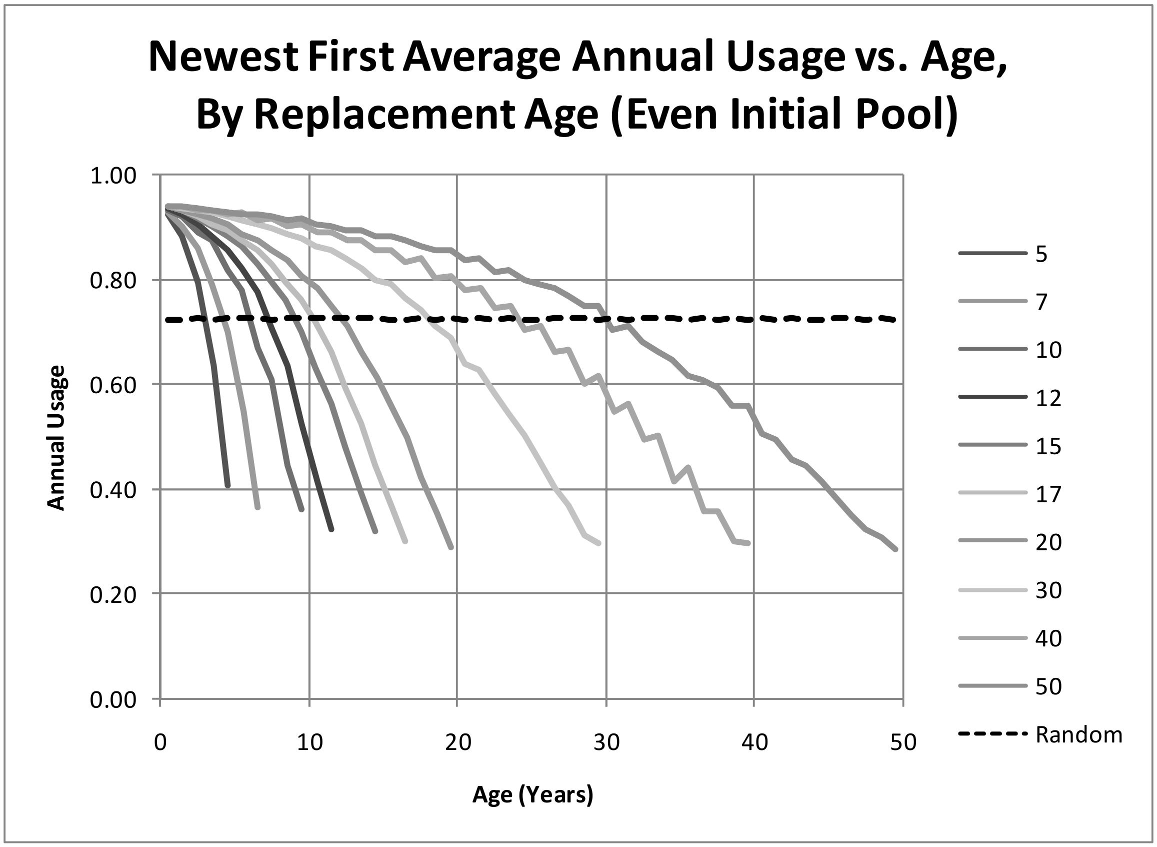


Figure 4.26: Newest first average annual usage vs. age, by replacement age in Experiment 2.i

Analysis of the cost curves from this revised experiment showed them to be very close to the results for the less balanced initial pool of assets, both in shape and values. The shift in equivalent annual cost (EAC) for annual usage-dependent costs at low replacement ages was no longer significant once ages were distributed more evenly throughout the pool.

4.4.2.3 Discounted costs and finding the minimum total EAC

As previously mentioned, the most common approach to replacement analysis has been to plot the total EAC versus replacement age. The optimal replacement age is selected where the EAC curve is at its minimum. The EAC curve is expected to form a “U” shape, as capital EAC decreases and maintenance EAC increases over time. Figure 4.27 demonstrates that the capital portion of the total EAC is consistent with these assumptions.

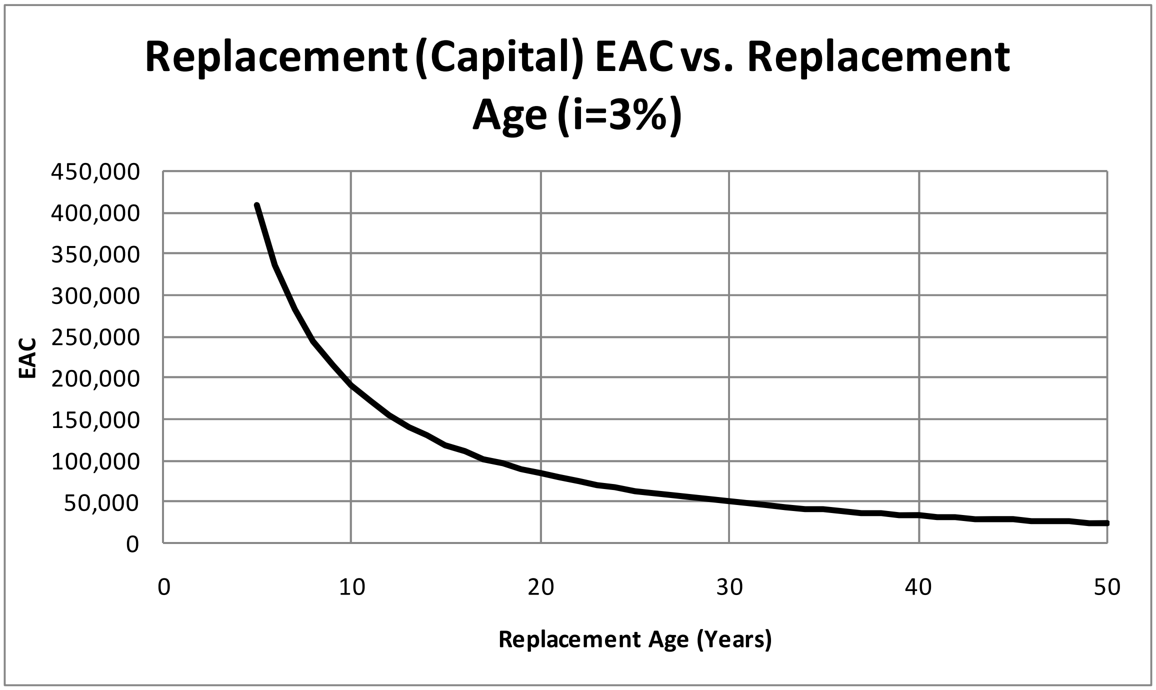


Figure 4.27: Replacement (capital) EAC vs. replacement age (i=3%) in Experiment 2

Most maintenance costs will end up decreasing for large replacement ages. Annual usage-dependent cost curves are essentially constant with respect to replacement age, prior to any discounting effect. Once a discounting effect is applied, they decrease with increasing replacement age. Age and cumulative usage-dependent cost curves with negative concavity approach a horizontal asymptote, so they level off to a nearly constant value for large replacement intervals. Applying a discounting effect transforms these curves so that they peak and then decline as replacement age is increased. Linear age and cumulative usage-dependent costs do not level off, but an exponential discounting function dominates over the linearly increasing function as replacement ages grow large. These curves also peak and then decrease.

Only two of the identified cost functions do not end up decreasing with respect to replacement age if discounting is applied: age-dependent and cumulative usage-dependent cost functions having positive concavity. Unless these make up the dominant costs, the total EAC curve decreases continuously for large replacement intervals. This does not result in an optimal replacement age. Instead, it suggests that assets should be kept indefinitely.

4.4.3 Using Single Replacement Age Data to Find Optimal Replacement Age

In a practical situation, it may be tempting to take a shortcut approach to finding the optimal replacement age for a pool of identical assets. If past usage/cost data are already available for a pool of assets, finding the optimal replacement age as the minimum of the EAC versus age curve for the average asset may seem feasible. This approach is misguided, however, unless the entire pool is purchased and replaced as a single group.

First, consider age-dependent costs. When looking at a pool, the sum of the ages of individual assets remains relatively constant from year to year (the degree of variation depends upon how the ages of the initial assets are distributed). In a situation analogous to the individual asset case – where all of the assets are purchased at the same time – the assets age together. The sum of their individual ages increases each year, until the replacement age is reached and all the assets are replaced. In both cases, each asset (along with its immediate predecessor or successor, perhaps) will spend exactly one year at each age, over the course of a cycle with length equal to the replacement age. Adding each year's pool-wide age total, across all the years of one cycle, will give the same sum in either case. With no discounting, both cases result in an equal EAC. However, if a discounting factor is applied, the pool with varied ages will have a greater EAC. The pool with uniform age has the majority of its cost in the discounted future, while costs for the pool with varied ages are more evenly distributed over time.

Annual usage-dependent costs are affected in a different manner. If average individual asset data are used to calculate the EAC for a replacement age less than the actual replacement age under which the data were obtained, the latter portion of that asset's usage versus age curve is effectively truncated. If a newest first job allocation policy is used, this will result in the total usage of the pool exceeding demand! Understandably, this will increase annual usage-dependent costs when compared to costs calculated at the appropriate level of usage. As a greater portion of the usage versus age curve is truncated, the degree of error will increase. Costs that are dependent upon cumulative usage will be affected by both of the factors described above.

To demonstrate the difference between these two methods of finding the optimal pool-wide replacement age, an example case has been created. Data for the individual asset case were obtained from Experiment 2.i., with a replacement age of fifty years. Costs for all assets in the stable cycles were averaged against age, and multiplied by twenty (the pool size) for direct comparison to pool-wide values.

A linear combination of multiple cost categories was taken to calculate a total cost. An annual usage-dependent cost function with positive concavity was given a relative weighting of ten, and a cumulative usage-dependent cost function with positive concavity was given a relative weighting of two. Capital costs were given a relative weighting of fifteen, and a discounting rate of 3% was applied. The results are shown in Figure 4.28.

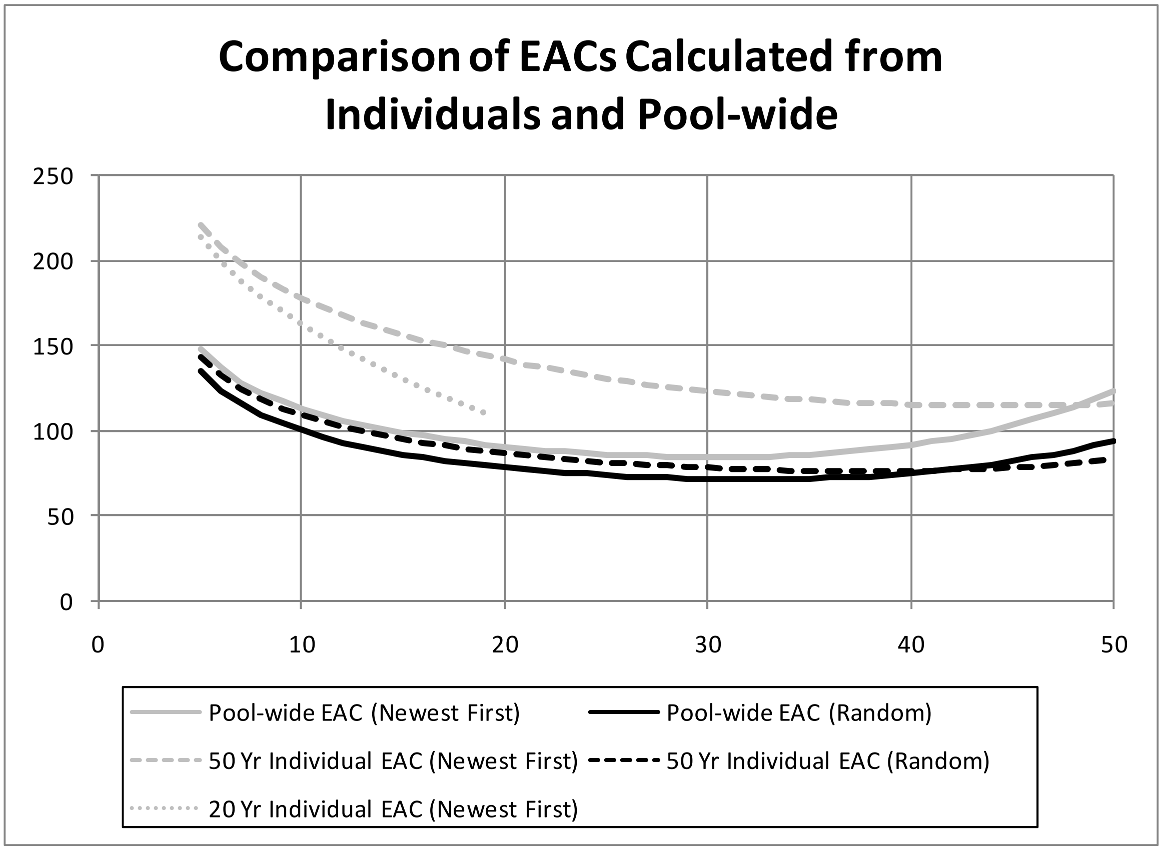


Figure 4.28: Comparison of EACs calculated from average individual asset and pool-wide using Experiment 2.i data

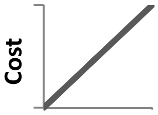
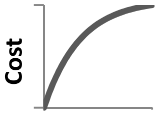
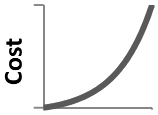
Analysis of the average individual asset resulted in optimal replacement ages of 37 and 44 years under random and newest first policies, respectively. Pool-wide analysis resulted in optimal replacement ages of 33 and 30 years for random and newest first policies, respectively. The two curves under the random policy remained relatively close, although they still resulted in significantly different optimal replacement ages. Under the newest first policy, the difference in cost between the two calculation methods was very large.

It may also be observed from Figure 4.28 that the optimal replacement age (calculated from average individual asset data) varied significantly based on the replacement history. A fifth curve was included in the plot, using average individual asset costs from a simulation with a 20-year replacement age. This curve diverged from the individual asset curve with a fifty year replacement age. If data that had been collected under a replacement age of twenty years were used, it would not be at all clear what the optimal replacement age should be. Extrapolating the curve above might even suggest that keeping assets indefinitely would minimize total costs.

4.5 CONCLUSIONS

Hypotheses were generated regarding the differences for a variety of cost types under two job allocation policies: random and newest first. These hypotheses assumed a fixed replacement age and were based upon hypothesized differences in asset usage versus age curves. They also assumed a stable replacement cycle, eliminating the otherwise unpredictable effects of initial state variable values for the initial pool of assets. Results of statistical simulation confirmed these hypotheses for each of the cost types analyzed. The findings are summarized in Table 4.2.

Table 4.2: Minimum cost job allocation policies by cost category under fixed-age replacement

	Cost Curve Shape, with Respect to Cost Category		
Cost Category	 (Linear)	 (Negative Concavity)	 (Positive Concavity)
Constant*	Either		
Age-Dependent	Either	Either	Either
Annual Usage-Dependent	Either	Newest First	Random
Cumulative Usage-Dependent	Random	Random	Random

*Cost curve shapes are not applicable to the constant cost category

If the total annual cost for an asset is generated from a linear combination of more than one of the cost categories described above, the conclusions from Table 4.2 can be used to find the optimal job allocation policy qualitatively, as summarized in Table 4.3. For example, consider a pool of assets where total annual maintenance costs have been broken down into two categories (e.g., fuel costs and repair costs) – costs A and B, respectively. If cost A is an annual usage-dependent cost with positive concavity and cost B is a cumulative usage-dependent cost with positive concavity, Table 4.2 indicates that the random allocation policy would be preferable in order to minimize both cost A and cost B. By looking up the intersection of optimal policies (random and random) for costs A and B, respectively, in Table 4.3, it can be found that the random allocation policy would also minimize total annual maintenance costs.

On the other hand, if cost A had been annual usage-dependent with negative concavity, Table 4.3 would tell us that an answer could not be found qualitatively (the optimal allocation policy would depend upon the particular parameters). If more than two types of costs are linearly combined to obtain the total cost, Table 4.3 can be used iteratively by taking two costs at a time. It was shown that statistical simulation could be used to obtain quantitative results demonstrating the difference between allocation policies.

Table 4.3: Minimum cost job allocation policies for a linear combination of cost categories

		Optimal Policy for Cost A		
		Newest First	**Random**	**Either**
Optimal Policy for Cost B	**Newest First**	Newest First	Depends upon parameters	Newest First
	Random	Depends upon parameters	Random	Random
	Either	Newest First	Random	Either

The differences in pool-wide costs between job allocation policies were also investigated as the replacement age was varied. Due to the complexity of these costs, statistical simulation was used to observe the results.

For each cost category identified in Table 4.2, the optimal allocation policy remained the same regardless of replacement age. Annual usage-dependent costs with negative concavity were the only category where the newest first allocation policy proved to be most cost-effective. In all other cases, the random policy was preferable or the two policies resulted in equal costs. Unless annual usage-dependent costs with negative concavity dominate a particular linear combination total cost, the random allocation policy will always result in lesser total costs. It follows that the optimal replacement age will be smaller under the newest first policy than it would be under the random policy unless annual usage-dependent costs with negative concavity are dominant.

The magnitude of the various costs changed in differing ways with respect to replacement age. All annual usage-dependent costs remained relatively constant, regardless of replacement age. Age and cumulative usage-dependent costs varied in a manner that was representative of the shape of their cost function curves. Furthermore, the difference in costs between allocation policies increased with respect to replacement age for cumulative usage-dependent costs where the cost function was linear or of positive concavity. The difference in costs narrowed with respect to replacement age for cumulative usage-dependent costs where the cost function had negative concavity.

Discounting some cost categories was found to result in unlimited economic life for an asset. Of the cost categories described in Table 4.2, only two did not result in costs that eventually began to decline with respect to replacement age after discounting: age-dependent and cumulative usage-dependent costs with positive concavity.

Finally, the method used in this study to find an optimal replacement age was contrasted with another approach. In this alternate approach, average asset cost data were taken from a pool with an established replacement policy. Those data were used to calculate an EAC versus age curve, which was then used to find an optimal replacement age. This alternate approach was shown to be incorrect. Age-dependent costs within a pool of assets of varying age behave differently when discounted than age-dependent costs for an individual asset. Usage-dependent costs also behave differently; since asset usage within the pool is constrained to meet demand, asset usage is interdependent within the pool. Because of these differences, using average data for equipment in a pool to compute an optimal replacement age is incorrect.

There are many opportunities for further investigation based upon the results of this study. Additional sensitivity analysis could be conducted on the effects of the makeup of the initial pool on costs under different allocation policies. Random and newest first job allocation policies could be evaluated under other replacement policies. In particular, analysis of costs under a budget-constrained replacement policy would be valuable, since many organizations operate under internal or external capital constraints. Due to the complexity of calculations in such a situation, statistical simulation would likely be a valuable tool. Finally, valuable research might be conducted on the best manner of fitting actual cost data to the cost categories described in Table 4.2.

5.0 COMPARISON OF OPTIMIZATION MODELS FOR EQUIPMENT REPLACEMENT AND SIMPLE EQUIPMENT PRIORITIZATION MODELS USING REAL FLEET DATA

This section documents a first step to quantitatively compare one of the advanced replacement models described in the academic literature with the age/use standard model (similar to that currently used at ODOT) using simulation. A cost model was created for the simulation based on ODOT data (from 1994-2002). Performance was analyzed within the context of minimizing fleet-wide costs over a finite planning horizon. The specific advanced model to be used is based on that described by Hartman (*2004*). This model uses a dynamic programming approach and assumes individual asset utilization levels are determined by the model.

5.1 GENERAL APPROACH

There were three general components of this research:

1. Cost model based on ODOT data that will be used to reflect realistic costs of a fleet in a simulation;

2. Discrete event simulation of vehicle use within a fleet that will be used to simulate the application of advanced replacement models described in the academic literature, and also the age/use standards models currently applied by many DOTs; and

3. Analysis of simulation performance to compare the different replacement analysis approaches.

The cost model (described in this section) was based on work by Kriett (*2009*), but was extended to allow for different usage patterns. Bethuyne (*1998*) has shown that altering utilization patterns can affect the economic life of an asset, so it was important to incorporate this ability into the cost model. The cost model was constructed using multiple regression. Explanatory variables such as age, annual usage, lifetime usage, and previous year's cost were considered in predicting the mean annual cost for an individual vehicle, as well as the variation. Models were created for repair costs and operating (e.g., fuel) costs. Since many of the potential explanatory variables are not applicable in the first year of a vehicle's life, separate models were created for year one and for all remaining years. The class of vehicles analyzed was heavy diesel trucks, due to the large amount of data available and the significance to ODOT's operations.

Hence, the numerical results of simulation were only applicable to this subset of the fleet. However, the general principles observed through simulation will likely be applicable fleet-wide. In the simulation, these cost models define the mean and variance of normally distributed costs as a function of other variables (e.g., age, cumulative use, etc.). Randomly generated observations from this normal distribution were used in the simulation model.

The discrete event simulation was written in Visual Basic. Jobs arrive into the system, which require the use of a vehicle. The choice of vehicle is determined based on the replacement model. The vehicle remains in use for some period of simulation time, and is then returned and made available for future use. Periodically, the fleet of vehicles is analyzed and several vehicles are replaced. At the end of each simulation year, costs for the previous year are assigned to each vehicle using the cost model described above. The cost model generates a mean response, which is transformed into a single observation according to an appropriate normal distribution. All data were stored in a Microsoft Access database for later analysis.

After simulation data were collected, total fleet-wide cost over the planning horizon was calculated, taking into consideration the time-value of money. Fleet-wide cost between the two model types was compared using a paired t-test, with each observation consisting of one simulation replication. Several alternatives within each model type could also be compared.

5.2 COST MODEL

Simple linear regression is used when a response variable (e.g., repair cost) is compared against a single explanatory variable (e.g., lifetime mileage). The relationship between the two variables can be represented as a line predicting the average response for each value of the explanatory variable. Actual observations of the response variable do not necessarily fall on the line, but the line forms a center point about which they vary randomly according to a normal distribution. A linear model is very useful because average response values can be interpolated between initial data points. For example, if the original data included repair costs at 100,000 miles and 150,000 miles, but an estimate for the average repair cost at 125,000 miles was desired, a simple linear regression model would allow us to easily calculate this.

With the ODOT data, more than one variable can reasonably be expected to affect the repair or operating cost in a given year. For example, repair costs might depend upon age, lifetime mileage, and the previous year's repair costs. In such a case, multiple regression is used. Multiple regression takes the simple linear regression model and extends it to include more than one explanatory variable.

One of the assumptions required when using multiple regression is that the variance of the response is constant for all values of the explanatory variables. For example, cost values should not vary more about their average at high usage levels than at low usage levels. The variance was not equal in the original ODOT data, so a logarithm (log) transformation was applied. In turn, this also required log-transforming the explanatory variables in order to maintain a linear relationship.

Selecting the appropriate explanatory variables is a major part of constructing a multiple regression model. Adding more explanatory variables will always improve the model at least slightly, but additional variables are not always meaningful. The best model incorporates the fewest explanatory variables that explain the response. For repair costs, the following explanatory variables were considered: age, annual usage, lifetime usage, and previous year's repair costs. For operating costs, the following explanatory variables were considered: age, annual usage, and lifetime usage. The previous year's operating costs were very closely correlated with the previous year's annual usage, so that variable was excluded from consideration, since the year-to-year usage patterns might change in the simulation.

Since a relatively low number of variables were under consideration, every possible combination was tested. Adjusted R^2 values were compared to gauge which model best explained the costs. Any model that failed at least one t-test at the $\alpha = 0.05$ level for significance of an individual term was not considered. These R^2 values were in the range of 0.5 to 0.8 for selected models. This reflected the significant variability inherent in the data, but did not raise any issues with the validity of the simulation. Since the regression model simply established a probability distribution from which observations are drawn, a realistic understanding of the variability in these costs enabled the simulation to generate data similar to that observed in reality. Personal judgment was used to select the model using fewer explanatory variables if results were very close. The resulting models for repair and operating costs were the following:

Avg. Repair Cost = $e^{-1.51277}$ * (Annual Usage)$^{0.24688}$ * (Lifetime Usage)$^{0.52312}$ * (Prev. Year's Repair Costs)$^{0.25262}$ (Adjusted R^2 = 0.5422)

Avg. Operating Cost = $e^{-3.48313}$ * (Annual Usage)$^{0.80964}$ * (Lifetime Usage)$^{0.34302}$ (Adjusted R^2 = 0.7969)

Please note that the above models applied only to vehicles of age two years and greater, since not all of the variables applied in year one. For example, in year one, annual usage and lifetime usage are equal. Separate models were constructed for costs in year one – a simple normal distribution for repair costs and a regression model for operating costs with annual usage as the sole explanatory variable.

Analysis of the initial cost model demonstrated the need to refine it with additional ODOT data. The source data covered a period of 9 years, but some vehicles' lives started prior to the period for which data were available. This means that lifetime usage data were not available for vehicles older than 9 years, which had a highly significant impact on the cost model. Annual usage tends to be fairly stable for the first 10-12 years of a vehicle's life, and then begins to rapidly diminish (Figure 5.1). Without including data beyond year 9, the model does not include this crucial behavior. Correcting the model will be easy once additional data are obtained, since the same methodology as before will be followed.

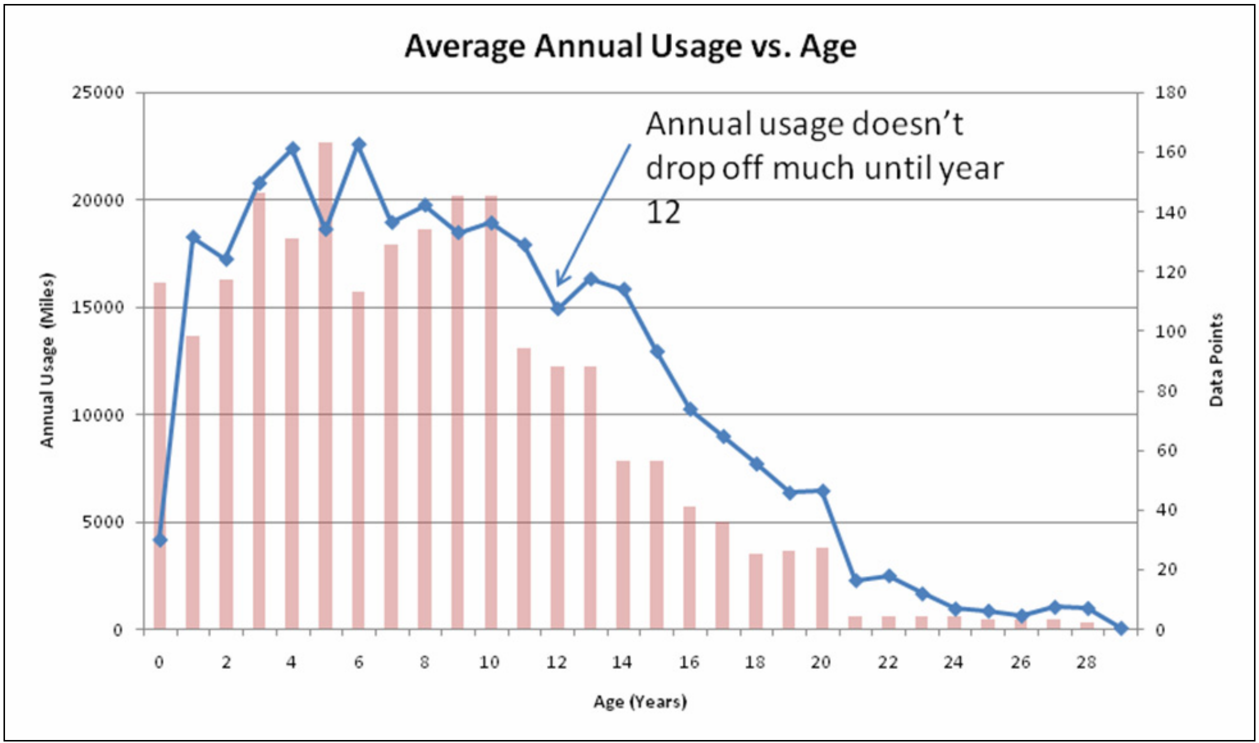


Figure 5.1: Average annual usage vs. age for ODOT heavy diesel trucks

5.3 SIMULATION

Simulation of an arrival-based vehicle use process was completed, although further tuning to reflect the usage patterns of the ODOT fleet will be necessary Figure 5.2 shows a matrix of different plots that show how a specific data category varies as a function of another category. The labels in the diagonals show the different types of data categories considered. For any individual graph in Figure 5.2, the data label on the diagonal matrix element in the same row is the row index for the graph, and the data label on the diagonal matrix element in the same column is the column index for the graph. Ideally the data relationships shown on the right of Figure 5.2 (generated by the simulation) should look the same as those on the left of Figure 5.2 (generated by ODOT data).. The simulation in its current state of development does not include enough variation in year-to-year vehicle usage, and usage currently drops off too quickly with age. Performance may be improved by adjusting the time between job arrivals, duration of jobs, and number of vehicles in the simulated fleet.

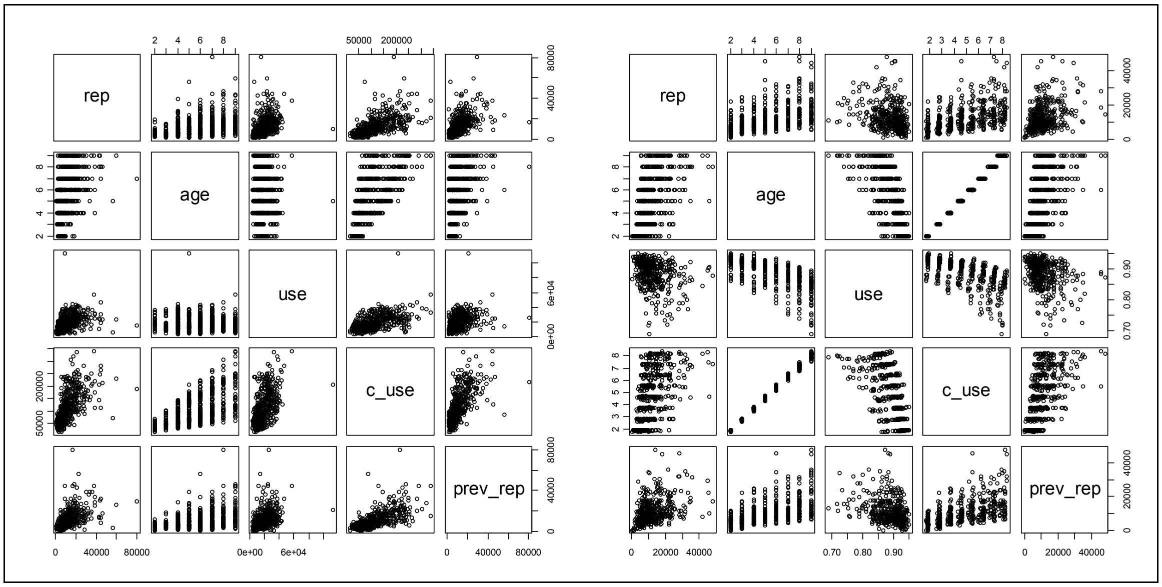


Figure 5.2: ODOT data on left, preliminary simulation data on right. "Rep" refers to repair costs, "use" refers to annual usage measured in miles, "c_use" refers to lifetime usage measured in miles, and "prev_rep" refers to repair costs in the previous year.

5.4 NEXT STEPS

The next steps in this research are to collect more data and refine the cost models so that both sides of Figure 5.2 "look the same". If data collected from the simulation (on the right) appear very similar to actual data collected by ODOT (on the left), that will be strong evidence that the simulation can successfully model vehicle use and replacement. Currently, the ODOT data is limited to showing the first nine years of life for heavy diesel trucks, which can differ substantially from the following twenty years or beyond. For example, annual usage begins to fall significantly, and it is important that the simulation reflects this drop and is compared to actual data for model validation. After this is completed the implementation of the Hartman (*2004*) model within the simulation will begin. The simulation that utilizes age/use standards as a vehicle replacement method has been fully modeled at this point. The dynamic programming alternative has not yet been modeled within the simulation. This will then be followed by experimentation using the simulation model.

It may be useful for ODOT to periodically examine cost data with respect to the explanatory variables – perhaps on a five year cycle. The primary goal would be to detect any significant changes to the relationships between variables, which might alter the assumptions of the simulation model. A matrix of scatter plots, similar to that used on the left in Figure , is a useful tool for this purpose.

6.0 CONDITION MODEL DATABASE STRUCTURE

The development of a new fleet replacement model or "condition model"[1] was conducted in several phases, as depicted in Figure 6.1. First, several ODOT Fleet Services personnel were interviewed to identify a set of key reports that the condition model needed to produce. Once these reports were identified, an Entity Relationship (ER) model was developed based on this information. Microsoft (MS) Access 2003 was the platform of choice to develop the condition model, per request of ODOT Fleet Services personnel.

As Figure 6.1 illustrates, the data that the condition model utilizes to produce its output comes from two main external systems: ODOT's equipment management system (EMS) and ODOT's financial system known as TEAMS. The final step in the design process was to carefully select the necessary fields from a series of input files to avoid data redundancy.

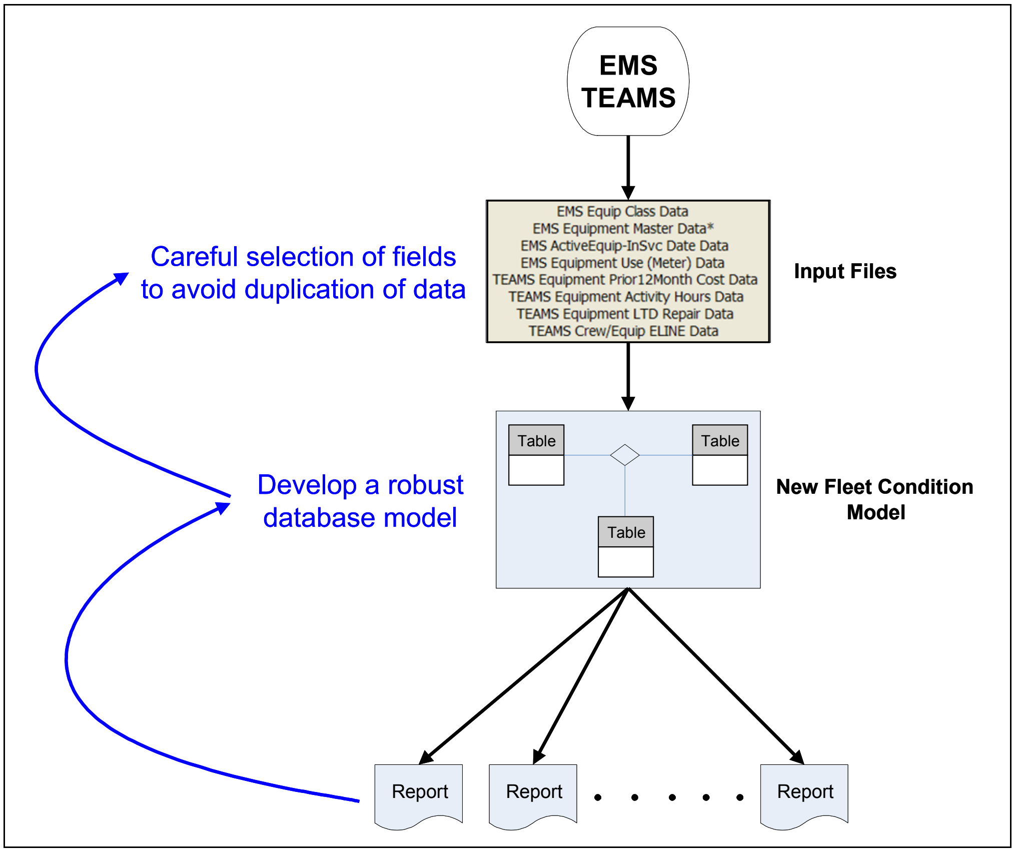

EMS
TEAMS
EMS Equip Class Data
EMS Equipment Master Data*
EMS ActiveEquip-InSvc Date Data
EMS Equipment Use (Meter) Data
TEAMS Equipment Prior12Month Cost Data
TEAMS Equipment Activity Hours Data
TEAMS Equipment LTD Repair Data
TEAMS Crew/Equip ELINE Data
Input Files
Careful selection of fields to avoid duplication of data
Table
Table
Table
New Fleet Condition Model
Develop a robust database model
Report
Report
Report

Figure 6.1: Design approach to develop new condition model

[1] The fleet replacement model is referred to as "condition model" internally by ODOT Fleet Services personnel. For consistency, the latter term will be used for the remainder of the document.

6.1 INPUT DATA FILES

As shown in Figure 6.1, the data that the condition model utilizes to produce its output comes from two main external systems: the equipment management system (EMS) and the financial system known as TEAMS. These data were organized in a series of input files that are then fed into the condition model. The data from the input files is used by the condition model to create a normalized database in MS Access 2003. It is critical that the structure of the different input files remain the same always (i.e., number, name and format of data fields) to ensure the accuracy of the condition model.

The condition model also uses data from other files to perform its function. These additional data comes from either an external file (see Section 6.1.2) or from static files (i.e., tables) already populated in the condition model. The following sections of this document describe these data sources in more detail.

6.1.1 Periodic Input Files

Periodic input files refer to the source data files obtained from TEAMS and EMS. These files are periodic in that they must be replaced with the most recent versions (or with files from the biennium of interest) before the condition model is run. Table 6.1 provides a list of the periodic input files and their descriptions. The original and new names are provided where a change in name has occurred. The model uses the most current file names.

Table 6.1: Description of periodic input data files

Original file name	New file name	Description
CECLASS.TXT		**List of equipment class descriptions**
DOWN41.xls		***Not used in model***
DOWN45.xls		**Equipment in service dates**
EQMSTR.xls		**Equipment master file**
Excel_KRL_E21666.csv	Excel_WLL_E25580.csv	**Prior 12 month equipment operation and repair costs**
Excel_KRL_E21667.csv	Excel_WLL_E25579.csv	**Life to date repair costs**
HPEM101M.xls		***Not used in model***
HPEMS37R.xls		**Eline report showing which appropriation crews are in**
HPEMS62R.xls		**Prior 24 month equipment usage report (EMSUse24)**
Usageactivityhrs-conditionmodel*PreviousBiennium*update.txt	*** (see note below)**	**TEAMS prior biennium activity hours**
Usageactivityhrs-conditionmodel*CurrentBiennium*-update.txt	**** (see note below)**	**TEAMS current biennium activity hours**

Note: For the current biennium (07-09), the “TEAMS activity hours” file names are:
*** = Usageactivityhrs-conditionmodel05-07update.txt (for previous biennium)**
**** = Usageactivityhrs-conditionmodel07-09update.txt (for current biennium)**

6.1.2 Links to External Files

The condition model is linked to an external MS Access database which contains the *Fleet Approval Information* for all biennia. This link is referred to in MS Access 2003 jargon as a "live link." A live link is a direct connection to external data that resides in an external reservoir (i.e., database or external folder) other than the database within the application. The link replicates the data as a table in the database. Any changes made to the original data are dynamically updated and reflected in the database. No changes can be made to the original data from within the database, so live links provide a secure way of availing data for use externally, while preserving the integrity of the original data. Deleting a live link automatically renders that data inaccessible and therefore unavailable for use by the application.

The condition model uses fleet approval information from the three most recent biennia (including the current one). The external MS Access database containing the *Fleet Approval Information* resides in the *Reports* folder where all input files are located and **must** be replaced with the most recent version **every** time the model is run. Also, if changes are made to the external MS Access database, the live link has to be redefined accordingly, so that the condition model uses the most recent data to determine *Active Turn-Ins* used for zero-ranking, as explained later in Section 6.5.3.

For example, in the current biennium (2007-2009), the condition model requires the following tables to be live linked into the model database:

- tbl-Fleet Approval Information 03-05
- tbl-Fleet Approval Information 05-07
- tbl-Fleet Approval Information 07-09

Note: The database name "Final Fleet Plan 99-01 Ed's Portion Only.mdb" **can** be changed **without** making any changes in the condition model. However, the fleet approval information **must** retain the same name format, i.e. "tbl-Fleet Approval Information *Biennium*".

6.1.3 Static Input Data Files

The condition model utilizes data from static input data files that have already been stored internally in the model in the form of database tables. These additional files are briefly described next.

- **tblTurnIn-TempReassignment**. The original data file was an MS Excel spreadsheet named "*temporary vehicle assignments.xls."* This file was imported into the condition model as a table named "*tblTurnIn-TempReassignment*." Initially, a live link was supposed to point to this file. However, since the file is constantly being updated and access to it from the source may be restricted, a better alternative was to have it imported into the database as described above. The most current version of this file **must** be imported **every** time the model is run (for current reports/current biennium).

- **tblStandards_Costs**. This table contains a list of standard parameters pertaining to age, usage, costs and related data and plays a critical role in the determination of the c*ondition index* (CI) of an asset. The CI (a quantitative measure) is then used to categorize equipment qualitatively from *very good* to *very poor* and also to prioritize (i.e., rank) equipment marked for replacement. This file/table resides in the database of the condition replacement model.

- **tblStandards_Costs**. This table contains a list of all equipment status codes and their descriptions.

- **CONDITION_STANDARDS**. This table includes a list of qualitative descriptors of the CI of equipment, based on predetermined thresholds. These descriptors and their corresponding threshold ranges are summarized in Table 6.2 below. The threshold ranges are in percentages.

Table 6.2: Condition Index descriptors and threshold ranges

Condition	Condition Index Lower Threshold	Condition Index Upper Threshold
Very Good	0	16
Good	16	36
Fair	36	66
Poor	66	86
Very Poor	86	Above 86*

* Since it is possible to have a CI > 100%, the default figure used in the condition model as the upper threshold is 9999999 (a high number).

- **BIENNIUM**. This table includes a list of biennia that is used later to populate a dropdown list on the graphical user interface. The user can select the biennium of interest from this dropdown list before running the model. An *administrator* can add a future biennium by adding a new entry in the "Biennium" field in this table.

- **REGION_DISTRICTS_LIST.** This table contains a list of regions and their corresponding districts, as well as crew assigned to those regions.

- **LOG_IN_RECORD**. This table is used to record run time, date and files uploaded whenever the model is run. The data in this table are used to run the model and is therefore for (internal) monitoring purposes only.

6.2 DATABASE DESCRIPTION

The following sections describe the database structure by first presenting a graphical view of the underlying model (i.e., Entity Relationship Diagram), followed by a detailed description of the entities and data structures (i.e., metadata).

6.2.1 Entity Relationship Diagram

Figure 6.2 depicts the Entity Relationship (ER) diagram for the ODOT Fleet Services condition model. The ER diagram includes a total of 12 tables. The structure of the tables and the relationships amongst them are defined dynamically as the model is run using the data provided by the input files described in Section 6.1.1.

6.2.2 Database Model Structure

The following is a description of the data model structure which provides definitions of all the fields within a table as well as its relationship(s) to other tables. Relationships are depicted as primary keys (PK) or foreign keys (FK). A primary key is a candidate field that uniquely identifies each row in an entity. These ensure that there are no duplicate data in the database. A foreign key identifies a column or a set of columns in one (referencing) table that refers to a column or set of columns in another (referenced) table.

6.2.2.1 EQUIP_MASTER Table

The table **EQUIP_MASTER** contains a list of all the equipment and their description. It is the normalized form of the data in the input file named ***EQMSTR.TXT***. Figure 5.3 depicts the structure of the table, including the fields that represent the primary and foreign keys, field names, field types and filed sizes, which are in parenthesis.

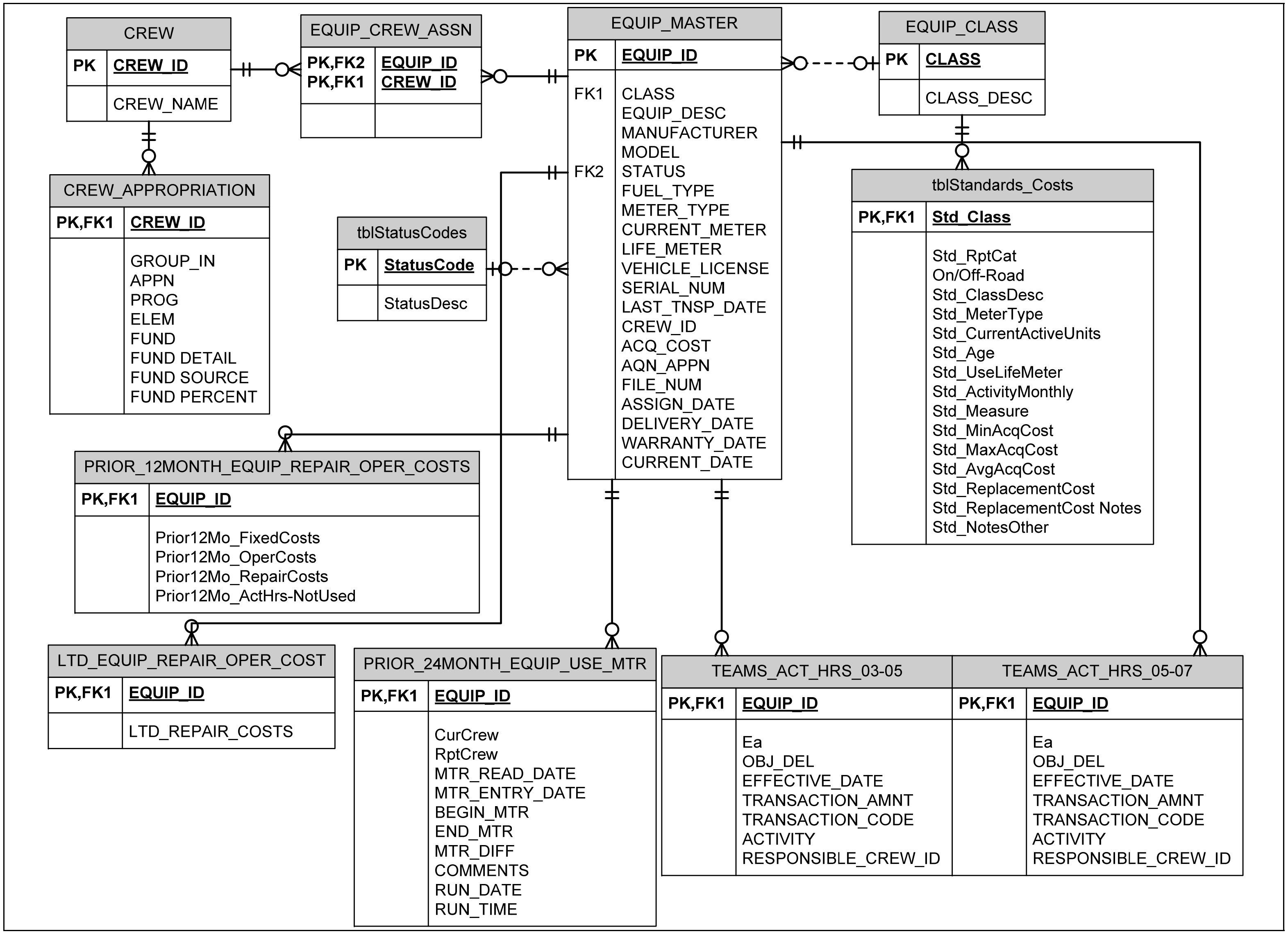
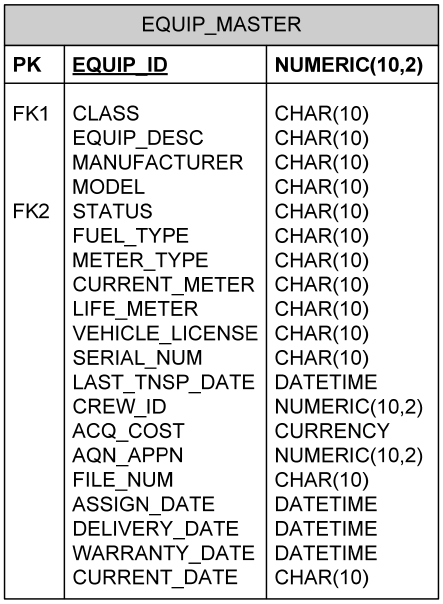


Figure 6.2: Entity Relationship diagram for the condition model



EQUIP_MASTER		
PK	**<u>EQUIP_ID</u>**	**NUMERIC(10,2)**
FK1	CLASS	CHAR(10)
	EQUIP_DESC	CHAR(10)
	MANUFACTURER	CHAR(10)
	MODEL	CHAR(10)
FK2	STATUS	CHAR(10)
	FUEL_TYPE	CHAR(10)
	METER_TYPE	CHAR(10)
	CURRENT_METER	CHAR(10)
	LIFE_METER	CHAR(10)
	VEHICLE_LICENSE	CHAR(10)
	SERIAL_NUM	CHAR(10)
	LAST_TNSP_DATE	DATETIME
	CREW_ID	NUMERIC(10,2)
	ACQ_COST	CURRENCY
	AQN_APPN	NUMERIC(10,2)
	FILE_NUM	CHAR(10)
	ASSIGN_DATE	DATETIME
	DELIVERY_DATE	DATETIME
	WARRANTY_DATE	DATETIME
	CURRENT_DATE	CHAR(10)

Figure 6.3: EQUIP_MASTER Table

6.3 CONDITION REPLACEMENT MODEL

This section describes the main operations and functionality of the condition model.

6.3.1 Condition model logic

Figure 5.4 depicts the main steps (i.e., logic) involved in running the condition model. The graphical user interface (GUI) that implements this logic is presented in Appendix B.

As Figure 5.4 shows, the process begins with the user selecting the current biennium. A check is performed to ensure that the user has selected the current biennium before proceeding. If this is the case, the condition model attempts to import the periodic input files described in Section 6.1.1. If the periodic input files cannot be imported successfully, there are two causes why this could happen (see left box in Figure 5.4): either some input files are missing from the directory; or the folder where the input files are stored is not located in the same folder as the application.

If all input files are imported successfully, the condition model is run. If the condition model runs successfully, a relational database with the structure shown in Figure 5.2 is created. Also, a series of reports is available to the user (see Section 5.6). If an error occurs, the user notes the error and reports it to ODOT's IT division.

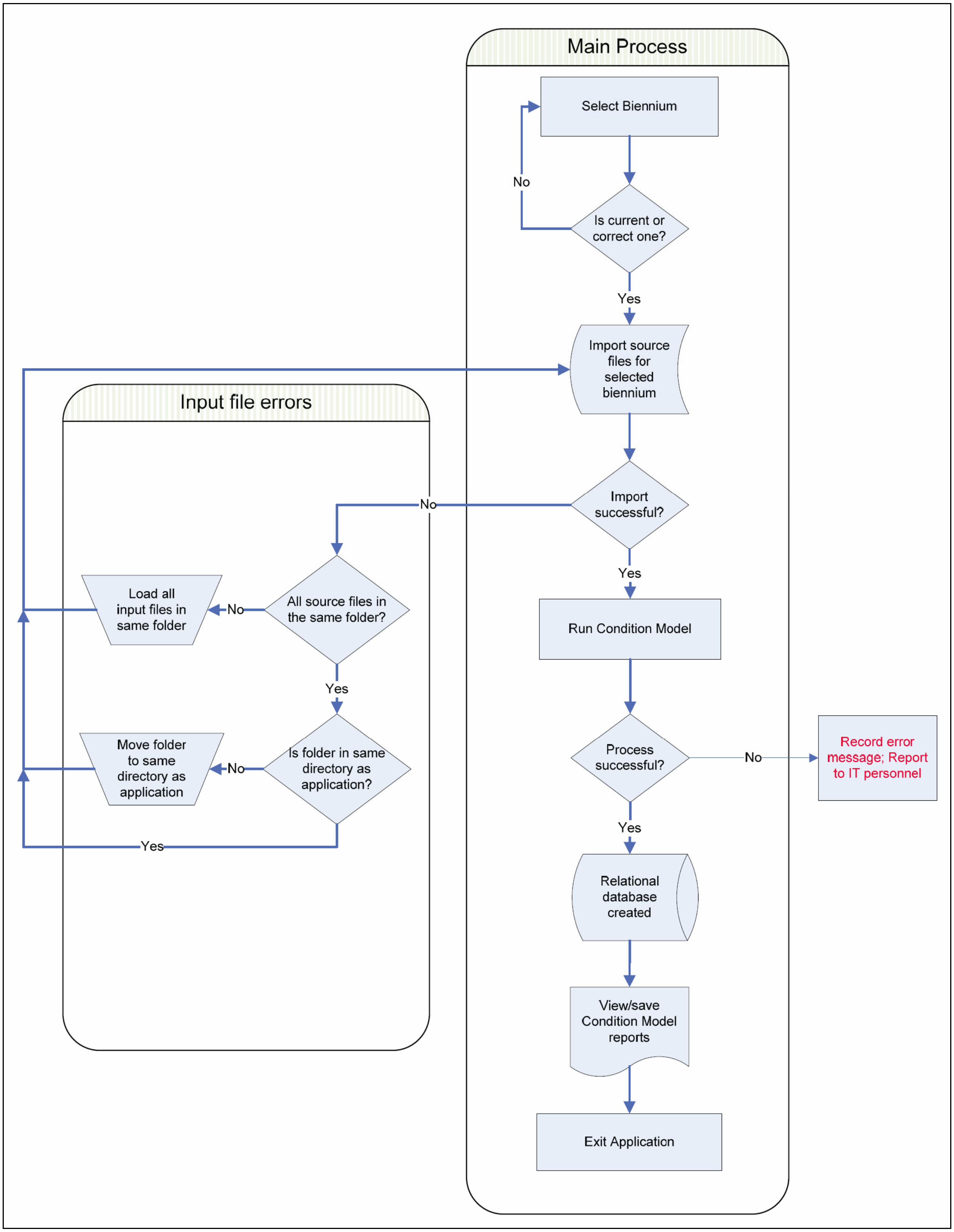

Main Process
Select Biennium
No
Is current or correct one?
Yes
Import source files for selected biennium
Import successful?
No
Yes
Run Condition Model
Process successful?
No
Record error message; Report to IT personnel
Yes
Relational database created
View/save Condition Model reports
Exit Application
Input file errors
All source files in the same folder?
No
Load all input files in same folder
Yes
Is folder in same directory as application?
No
Move folder to same directory as application
Yes

Figure 6.4: Conceptual flow diagram of condition model logic

6.3.2 Equipment replacement criteria

This section presents the two methodologies implemented in the condition model to determine the *condition index* (CI) and *rank* for the different equipment assets. The CI is a ratio of the "fitness" of each equipment asset based on predetermined standards. Three criteria, namely age, usage, and cost, are considered. Once the CI is computed, equipment assets are ranked in descending order of their CI. Equipment assets that are not to be considered for replacement in the current biennium are given a rank of zero, as described in Section 6.3.3.

Various analyses were carried out to establish the "best" criteria to use in determining the ranking of equipment to be replaced. The three-factor condition model uses age, usage, and cost to compute the CI, as depicted by Equation 6-1.

$$\frac{\text{Eq Age/Age Std + Eq Cost/Acq Cost + Eq Life Use/Use Std*}}{\text{2 or 3 (depending on existence of Use Std)}} \qquad \textbf{(6-1)}$$

The two-factor condition model considers only age and usage, and the calculation of the index is shown by Equation 6-2. This is the **only** difference between both models; therefore all other aspects described in this section apply to both – no separate documentation is necessary.

$$\frac{\text{Eq Age/Age Std + Eq Life Use/Use Std*}}{\text{1 or 2 (depending on existence of Use Std)}} \qquad \textbf{(6-2)}$$

6.3.3 Zero ranking analysis

All equipment assets that should not be considered for replacement in the **current** biennium are kept in a separate application known as the fleet plan. Eventually, these assets are designated a zero rank in the final condition model report. The new condition model includes an external link to the fleet plan (see Section 6.1.2) and automatically selects those items that should not be considered for replacement in the current biennium and assigns them a rank of zero.

Figure 5.5 depicts the process that is performed to select which assets should be assigned a rank of zero this biennium. It is important to mention that this process was performed manually before. Now the new condition model automatically performs this process.

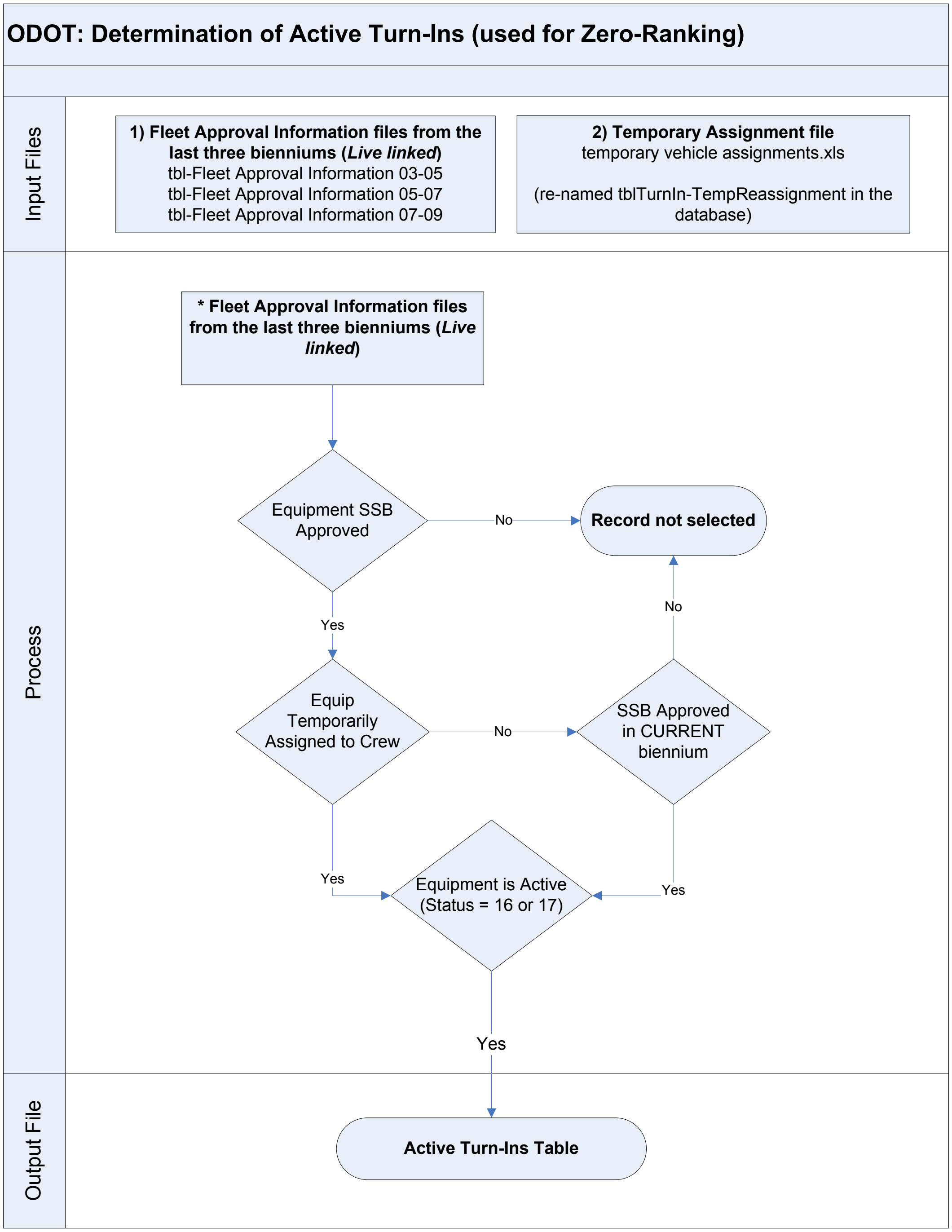
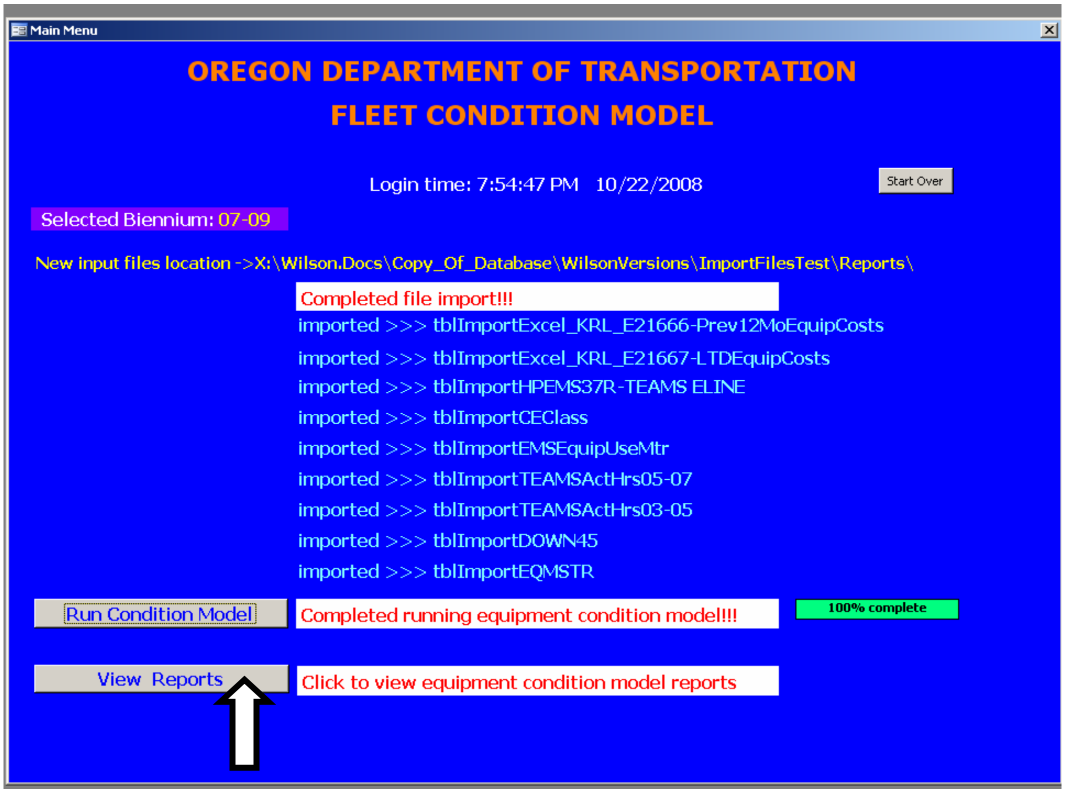

ODOT: Determination of Active Turn-Ins (used for Zero-Ranking)
Input Files
Process
Output File
1) Fleet Approval Information files from the last three bienniums (Live linked)
tbl-Fleet Approval Information 03-05
tbl-Fleet Approval Information 05-07
tbl-Fleet Approval Information 07-09
2) Temporary Assignment file
temporary vehicle assignments.xls
(re-named tblTurnIn-TempReassignment in the database)
* Fleet Approval Information files from the last three bienniums (Live linked)
Equipment SSB Approved
No
Record not selected
Yes
No
Equip Temporarily Assigned to Crew
No
SSB Approved in CURRENT biennium
Yes
Equipment is Active (Status = 16 or 17)
Yes
Yes
Active Turn-Ins Table

Figure 6.5: Determination of active turn-ins

6.4 REPORTS

Once the condition model is run successfully, customized reports are automatically generated. These automatic reports can be viewed and saved by the user from the reports user interface (see Figure 5.6). The reports user interface is accessed by clicking the “View Reports” button indicated by the arrow in Figure 5.6.



Main Menu

OREGON DEPARTMENT OF TRANSPORTATION

FLEET CONDITION MODEL

Login time: 7:54:47 PM 10/22/2008

Start Over

Selected Biennium: 07-09

New input files location ->X:\Wilson.Docs\Copy_Of_Database\WilsonVersions\ImportFilesTest\Reports\

Completed file import!!!

imported >>> tblImportExcel_KRL_E21666-Prev12MoEquipCosts

imported >>> tblImportExcel_KRL_E21667-LTDEquipCosts

imported >>> tblImportHPEMS37R-TEAMS ELINE

imported >>> tblImportCEClass

imported >>> tblImportEMSEquipUseMtr

imported >>> tblImportTEAMSActHrs05-07

imported >>> tblImportTEAMSActHrs03-05

imported >>> tblImportDOWN45

imported >>> tblImportEQMSTR

Run Condition Model

Completed running equipment condition model!!!

100% complete

View Reports

Click to view equipment condition model reports

Figure 6.6: Reports user interface of the condition model

The old condition model was only capable of producing a single report. The new condition model includes a total of 7 report categories (for a total of 21 reports), thus avoiding the need for ODOT Fleet Services personnel to export data to other applications (e.g., Microsoft Excel) to perform additional data analyses. Following is a list of the different report categories and their corresponding individual reports line up:

1. **Final equipment replacement report**. This is the final report containing only ranked active equipment, i.e. equipment with status 16 (Active Non-Maintenance) or 17 (Active Maintenance).

2. **Repair history report**. Contains cumulative repair costs for the prior 12 months for all active equipment.

3. **Cost history report (fixed and operating costs)**. Contains cumulative fixed and operating costs for the prior 12-months for all active equipment.

4. **Categorized replacement report by equipment type**:

   a) By light fleet (diesel, gas)

   b) By heavy equipment

   c) By attachments

5. **Categorized replacement report for**:

   a) Crews

   b) Districts

   c) Regions

   d) State wide

6. **Replacement report, grouped by very poor, poor and fair conditions, and by appropriation**:

   a) Appropriation 010

   b) Appropriation 050

   c) Appropriation 080

   d) Appropriation 110

   e) Appropriation 120

   f) Appropriation 130

   g) Appropriation 170

   h) Appropriation 200

   i) Appropriation 160

7. **Categorized metered (hrs/miles) and non-metered**:

   a) Metered equipment - those with standard measures in miles or hours.

   b) Non-metered equipment – measured by activity

7.0 SUMMARY AND CONCLUSION

This project focused on two interrelated areas in equipment replacement modeling for fleets. The first area was research-oriented and addressed fundamental assumptions in engineering economic replacement modeling. The second area addressed the need of the Oregon Department of Transportation (ODOT) Fleet Services Section for a modern, user-friendly, well constructed and documented fleet condition model. This project was completed while working closely with ODOT Fleet Services, and involved the use of many different approaches. Analytical analysis, simulation, and database and software application development were all conducted in this project. Regular reviews by ODOT Research and the Technical Advisory Committee were held to keep the project focused and on schedule.

Analysis of fleet replacement yielded the following significant findings:

- If simple ranking criteria are used to prioritize replacement within a single equipment class, simply replacing the oldest equipment first was overall the most cost effective replacement prioritization model.
- If equipment in a fleet is utilized such that newer equipment is generally preferred over older equipment, thus resulting in decreased equipment usage with age, then
  - a) fleet operating costs will be higher than if all equipment is equally utilized; and
  - b) the age standard for the fleet should be younger than if equipment in the fleet is equally utilized.

Other DOTs throughout the nation should find these results applicable to their particular replacement situation.

Accomplishments and significant findings related to the development of a modern, user-friendly, well constructed and documented fleet condition model are:

- Replacement practices at ODOT were in line with other DOTs throughout the nation, and a survey of the academic literature revealed little work that is directly applicable at ODOT.
- A new software tool was developed and installed that includes the following features:
  - It implements a formal Entity Relationship model;
  - It produces several reports automatically;
  - Recurrent procedures are dynamic;
  - It consolidates replacement rankings with fleet plan data; and
  - It implements an error-proof, user-friendly graphical user interface.

The model developed as part of this research project has been implemented by ODOT. The newly developed model [(Age/Age standard)+(Usage/Use standard) + (LTD Cost/ Acquisition Cost)] is being used by ODOT Fleet Services as a measure to prioritize replacements. The model performs similar to several other measures and seems adequate if age and use standards are updated.

7.1 FURTHER RESEARCH

In the future, additional research could be completed that could investigate additional variables in the prioritization ranking models in an attempt to increase their effectiveness. It was suggested that employee input after use of the particular asset would be good anecdotal data to collect. It could be easily entered via a web interface when the employee returns the asset. This additional data could be normalized and analyzed to see if it was significant and could contribute to the formulas. It would, additionally, enhance the fleet manager's decision process with data from an employee that recently used the asset. Data collection was estimated to be random and set to a frequency, such that every asset would have one entry every quarter.

Further research could be continued to determine how much more efficient the leading formula per class of asset is in comparison to the ODOT implemented model. Although acknowledgements were made that implementation of several formulas, each per class of asset, would be more troublesome to implement, further research could be done in this area. This continuation would also require more data from each asset class.

Additional research could be done with regard to the interaction of a particular class of asset, when a new asset is introduced into the group. In particular, this interaction would be best research for vehicle assets.

8.0 REFERENCES

Ayres, R.M. and J. Waizeneker. A Practical Approach to Vehicle Replacement. *Chartered Mechanical Engineer*, Vol. 25, No. 9, 1978, pp. 73-75.

Bethuyne, G. Optimal Replacement Under Variable Intensity of Utilization and Technological Progress. *The Engineering Economist*, 1998, pp. 85-105.

Buddhakulsomsiri, J. and P. Parthanadee. Parallel Replacement Problem for a Fleet with Dependent Use. Presented at the Industrial Engineering Research Conference (IERC), Orlando, FL, 2006.

Chee, P.C.F. A Practical Vehicle Replacement Policy for Ontario Hydro. *Ontario Hydro Research Quarterly,* Vol. 27, No. 3, Third Quarter, 1975, pp. 3-6.

Davenport, N.S., M.D. Anderson, and P.A. Farrington. Development and Application of a Vehicle Procurement Model for Rural Fleet Asset Management. *Transportation Research Record*, No. 1927, 2005, pp. 123-127.

Dhillon, B.S. *Life Cycle Costing*. Gordon and Breach Science Publishers, New York. 1989.

Drinkwater, R.W. and N.A.J. Hastings. An Economic Replacement Model. *Operational Research Society*, OR, Vol. 18, No. 2, 1967, pp. 121-138.

Eilon, S., J. R. King, and D. E Hutchinson. A Study in Equipment Replacement; *Operational Research Society*, OR, Vol. 17, No. 1, 1966 pp. 59-71.

Grant, E.L., W.G. Ireson, and R.S. Leavenworth. *Principles of Engineering Economy.* John Wiley and Sons, New York, 1990.

Hartman, J. C. A General Procedure for Incorporating Asset Utilization Decisions into Replacement Analysis. *Engineering Economist*, Vol. 44, No. 3, 1999, pp. 217.

Hartman, J. C. An Economic Replacement Model with Probabilistic Asset Utilization. *IIE Transactions*, 2001, pp. 717-727.

Hartman, J. C. Multiple Asset Replacement Analysis Under Variable Utilization and Stochastic Demand. *European Journal of Operational Research*, Vol. 159, No. 1, 2004, pp. 145-165.

Hastings, N.A.J. The Repair Limit Replacement Method. *Operational Research Society,* OR, Vol. 20, No. 3, 1969, pp. 337-349.

Karabakal, N., J.R. Lohmann, and J.C. Bean. Parallel Replacement Under Capital Rationing Constraints. *Management Science*, 1994, pp. 305-319.

Khasnabis, S., J. Bartus, and R.D. Ellis. Asset Management Framework for State Departments of Transportation to Meet Transit Fleet Requirements. *Transportation Research Record*, No. 1835, 2003, pp. 74-83.

Kobbacy, K., & Nicol, S. (1994). Sensitivity analysis of rent replacement models. *International Journal of Production Economics* , 267-279.

Kriett, P. (2008, October 17). Cost Data for ODOT Vehicles. (T. Wagner, Interviewer)

Love, C.E., R. Rodger, and G. Blazenko. Repair Limit Policies for Vehicle Replacement. *INFOR Journal,* Vol. 20, 1982,pp. 226-37.

Nakagawa, T. and S. Osaki. The Optimum Repair Limit Replacement Policies. *Operational Research Quarterly*, Vol. 25, No. 2, 1974, pp. 311-317.

Randhawa, S. and K. Douglas, (1998) "Fleet Replacement: Methodology and Evaluation", Final report for ODOT/OSU Interagency Agreement 15520.

Redmer, A. Vehicle Replacement Planning in Freight Transportation Companies. Proceedings of the 16th Mini - EURO Conference and 10th Meeting of EURO Working Group Transportation, Poznan, Poland., 2005.

Rees, L.P., E.R. Clayton, and B.W. III Taylor. Network Simulation Model for Police Patrol Vehicle Maintenance and Replacement Analysis. *Comps. Environ. & Urban Sys*, Vol. 7, No. 3, 1982, pp. 191-196l.

Simms, B.W., B.G. Lamarre, and A.K.K. Jardine, Optimal Buy, Operate and Sell Policies for Fleets of Vehicles. *European Journal of Operational Research*, Vol. 15, No. 2, 1984, pp. 183-195.

Venkatakrishnan, K.S. and S. Venmathi. Optimal Replacement Time of Equipment via Simulation for Truncated Failure Distributions; *Microelectronics and Reliability*, Vol. 29, No. 1, 1989, pp. 49-52.

Weissmann, J., A.J. Weissmann, and S. Gona, Computerized Equipment Replacement Methodology. *Transportation Research Record*, No. 1824, 2003, pp. 77-83.

APPENDIX A: CONDITION MODEL USER GUIDE

A-1 RUNNING THE CONDITION MODEL

This appendix describes the necessary steps to run the condition model MS Access application. These instructions should be used in conjunction with the logic described in Section 5.5.1.

A-1.1 STARTING THE CONDITION MODEL APPLICATION

The condition model application is started by clicking on the application's icon on the computer's desktop (if available), or by double clicking on the MS Access file that contains the condition model. This will open the "Welcome Screen," depicted in **Figure A-1**.

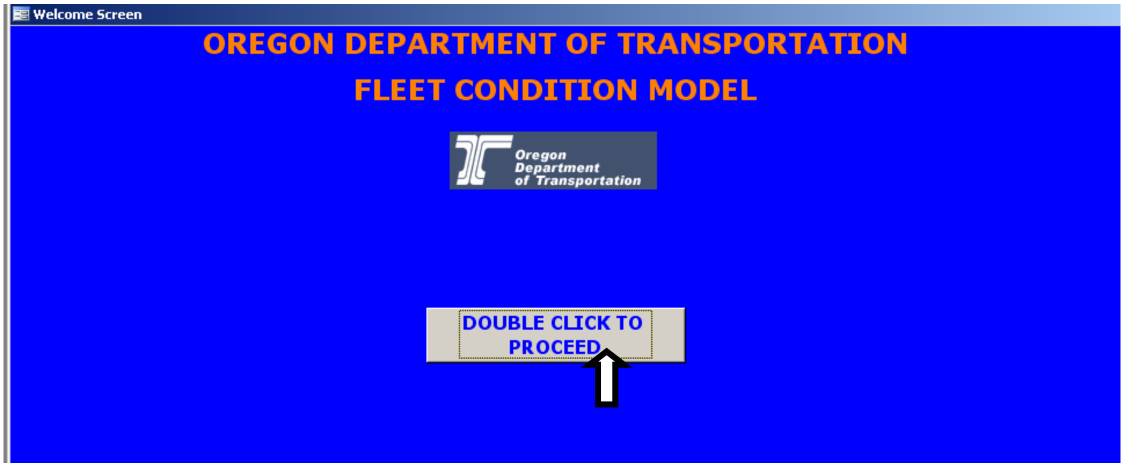

Welcome Screen
OREGON DEPARTMENT OF TRANSPORTATION
FLEET CONDITION MODEL
Oregon Department of Transportation
DOUBLE CLICK TO PROCEED

Figure A-1: Condition Model Welcome Screen

The user must double click on the button indicated with an arrow in **Figure A-1** in order to proceed to the "Main Menu" screen.

A-1.2 INTERACTING WITH THE MAIN MENU SCREEN

Selecting the Desired Biennium
The first step after the user has accessed the "Main Menu" screen (depicted in **Figure A-2**), is to select the biennium of interest. This is done by clicking the arrow to the right of the combo box shown in **Figure A-2** and then selecting the desired biennium from the dropdown list (see **Figure A-3**).

Selecting the **correct** biennium is a very critical step as it sets the time period that the condition model will use to perform computations and hence directly impacts the output obtained from running it.

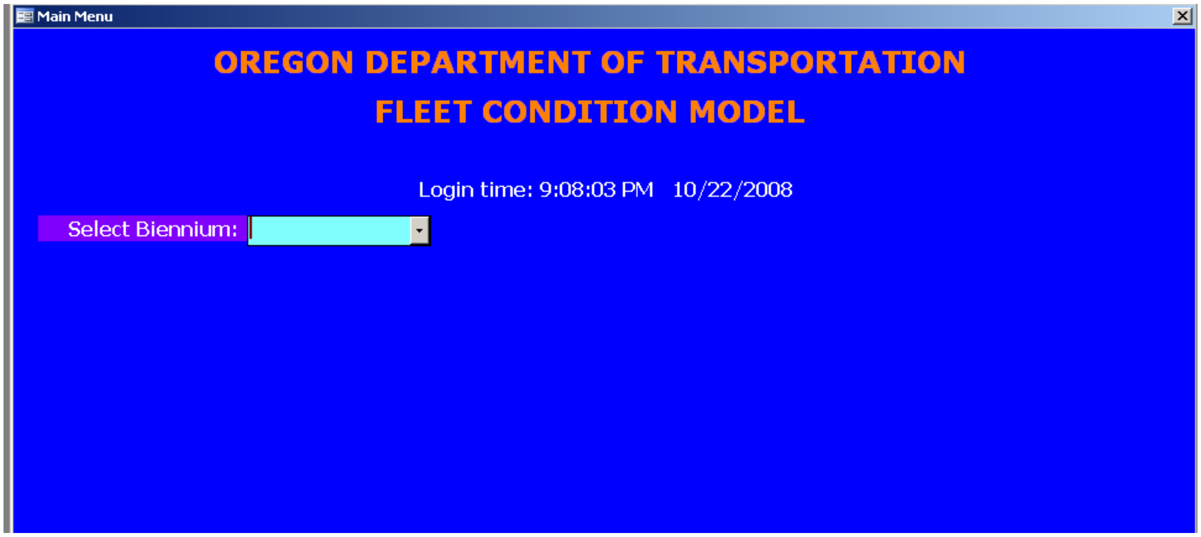

Main Menu
OREGON DEPARTMENT OF TRANSPORTATION
FLEET CONDITION MODEL
Login time: 9:08:03 PM 10/22/2008
Select Biennium:

Figure A-2: Main Menu Screen – Select Biennium Combo Box

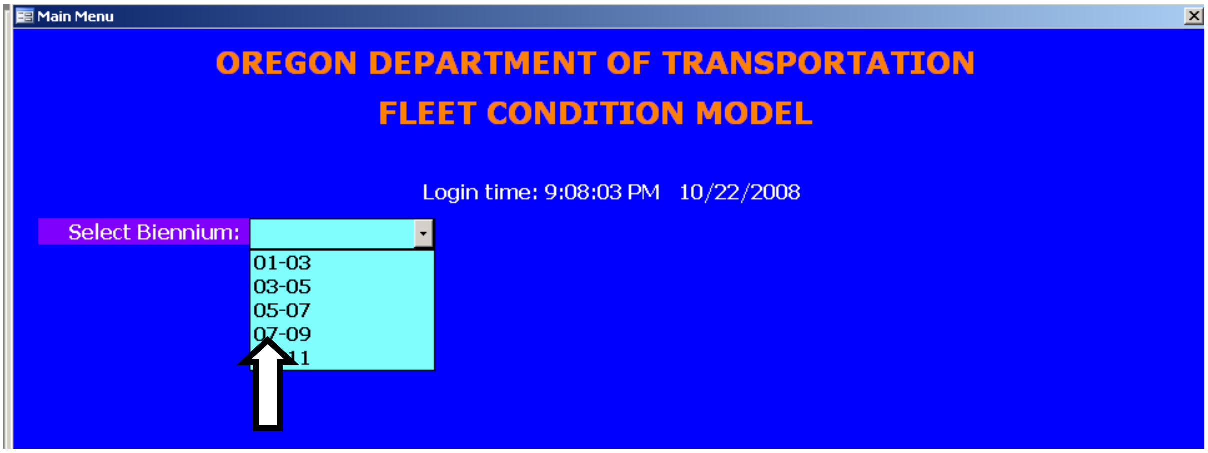
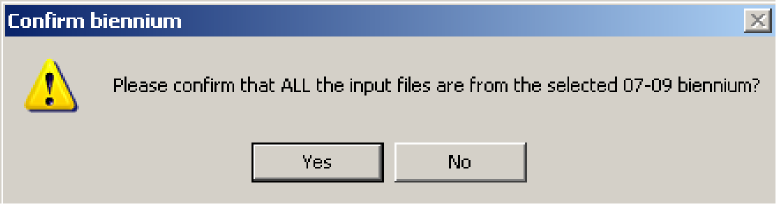


Figure A-3. Main Menu Screen – Selecting a Biennium

Once the user selects a biennium from the list, a message box appears requesting confirmation that this is the desired choice (see **Figure A-4**). This step ensures that the user does not inadvertently select the wrong biennium. It also offers an opportunity to select a different biennium, if need be.

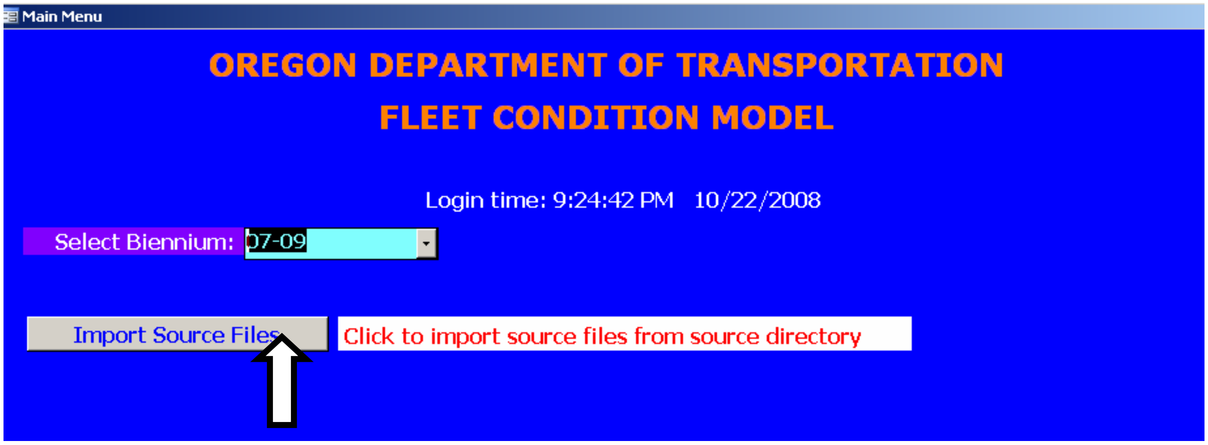


Please confirm that ALL the input files are from the selected 07-09 biennium?

Figure A-4. Main Menu Screen – Correct Biennium Message Box

A-1.2.1 Importing Input Files

Once the biennium is selected and confirmed, the application locks in the selected option and proceeds to the input file import function. A new button labeled "Import Source Files" is now visible (see **Figure A-5**). The input file process is started by clicking on this button.



OREGON DEPARTMENT OF TRANSPORTATION

FLEET CONDITION MODEL

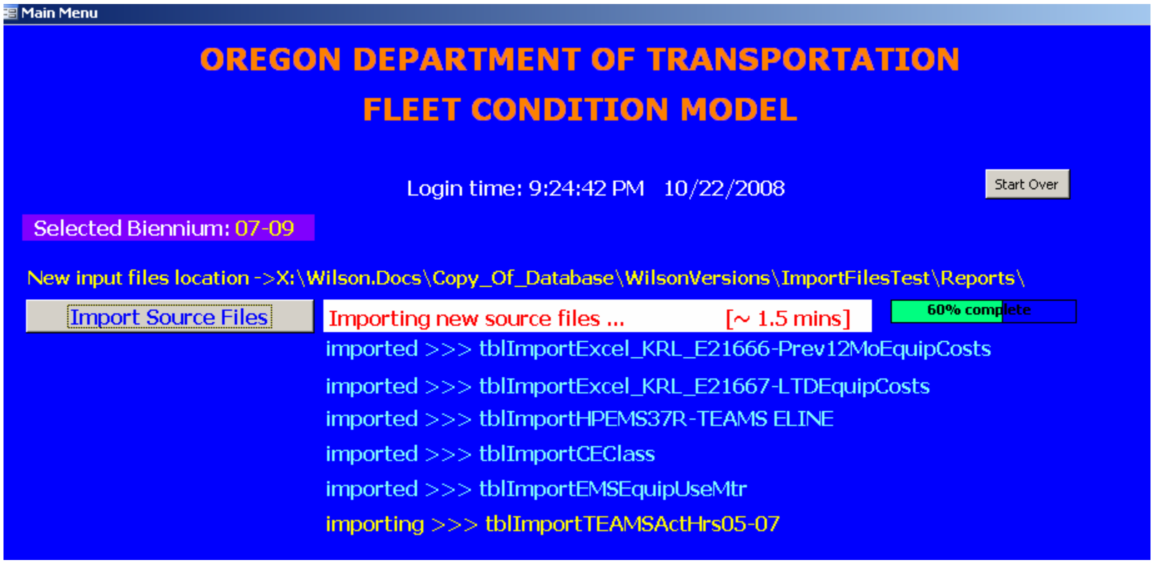
Login time: 9:24:42 PM 10/22/2008

Select Biennium: 07-09

Click to import source files from source directory

Figure A-5. Main Menu Screen – Importing Input Files

As the different input files are imported by the condition model, a bright green progress bar indicating the percentage of completion will be visible to the right of the label (see **Figure A-6**). Before importing new input files, the application deletes the existing files from the condition model database. This process takes a few seconds and may not be noticeable. The files being imported are listed sequentially below the label. A file in the process of being imported is highlighted in yellow font, and once it is successfully imported, the font turns to light blue.



OREGON DEPARTMENT OF TRANSPORTATION
FLEET CONDITION MODEL

Login time: 9:24:42 PM 10/22/2008

Selected Biennium: 07-09

New input files location ->X:\Wilson.Docs\Copy_Of_Database\WilsonVersions\ImportFilesTest\Reports\

Importing new source files ... [~ 1.5 mins]

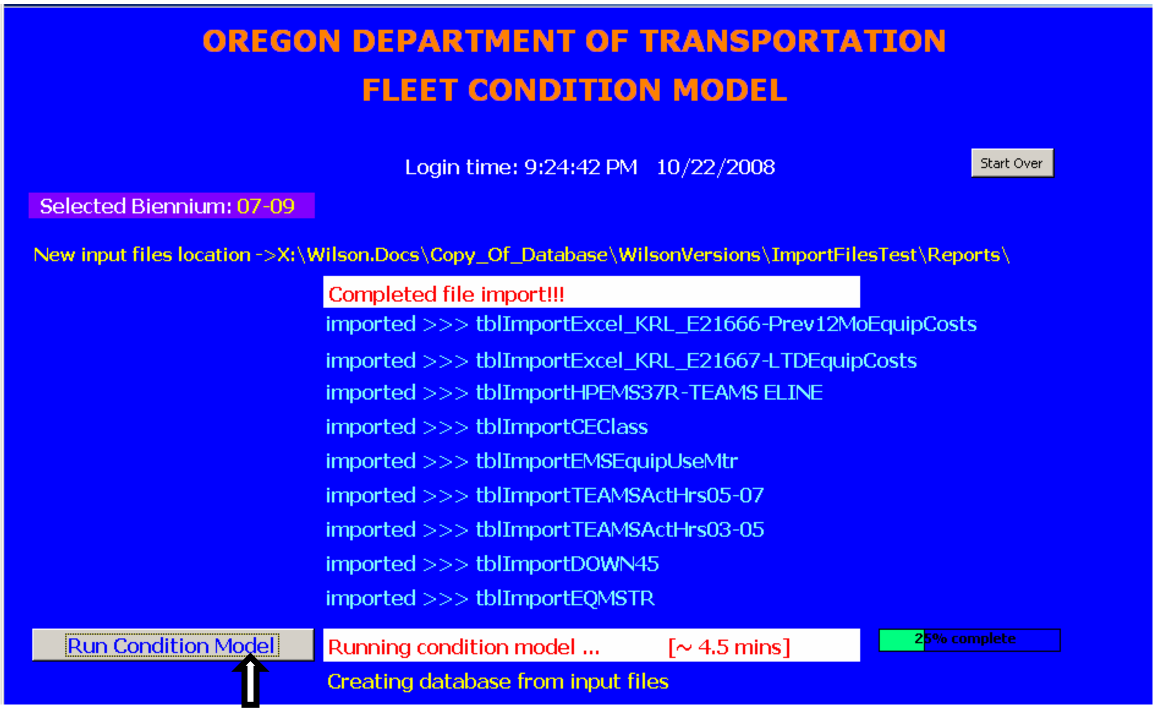
```
imported >>> tblImportExcel_KRL_E21666-Prev12MoEquipCosts
imported >>> tblImportExcel_KRL_E21667-LTDEquipCosts
imported >>> tblImportHPEMS37R-TEAMS ELINE
imported >>> tblImportCEClass
imported >>> tblImportEMSEquipUseMtr
importing >>> tblImportTEAMSActHrs05-07
```

Figure A-6. Main Screen Menu – Progress Bar and Input File Status

Once the import process is complete, the computer produces a beep sound and the button for the next process (i.e., Running Condition Model) becomes visible.

A-1.2.2 Executing the Condition Model

The condition model is executed by clicking on the button labeled "Run Condition Model" (see **Figure A-7**). The progress bar should align to the corresponding label and a new label describing the procedures taking place should appear below the button description label.



OREGON DEPARTMENT OF TRANSPORTATION

FLEET CONDITION MODEL

Login time: 9:24:42 PM 10/22/2008

Selected Biennium: 07-09

New input files location ->X:\Wilson.Docs\Copy_Of_Database\WilsonVersions\ImportFilesTest\Reports\

Completed file import!!!

```
imported >>> tblImportExcel_KRL_E21666-Prev12MoEquipCosts
imported >>> tblImportExcel_KRL_E21667-LTDEquipCosts
imported >>> tblImportHPEMS37R-TEAMS ELINE
imported >>> tblImportCEClass
imported >>> tblImportEMSEquipUseMtr
imported >>> tblImportTEAMSActHrs05-07
imported >>> tblImportTEAMSActHrs03-05
imported >>> tblImportDOWN45
imported >>> tblImportEQMSTR
```

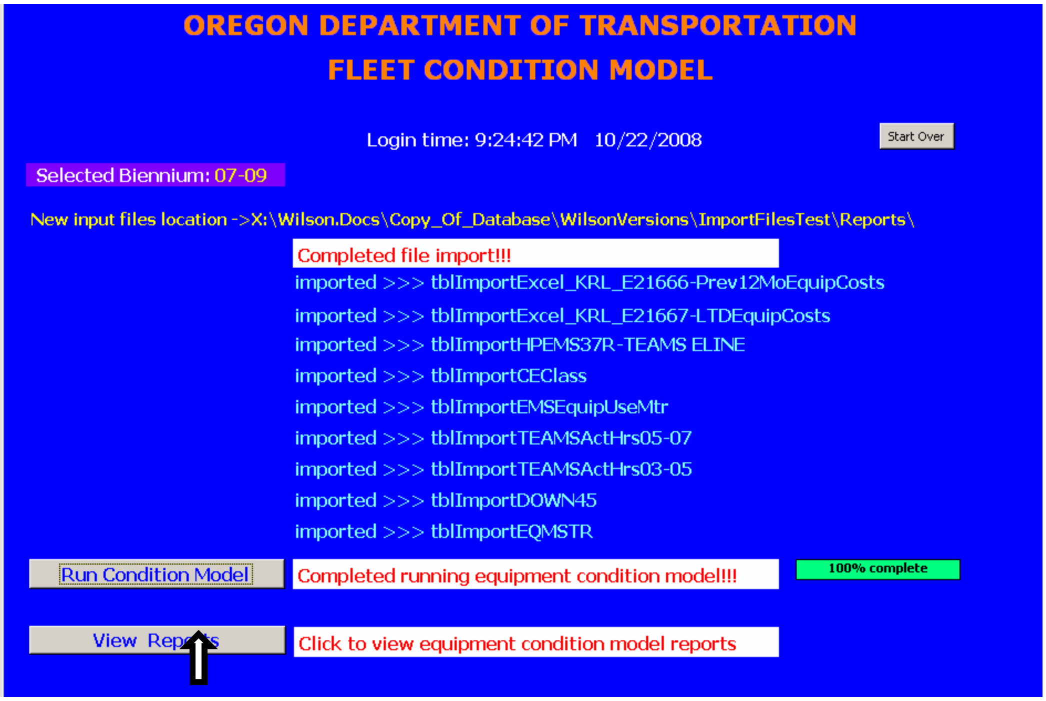
Running condition model ... [~ 4.5 mins]

Creating database from input files

Figure A-7: Main Menu Screen – Executing the Condition Model

A-1.2.3 Viewing Reports

Once the model executes successfully, the "View reports" button becomes visible (see **Figure A-8**). Clicking on this button opens the reports interface form (see **Figure A-9**). All reports produced by the condition model are listed on this page. The user has the option of viewing the reports in an MS Access spreadsheet format and/or save them to a local drive as MS Excel spreadsheets. The default location where files are saved is the same directory where the application resides. This is indicated in the textbox next to the "File Location" label depicted in **Figure A-9**. The user also has the option to save the reports in a different location by typing the appropriate directory path. The "Reset" button resets the file location path to the default value.



OREGON DEPARTMENT OF TRANSPORTATION

FLEET CONDITION MODEL

Login time: 9:24:42 PM 10/22/2008

Selected Biennium: 07-09

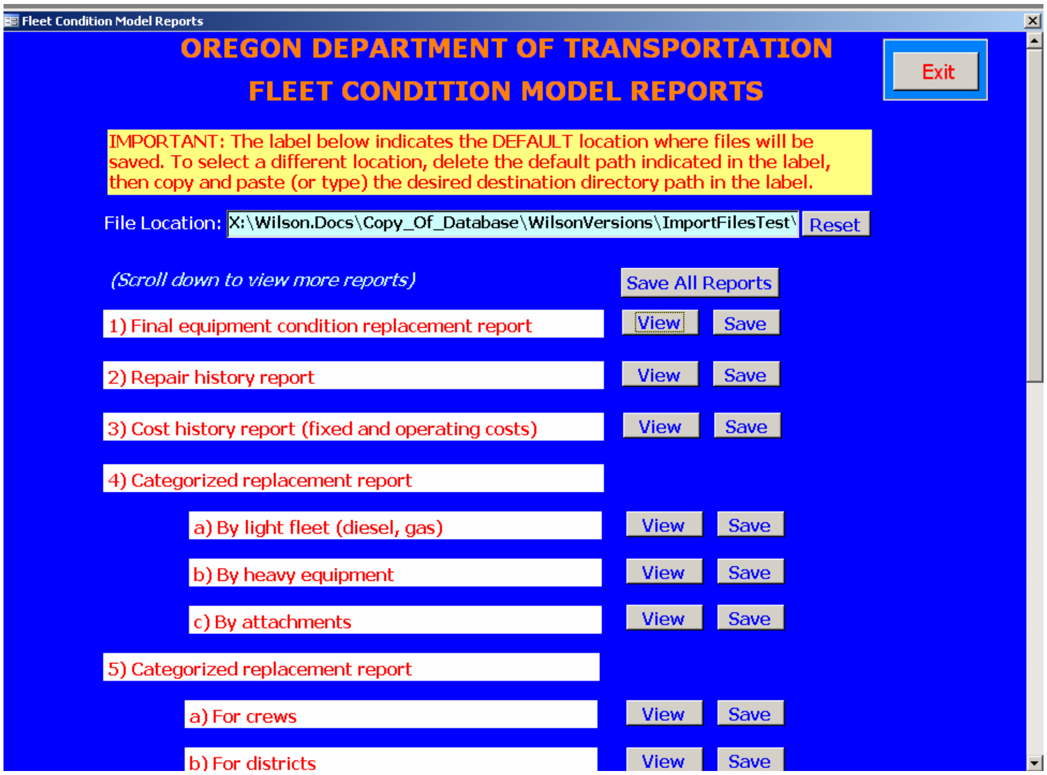
New input files location ->X:\Wilson.Docs\Copy_Of_Database\WilsonVersions\ImportFilesTest\Reports\

Completed file import!!!
imported >>> tblImportExcel_KRL_E21666-Prev12MoEquipCosts
imported >>> tblImportExcel_KRL_E21667-LTDEquipCosts
imported >>> tblImportHPEMS37R-TEAMS ELINE
imported >>> tblImportCEClass
imported >>> tblImportEMSEquipUseMtr
imported >>> tblImportTEAMSActHrs05-07
imported >>> tblImportTEAMSActHrs03-05
imported >>> tblImportDOWN45
imported >>> tblImportEQMSTR

Completed running equipment condition model!!!

Click to view equipment condition model reports

Figure A-8: Main Menu Screen – View Reports Button



OREGON DEPARTMENT OF TRANSPORTATION

FLEET CONDITION MODEL REPORTS

IMPORTANT: The label below indicates the DEFAULT location where files will be saved. To select a different location, delete the default path indicated in the label, then copy and paste (or type) the desired destination directory path in the label.

File Location: X:\Wilson.Docs\Copy_Of_Database\WilsonVersions\ImportFilesTest\

(Scroll down to view more reports)

1) Final equipment condition replacement report
2) Repair history report
3) Cost history report (fixed and operating costs)
4) Categorized replacement report
a) By light fleet (diesel, gas)
b) By heavy equipment
c) By attachments
5) Categorized replacement report
a) For crews
b) For districts

Figure A-9: Main Menu Screen – Reports Form Interface

Report files can be saved individually by clicking the “Save” button next to the corresponding report (see **Figure A-9**). However, all reports can be saved simultaneously in the same directory by clicking the “Save All Reports” button located below the “Reset” button.

The “Exit” button located on the top right corner of the “Reports Form” closes the application and exits MS Access, essentially ending the session and ending the entire program. A message box warns the user and requests confirmation of action when this button is clicked.

APPENDIX B: DEVELOPER’S DOCUMENTATION

This appendix describes the various MS Access queries used to develop the relational database and to produce the end-user reports. The types of queries written include *select*, *action* (i.e., insert, update and delete), *make table* and *data definition* (i.e., alter table) queries. The queries are written in Structured Query Language (SQL).

Queries are listed by query name in the following sections for ease of reference (i.e., referenced by subsection number which represents the query number in this document). Also, they are listed in the same order in which they are executed when the model is run.

For each query, a brief description of its main purpose and the SQL code are included.

QUERIES

qry_Update_Mtr_null_in_tblImportEMSEquipUseMtr

Function	Updates null values for END_METER to zero in tblImportEMSEquipUseMtr, essentially converting them from string to numeric
SQL code	UPDATE tblImportEMSEquipUseMtr SET tblImportEMSEquipUseMtr.EndMtr = 0 WHERE (((tblImportEMSEquipUseMtr.EndMtr) Is Null));

qry_create_EQUIP_CLASS

Function	Runs a make table query to create a new table EQUIP_CLASS from input file CECLASS.TXT – list of equipment class descriptions
SQL code	SELECT tblImportCEClass.CL_CLASS AS CLASS, tblImportCEClass.CL_CLASS_DESC AS CLASS_DESC INTO EQUIP_CLASS FROM tblImportCEClass;

qry_create_EQUIP_MASTER

Function	Runs a make table query to create a new table EQUIP_MASTER from input file EQMSTR.xls – Equipment master file

SQL code	SELECT tblImportEQMSTR.[VEH NUMBER] AS EQUIP_ID, tblImportEQMSTR.[VM CLASS] AS CLASS, tblImportEQMSTR.[VEH DESC] AS EQUIP_DESC, tblImportEQMSTR.[VEH MANUF] AS MANUFACTURER, tblImportEQMSTR.[VEH MODEL] AS MODEL, tblImportEQMSTR.STATUS, tblImportEQMSTR.[FUEL TYPE] AS FUEL_TYPE, tblImportEQMSTR.[MI/HR IND] AS METER_TYPE, tblImportEQMSTR.[CURRENT METER] AS CURRENT_METER, tblImportEQMSTR.[LIFE METER] AS LIFE_METER, tblImportEQMSTR.[VEH LICENSE] AS VEH_LICENSE, tblImportEQMSTR.[SERIAL NUMBER] AS SERIAL_NUM, tblImportEQMSTR.[LAST INSP DATE] AS LAST_INSP_DATE, tblImportEQMSTR.[VEH CREW] AS CREW_ID, tblImportEQMSTR.[ACQ COST] AS ACQ_COST, tblImportEQMSTR.[ACQ APPN] AS AQC_APPN, tblImportEQMSTR.[FILE NUMBER] AS FILE_NUM, tblImportEQMSTR.[ASSIGN DATE] AS ASSIGN_DATE, tblImportEQMSTR.[DELVRY DATE] AS DELIVER_DATE, tblImportEQMSTR.[WARRANTY DATE] AS WARRANTY_DATE, tblImportEQMSTR.[CURRENT DATE] AS [CURRENT_DATE] INTO EQUIP_MASTER
	FROM tblImportEQMSTR;

qry_create_EQUIP_CREW_ASSN

Function	Runs a make table query to create table EQUIP_CREW_ASSN which associates the EQUIP_MASTER and CREW tables

SQL code	SELECT DISTINCT tblImportEQMSTR.[VEH NUMBER] AS EQUIP_ID, tblImportEQMSTR.[VEH CREW] AS CREW_ID INTO EQUIP_CREW_ASSN
	FROM tblImportEQMSTR;

qry_create_CREW

Function	Runs a make table query to create table CREW which contains the crew identification number and crew description. This is filtered from the equipment master file, ensuring that ALL the crew existing at the time when the report is generated are accounted for in the model.

SQL code	SELECT DISTINCT tblImportEQMSTR.[VEH CREW] AS CREW_ID, tblImportEQMSTR.[CREW NAME] AS CREW_NAME INTO CREW
	FROM tblImportEQMSTR;

qry_create_PRIOR_24MONTH_EQUIP_USE_MTR

Function	Runs a make table query to create table PRIOR_24MONTH_EQUIP_USE_MTR from input file HPEMS62R.xls – Prior 24 month equipment usage report (EMSUse24)

SQL code	SELECT DISTINCT tblImportEMSEquipUseMtr.EquipID AS EQUIP_ID, tblImportEMSEquipUseMtr.CurCrew, tblImportEMSEquipUseMtr.RptCrew, tblImportEMSEquipUseMtr.MtrReadDt AS MTR_READ_DATE, tblImportEMSEquipUseMtr.EntryDate AS MTR_ENTRY_DATE, tblImportEMSEquipUseMtr.BeginMtr AS BEGIN_MTR, tblImportEMSEquipUseMtr.EndMtr AS END_MTR, [EndMtr]-[BeginMtr] AS MTR_DIFF, tblImportEMSEquipUseMtr.Comments AS COMMENTS, tblImportEMSEquipUseMtr.RunDate AS RUN_DATE, tblImportEMSEquipUseMtr.RunTime AS RUN_TIME INTO PRIOR_24MONTH_EQUIP_USE_MTR
	FROM tblImportEMSEquipUseMtr;

qry_create_TEAMS_ACT_HRS_03-05

Function	Runs a make table query to create table TEAMS_ACT_HRS_03-05 from input file Usageactivityhrs-conditionmodel03-05update.txt – TEAMS prior biennium activity hours

SQL code	SELECT DISTINCT [tblImportTEAMSActHrs03-05].Ea AS EQUIP_ID, [tblImportTEAMSActHrs03-05].Ea, [tblImportTEAMSActHrs03-05].[Obj Detl] AS OBJ_DETL, [tblImportTEAMSActHrs03-05].[Effective Date] AS EFFECTIVE_DATE, [tblImportTEAMSActHrs03-05].[Transaction Amount] AS TRANSACTION_AMNT, [tblImportTEAMSActHrs03-05].[Transaction Qty] AS TRANSACTION_QTY, [tblImportTEAMSActHrs03-05].[Trans Code] AS TRANSACTION_CODE, [tblImportTEAMSActHrs03-05].Activity AS ACTIVITY, [tblImportTEAMSActHrs03-05].[Responsible Unit] AS RESPONSIBLE_CREW_ID INTO [TEAMS_ACT_HRS_03-05]
	FROM [tblImportTEAMSActHrs03-05]
	WHERE [tblImportTEAMSActHrs03-05].Ea IS NOT NULL;

qry_parse_Ea_TEAMS_ACT_HRS_03-05

Function	Updates EQUIP_ID by parsing the two-prefix Ea (EQUIP_ID) values in table TEAMS_ACT_HRS_03-05, thus eliminating the prefix and leaving only the equipment ID for consistency and compatibility purposes.

SQL code	UPDATE [TEAMS_ACT_HRS_03-05] SET EQUIP_ID = right([EQUIP_ID],(len([EQUIP_ID])-2))
	WHERE EQUIP_ID Is Not Null;

qry_create_TEAMS_ACT_HRS_05-07

Function	Runs a make table query to create table TEAMS_ACT_HRS_05-07 from input file Usageactivityhrs-conditionmodel05-07update.txt – TEAMS prior biennium

	activity hours

SQL code	SELECT DISTINCT [tblImportTEAMSActHrs05-07].Ea AS EQUIP_ID, [tblImportTEAMSActHrs05-07].Ea, [tblImportTEAMSActHrs05-07].[Obj Detl] AS OBJ_DETL, [tblImportTEAMSActHrs05-07].[Effective Date] AS EFFECTIVE_DATE, [tblImportTEAMSActHrs05-07].[Transaction Amount] AS TRANSACTION_AMNT, [tblImportTEAMSActHrs05-07].[Transaction Qty] AS TRANSACTION_QTY, [tblImportTEAMSActHrs05-07].[Trans Code] AS TRANSACTION_CODE, [tblImportTEAMSActHrs05-07].Activity AS ACTIVITY, [tblImportTEAMSActHrs05-07].[Responsible Unit] AS RESPONSIBLE_CREW_ID INTO [TEAMS_ACT_HRS_05-07]
	FROM [tblImportTEAMSActHrs05-07]
	WHERE [tblImportTEAMSActHrs05-07].Ea IS NOT NULL;

qry_parse_Ea_TEAMS_ACT_HRS_05-07

Function	Updates EQUIP_ID by parsing the two-prefix Ea (EQUIP_ID) values in table TEAMS_ACT_HRS_05-07, thus eliminating the prefix and leaving only the equipment ID for consistency and compatibility purposes.

SQL code	UPDATE [TEAMS_ACT_HRS_05-07] SET EQUIP_ID = right([EQUIP_ID],(len([EQUIP_ID])-2))
	WHERE EQUIP_ID Is Not Null;

qry_create_PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS

Function	Runs a make table query to create table PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS from input file Excel_KRL_E21666.csv – Prior 12 month equipment operation and repair costs

SQL code	SELECT [tblImportExcel_KRL_E21666-Prev12MoEquipCosts].Prior12Mo_EA AS EQUIP_ID, [tblImportExcel_KRL_E21666-Prev12MoEquipCosts].Prior12Mo_FixedCosts, [tblImportExcel_KRL_E21666-Prev12MoEquipCosts].Prior12Mo_OperCosts, [tblImportExcel_KRL_E21666-Prev12MoEquipCosts].Prior12Mo_RepairCosts, [tblImportExcel_KRL_E21666-Prev12MoEquipCosts].[Prior12Mo_ActHrs-NotUsed] INTO PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS
	FROM [tblImportExcel_KRL_E21666-Prev12MoEquipCosts]
	WHERE ((([tblImportExcel_KRL_E21666-Prev12MoEquipCosts].Prior12Mo_EA) Not Like '*"*'));

qry_parse_Ea_PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS

Function	Updates EQUIP_ID by parsing the two-prefix Ea (EQUIP_ID) values in table PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS

SQL code	UPDATE PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS SET EQUIP_ID = right([EQUIP_ID],(len([EQUIP_ID])-2))
	WHERE EQUIP_ID Is Not Null;

qry_create_LTD_EQUIP_REPAIR_OPER_COSTS

Function	Runs a make table query to create table LTD_EQUIP_REPAIR_OPER_COSTS from input file Excel_KRL_E21667.csv – Life to date repair costs

SQL code	SELECT DISTINCT [tblImportExcel_KRL_E21667-LTDEquipCosts].LTD_EA AS EQUIP_ID, [tblImportExcel_KRL_E21667-LTDEquipCosts].LTD_Repair_Costs AS LTD_REPAIR_COSTS INTO LTD_EQUIP_REPAIR_OPER_COSTS
	FROM [tblImportExcel_KRL_E21667-LTDEquipCosts];

qry_parse_Ea_LTD_EQUIP_REPAIR_OPER_COSTS

Function	Updates EQUIP_ID by parsing the two-prefix Ea (EQUIP_ID) values in table LTD_EQUIP_REPAIR_OPER_COSTS

SQL code	UPDATE LTD_EQUIP_REPAIR_OPER_COSTS SET EQUIP_ID = right([EQUIP_ID],(len([EQUIP_ID])-2))
	WHERE EQUIP_ID Is Not Null;

qry_create_CREW_APPRPRIATION

Function	Runs a make table query to create table CREW_APPROPRIATION from input file HPEMS37R.xls – Eline report showing which appropriation crews are in

SQL code	SELECT DISTINCT [tblImportHPEMS37R-TEAMS ELINE].CREWx AS CREW_ID, [tblImportHPEMS37R-TEAMS ELINE].GROUPx AS GROUP_IN, [tblImportHPEMS37R-TEAMS ELINE].APPNx AS APPN, [tblImportHPEMS37R-TEAMS ELINE].PROGx AS PROG, [tblImportHPEMS37R-TEAMS ELINE].ELEMx AS ELEM, [tblImportHPEMS37R-TEAMS ELINE].FUNDx AS FUND, [tblImportHPEMS37R-TEAMS ELINE].[FUND DETAILx] AS [FUND DETAIL], [tblImportHPEMS37R-TEAMS ELINE].[FUND SOURCEx] AS [FUND SOURCE], [tblImportHPEMS37R-TEAMS ELINE].[FUND PERCENTx] AS [FUND PERCENT] INTO CREW_APPROPRIATION
	FROM [tblImportHPEMS37R-TEAMS ELINE];

qry_PK_EQUIP_MASTER

Function	Sets EQUIP_ID as the primary key in table EQUIP_MASTER
SQL code	ALTER TABLE EQUIP_MASTER ADD PRIMARY KEY (EQUIP_ID);

qry_PK_EQUIP_CLASS

Function	Sets CL_CLASS as the primary key in table EQUIP_CLASS
SQL code	ALTER TABLE EQUIP_CLASS ADD PRIMARY KEY (CLASS);

qry_PK_EQUIP_CREW_ASSN

Function	Sets EQUIP_ID, CREW_ID as the composite primary keys in table EQUIP_CREW_ASSN
SQL code	ALTER TABLE EQUIP_CREW_ASSN ADD PRIMARY KEY (EQUIP_ID, CREW_ID);

qry_PK_CREW

Function	Sets CREW_ID as the primary keys in table CREW
SQL code	ALTER TABLE CREW ADD PRIMARY KEY (CREW_ID);

qry_PK_TEAMS_ACT_HRS_03-05

Function	Sets EQUIP_ID, EFFECTIVE_DATE, TRANSACTION_AMNT, TRANSACTION_QTY, TRANSACTION_CODE, ACTIVITY, RESPONSIBLE_CREW_ID as the composite primary keys in table TEAMS_ACT_HRS_03-05
SQL code	ALTER TABLE [TEAMS_ACT_HRS_03-05] ADD PRIMARY KEY (EQUIP_ID, EFFECTIVE_DATE, TRANSACTION_AMNT, TRANSACTION_QTY, TRANSACTION_CODE, ACTIVITY, RESPONSIBLE_CREW_ID);

qry_PK_TEAMS_ACT_HRS_05-07

Function	Sets EQUIP_ID, EFFECTIVE_DATE, TRANSACTION_AMNT, TRANSACTION_QTY, TRANSACTION_CODE, ACTIVITY, RESPONSIBLE_CREW_ID as the composite primary keys in table TEAMS_ACT_HRS_05-07
SQL code	ALTER TABLE [TEAMS_ACT_HRS_05-07] ADD PRIMARY KEY (EQUIP_ID, EFFECTIVE_DATE, TRANSACTION_AMNT, TRANSACTION_QTY, TRANSACTION_CODE, ACTIVITY, RESPONSIBLE_CREW_ID);

qry_PK_PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS

Function	sets EQUIP_ID as the primary keys in table PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS
SQL code	ALTER TABLE PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS ADD PRIMARY KEY (EQUIP_ID);

qry_PK_LTD_EQUIP_REPAIR_OPER_COSTS

Function	Sets EQUIP_ID as the primary key in table LTD_EQUIP_REPAIR_OPER_COSTS
SQL code	ALTER TABLE LTD_EQUIP_REPAIR_OPER_COSTS ADD PRIMARY KEY (EQUIP_ID);

qry_PK_CREW_APPROPRIATION

Function	Sets CREW_ID as the primary key in table CREW_APPROPRIATION
SQL code	ALTER TABLE CREW_APPROPRIATION ADD PRIMARY KEY (CREW_ID);

qry_create_EQUIP_CONDITION_MODEL_REPORT

Function	Creates table EQUIP_CONDITION_MODEL_REPORT; this is a temporary table used to create the final one. %AgeStd, %CostStd and %UseStd set to 0 and updated in another query based on standards from table tblStandards_Costs.

SQL code	SELECT DISTINCT 0 AS CONDITION_INDEX, " AS CONDITION, EQUIP_MASTER.EQUIP_ID, tblStandards_Costs.Std_RptCat AS EQUIP_CATEGORY, EQUIP_MASTER.CLASS, EQUIP_CLASS.CLASS_DESC, EQUIP_MASTER.EQUIP_DESC, EQUIP_MASTER.STATUS AS EQUIP_STATUS, EQUIP_MASTER.MANUFACTURER, EQUIP_MASTER.MODEL, EQUIP_MASTER.CURRENT_DATE, EQUIP_MASTER.DELIVER_DATE, EQUIP_MASTER.ASSIGN_DATE, ([CURRENT_DATE]-[DELIVER_DATE])/365 AS AGE, tblStandards_Costs.Std_Age, 0 AS [%AgeStd], [AGE]>[Std_Age] AS [>AgeStd], tblStandards_Costs.Std_Measure, EQUIP_MASTER.LIFE_METER, tblStandards_Costs.Std_UseLifeMeter, 0 AS [%UseStd], [LIFE_METER]>[Std_UseLifeMeter] AS [>UseStd], 0 AS LTD_EQUIP_REPAIR_OPER_COSTS, EQUIP_MASTER.ACQ_COST, [LTD_EQUIP_REPAIR_OPER_COSTS]>[ACQ_COST] AS [>CostStd], tblStandards_Costs.Std_ReplacmentCost, 0 AS [%CostStd], 0 AS PRIOR_12MONTH_EQUIP_FIXED_COSTS, 0 AS PRIOR_12MONTH_EQUIP_OPER_COSTS, 0 AS PRIOR_12MONTH_EQUIP_REPAIR_COSTS, [PRIOR_12MONTH_EQUIP_FIXED_COSTS]+[PRIOR_12MONTH_EQUIP_OPER_COSTS]+[PRIOR_12MONTH_EQUIP_REPAIR_COSTS] AS 1YrTotalCost, CREW.CREW_ID, CREW.CREW_NAME, EQUIP_MASTER.AQC_APPN, [ActiveTurn-Ins].[Plan Note] INTO EQUIP_CONDITION_MODEL_REPORT
	FROM [ActiveTurn-Ins] RIGHT JOIN ((tblStandards_Costs RIGHT JOIN (EQUIP_MASTER INNER JOIN EQUIP_CLASS ON EQUIP_MASTER.CLASS = EQUIP_CLASS.CLASS) ON tblStandards_Costs.Std_Class = EQUIP_CLASS.CLASS) INNER JOIN CREW ON EQUIP_MASTER.CREW_ID = CREW.CREW_ID) ON [ActiveTurn-Ins].[Equip ID (Turn-In)] = EQUIP_MASTER.EQUIP_ID;

qry_PK_EQUIP_CONDITION_MODEL_REPORT

Function	Sets EQUIP_ID as the primary key in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ADD PRIMARY KEY (EQUIP_ID);

qry_alter_ECMReport_ACQ_COST_datatype

Function	Alters ACQ_COST data type to CURRENCY in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN ACQ_COST CURRENCY;

qry_alter_ECMReport_AgeStd_datatype

Function	Alters %AgeStd data type to DOUBLE in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [%AgeStd] Double;

qry_alter_ECMReport_CostStd_datatype

Function	Alters %CostStd data type to DOUBLE in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [%CostStd] Double;

qry_alter_ECMReport_LTD_Costs_datatype

Function	Alters LTD_EQUIP_REPAIR_OPER_COSTS data type to CURRENCY in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [LTD_EQUIP_REPAIR_OPER_COSTS] CURRENCY;

qry_alter_ECMReport_Prior12Month_FixedCosts_datatype

Function	Alters PRIOR_12MONTH_EQUIP_FIXED_COSTS data type to CURRENCY in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [PRIOR_12MONTH_EQUIP_FIXED_COSTS] CURRENCY;

qry_alter_ECMReport_Prior12Month_OperCosts_datatype

Function	Alters PRIOR_12MONTH_EQUIP_OPER_COSTS data type to CURRENCY in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [PRIOR_12MONTH_EQUIP_OPER_COSTS] CURRENCY;

qry_alter_ECMReport_Prior12Month_RepairCosts_datatype

Function	Alters PRIOR_12MONTH_EQUIP_REPAIR_COSTS data type to CURRENCY in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [PRIOR_12MONTH_EQUIP_REPAIR_COSTS] CURRENCY;

qry_alter_ECMReport_UseStd_datatype

Function	Alters %UseStd data type to DOUBLE in table

	EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [%UseStd] Double;

qry_alter_ECMReport_CONDITION_INDEX_datatype

Function	Alters CONDITION_INDEX data type to DOUBLE in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [CONDITION_INDEX] Double;

qry_create_%AgeStd

Function	Computes %AgeStd and saves output in table “%AgeStd”

SQL code	SELECT DISTINCT EQUIP_MASTER.EQUIP_ID, ([CURRENT_DATE]-[DELIVER_DATE])/365 AS AGE, tblStandards_Costs.Std_Age, [AGE]/[Std_Age] AS [%AgeStd] INTO [%AgeStd]
	FROM tblStandards_Costs INNER JOIN (EQUIP_MASTER INNER JOIN EQUIP_CLASS ON EQUIP_MASTER.CLASS = EQUIP_CLASS.CLASS) ON tblStandards_Costs.Std_Class = EQUIP_CLASS.CLASS;

qry_create_%CostStd

Function	Computes %CostStd and saves the output in table “%CostStd”

SQL code	SELECT DISTINCT EQUIP_MASTER.EQUIP_ID, EQUIP_MASTER.ACQ_COST, LTD_EQUIP_REPAIR_OPER_COSTS.LTD_REPAIR_COSTS, [LTD_REPAIR_COSTS]/[ACQ_COST] AS [%CostStd] INTO [%CostStd]
	FROM (tblStandards_Costs INNER JOIN (EQUIP_MASTER INNER JOIN EQUIP_CLASS ON EQUIP_MASTER.CLASS = EQUIP_CLASS.CLASS) ON tblStandards_Costs.Std_Class = EQUIP_CLASS.CLASS) INNER JOIN LTD_EQUIP_REPAIR_OPER_COSTS ON EQUIP_MASTER.EQUIP_ID = LTD_EQUIP_REPAIR_OPER_COSTS.EQUIP_ID
	WHERE (((EQUIP_MASTER.ACQ_COST)>0) AND ((LTD_EQUIP_REPAIR_OPER_COSTS.LTD_REPAIR_COSTS)>0));

qry_create_%UseStd

Function	Computes %UseStd and saves the output in table "%UseStd"

SQL code	SELECT DISTINCT EQUIP_MASTER.EQUIP_ID, EQUIP_MASTER.LIFE_METER, tblStandards_Costs.Std_UseLifeMeter, [LIFE_METER]/[Std_UseLifeMeter] AS [%UseStd] INTO [%UseStd]
	FROM tblStandards_Costs INNER JOIN (EQUIP_MASTER INNER JOIN EQUIP_CLASS ON EQUIP_MASTER.CLASS = EQUIP_CLASS.CLASS) ON tblStandards_Costs.Std_Class = EQUIP_CLASS.CLASS
	WHERE (((EQUIP_MASTER.LIFE_METER)>0) AND ((tblStandards_Costs.Std_UseLifeMeter)>0));

qry_update_%AgeStd_ECMReprort

Function	Updates %AgeStd field in table EQUIP_CONDITION_MODEL_REPORT with values computed in query 0
SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT INNER JOIN [%AgeStd] ON EQUIP_CONDITION_MODEL_REPORT.EQUIP_ID = [%AgeStd].EQUIP_ID SET EQUIP_CONDITION_MODEL_REPORT.[%AgeStd] = [%AgeStd]![%AgeStd];

qry_update_%CostStd_ECMReport

Function	Updates %AgeStd field in table EQUIP_CONDITION_MODEL_REPORT with values computed in query 0
SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT INNER JOIN [%CostStd] ON EQUIP_CONDITION_MODEL_REPORT.EQUIP_ID = [%CostStd].EQUIP_ID SET EQUIP_CONDITION_MODEL_REPORT.LTD_EQUIP_REPAIR_OPER_COSTS = [%CostStd]!LTD_REPAIR_COSTS, EQUIP_CONDITION_MODEL_REPORT.[%CostStd] = [%CostStd]![%CostStd];

qry_update_%UseStd_ECMReport

Function	Updates %AgeStd field in table EQUIP_CONDITION_MODEL_REPORT with values computed in query 0
SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT INNER JOIN [%UseStd] ON EQUIP_CONDITION_MODEL_REPORT.EQUIP_ID = [%UseStd].EQUIP_ID SET EQUIP_CONDITION_MODEL_REPORT.[%UseStd] = [%UseStd]![%UseStd];

qry_update_ECMReoprt_Prior_12Month_Costs

Function	Updates PRIOR_12MONTH_EQUIP_FIXED_COSTS field in table EQUIP_CONDITION_MODEL_REPORT with values from normalized table PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS
SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT INNER JOIN PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS ON EQUIP_CONDITION_MODEL_REPORT.EQUIP_ID = PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS.EQUIP_ID SET EQUIP_CONDITION_MODEL_REPORT.PRIOR_12MONTH_EQUIP_FIXED_COSTS = PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS!Prior12Mo_FixedCosts, EQUIP_CONDITION_MODEL_REPORT.PRIOR_12MONTH_EQUIP_OPER_COSTS = PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS!Prior12Mo_OperCosts, EQUIP_CONDITION_MODEL_REPORT.PRIOR_12MONTH_EQUIP_REPAIR_COSTS = PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS!Prior12Mo_RepairCosts;

qry_update_ECMReport_1YrTotalCost

Function	Updates 1YrTotalCost field in table EQUIP_CONDITION_MODEL_REPORT with values from normalized table PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS i.e. sum of [PRIOR_12MONTH_EQUIP_FIXED_COSTS] + [PRIOR_12MONTH_EQUIP_OPER_COSTS] + [PRIOR_12MONTH_EQUIP_REPAIR_COSTS];
SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT SET EQUIP_CONDITION_MODEL_REPORT.[1YrTotalCost] = [PRIOR_12MONTH_EQUIP_FIXED_COSTS]+[PRIOR_12MONTH_EQUIP_OPER_COSTS]+[PRIOR_12MONTH_EQUIP_REPAIR_COSTS];

qry_create_CONDITION_INDEX

Function	Creates CONDITION_INDEX table, used to compute the CI for each equipment. The CI is initially set to zero, and update once the value is computed in a later query. Also, by default, the denominator used in the three parameter (age, usage, and cost) model is set to 3. This value will be updated to 2 later where there is NO use standard (0). All CONDITIONS are set to “Invalid Percent” by default, and updated accordingly once the CI is computed.
SQL code	SELECT EQUIP_CONDITION_MODEL_REPORT.EQUIP_ID, EQUIP_CONDITION_MODEL_REPORT.[%AgeStd], EQUIP_CONDITION_MODEL_REPORT.[%UseStd], EQUIP_CONDITION_MODEL_REPORT.[%CostStd], [%AgeStd]+[%UseStd]+[%CostStd] AS NUMERATOR, 3 AS DENOMINATOR, 0 AS CONDITION_INDEX, 'Invalid Percent' AS CONDITION INTO CONDITION_INDEX FROM EQUIP_CONDITION_MODEL_REPORT;

qry_alter_Condition_Index_datatype

Function	Alters CONDITION_INDEX data type to DOUBLE in table CONDITION_INDEX

SQL code	ALTER TABLE CONDITION_INDEX ALTER COLUMN [CONDITION_INDEX] Double;

qry_update_Condition_Index_Denominator

Function	Updates the Condition Index denominator to 2 where there is NO use standard

SQL code	UPDATE CONDITION_INDEX SET CONDITION_INDEX.DENOMINATOR = 2
	WHERE (((CONDITION_INDEX.[%UseStd])=0));

qry_compute_Condition_Index

Function	Computes the Condition Indices and updates the corresponding field in table CONDITION_INDEX
SQL code	UPDATE CONDITION_INDEX SET CONDITION_INDEX.CONDITION_INDEX = CONDITION_INDEX!NUMERATOR/CONDITION_INDEX!DENOMINATOR;

qry_update_CONDITION_in_tbl_CONDITION_INDEX

Function	Updates the CONDITION field in table CONDITION_INDEX with values determined from the “CONDITION_STATNDARDS” table, depending on the range within which the CI falls, i.e. very good, good, fair, poor, very poor. Any CI values that fall out of range will retain the default CONDITION of “Invalid Percent” as designed in 0
SQL code	

qry_update_EQUIP_MASTER_CONDITION_and_CONDITION_INDEX

Function	Updates CONDITION and CONDITION_INDEX fields in table EQUIP_CONDITION_MODEL_REPORT with values from table CONDITION_INDEX

SQL code	UPDATE CONDITION_INDEX, CONDITION_STANDARDS SET
	CONDITION_INDEX.CONDITION = CONDITION_STANDARDS!CONDITION
	WHERE
	((([CONDITION_INDEX]![CONDITION_INDEX]*100)>([CONDITION_STANDARDS]![COND_INDEX_LOWER_THRESHOLD]) And
	([CONDITION_INDEX]![CONDITION_INDEX]*100)<([CONDITION_STANDARDS]![COND_INDEX_UPPER_THRESHOLD])) AND
	(([CONDITION_INDEX]![CONDITION_INDEX]) Is Not Null And
	([CONDITION_INDEX]![CONDITION_INDEX])>=0));

qry_alter_ECMReport_>AgeStd_datatype

Function	Alters >AgeStd data type to DOUBLE in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [>AgeStd] Text;

qry_alter_ECMReport_>CostStd_datatype

Function	Alters >CostStd data type to DOUBLE in table

	EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [>CostStd] Text;

qry_alter_ECMReport_>UseStd_datatype

Function	Alters >UseStd data type to DOUBLE in table EQUIP_CONDITION_MODEL_REPORT
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT ALTER COLUMN [>UseStd] Text;

qry_update_ECMReport_>AgeStds_to_Yes

Function	Updates “>AgeStds” values from “binary” (-1/0) to Yes/No strings

SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT SET
	EQUIP_CONDITION_MODEL_REPORT.[>AgeStd] = "Yes"
	WHERE (((EQUIP_CONDITION_MODEL_REPORT.[>AgeStd]) Like "-1"));

qry_update_ECMReport_<AgeStds_to_No

Function	Updates “<AgeStds” values from “binary” (-1/0) to Yes/No strings

SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT SET
	EQUIP_CONDITION_MODEL_REPORT.[>AgeStd] = "No"
	WHERE (((EQUIP_CONDITION_MODEL_REPORT.[>AgeStd]) Like "0" And (EQUIP_CONDITION_MODEL_REPORT.[>AgeStd]) Is Not Null));

qry_update_ECMReport_>UseStds_to_Yes

Function	Updates “>UseStds” values from “binary” (-1/0) to Yes/No strings

SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT SET
	EQUIP_CONDITION_MODEL_REPORT.[>UseStd] = "Yes"
	WHERE (((EQUIP_CONDITION_MODEL_REPORT.[>UseStd]) Like "-1"));

qry_update_ECMReport_<UseStds_to_No

Function	Updates “<UseStds” values from “binary” (-1/0) to Yes/No strings

SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT SET
	EQUIP_CONDITION_MODEL_REPORT.[>UseStd] = "No"
	WHERE (((EQUIP_CONDITION_MODEL_REPORT.[>UseStd]) Like "0" And (EQUIP_CONDITION_MODEL_REPORT.[>UseStd]) Is Not Null));

qry_update_ECMReport_>CostStds_to_Yes

Function	Updates “>CostStds” values from “binary” (-1/0) to Yes/No strings

SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT SET
	EQUIP_CONDITION_MODEL_REPORT.[>CostStd] = "Yes"
	WHERE (((EQUIP_CONDITION_MODEL_REPORT.[>CostStd]) Like "-1"));

qry_update_ECMReport_<CostStds_to_No

Function	Updates “<CostStds” values from “binary” (-1/0) to Yes/No strings

SQL code	UPDATE EQUIP_CONDITION_MODEL_REPORT SET
	EQUIP_CONDITION_MODEL_REPORT.[>CostStd] = "No"
	WHERE (((EQUIP_CONDITION_MODEL_REPORT.[>CostStd]) Like "0" And (EQUIP_CONDITION_MODEL_REPORT.[>CostStd]) Is Not Null));

qry_create_EQUIP_CONDITION_REPORT_RANKED

Function	Creates a new ranked table EQUIP_CONDITION_REPORT_RANKED by sorting equipment in table EQUIP_CONDITION_MODEL_REPORT based on Condition Index in descending order
SQL code	SELECT DISTINCT EQUIP_CONDITION_MODEL_REPORT.CONDITION_INDEX, EQUIP_CONDITION_MODEL_REPORT.CONDITION, EQUIP_CONDITION_MODEL_REPORT.EQUIP_ID, EQUIP_CONDITION_MODEL_REPORT.EQUIP_CATEGORY, EQUIP_CONDITION_MODEL_REPORT.CLASS, EQUIP_CONDITION_MODEL_REPORT.CLASS_DESC, EQUIP_CONDITION_MODEL_REPORT.EQUIP_DESC, EQUIP_CONDITION_MODEL_REPORT.EQUIP_STATUS, EQUIP_CONDITION_MODEL_REPORT.MANUFACTURER, EQUIP_CONDITION_MODEL_REPORT.MODEL, EQUIP_CONDITION_MODEL_REPORT.CURRENT_DATE, EQUIP_CONDITION_MODEL_REPORT.DELIVER_DATE, EQUIP_CONDITION_MODEL_REPORT.ASSIGN_DATE, EQUIP_CONDITION_MODEL_REPORT.AGE, EQUIP_CONDITION_MODEL_REPORT.Std_Age, EQUIP_CONDITION_MODEL_REPORT.[%AgeStd], EQUIP_CONDITION_MODEL_REPORT.[>AgeStd], EQUIP_CONDITION_MODEL_REPORT.Std_Measure, EQUIP_CONDITION_MODEL_REPORT.LIFE_METER, EQUIP_CONDITION_MODEL_REPORT.Std_UseLifeMeter, EQUIP_CONDITION_MODEL_REPORT.[%UseStd], EQUIP_CONDITION_MODEL_REPORT.[>UseStd], EQUIP_CONDITION_MODEL_REPORT.LTD_EQUIP_REPAIR_OPER_COSTS, EQUIP_CONDITION_MODEL_REPORT.ACQ_COST, EQUIP_CONDITION_MODEL_REPORT.[>CostStd], EQUIP_CONDITION_MODEL_REPORT.Std_ReplacmentCost, EQUIP_CONDITION_MODEL_REPORT.[%CostStd], EQUIP_CONDITION_MODEL_REPORT.PRIOR_12MONTH_EQUIP_FIXED_COSTS, EQUIP_CONDITION_MODEL_REPORT.PRIOR_12MONTH_EQUIP_OPER_COSTS, EQUIP_CONDITION_MODEL_REPORT.PRIOR_12MONTH_EQUIP_REPAIR_COSTS, EQUIP_CONDITION_MODEL_REPORT.[1YrTotalCost], EQUIP_CONDITION_MODEL_REPORT.CREW_ID, EQUIP_CONDITION_MODEL_REPORT.CREW_NAME, EQUIP_CONDITION_MODEL_REPORT.AQC_APPN, EQUIP_CONDITION_MODEL_REPORT.[Plan Note] INTO EQUIP_CONDITION_MODEL_REPORT_RANKED FROM EQUIP_CONDITION_MODEL_REPORT ORDER BY EQUIP_CONDITION_MODEL_REPORT.CONDITION_INDEX DESC;

qry_alter_ECMREPORT_RANKED_add_RANK

Function	Creates a new field “RANK” in EQUIP_CONDITION_REPORT_ and assigns a sequential rank in ascending order. RANK is an autonumber filed.
SQL code	ALTER TABLE EQUIP_CONDITION_MODEL_REPORT_RANKED ADD COLUMN RANK AUTOINCREMENT;

qry_create_TurnInEquipment_First_Biennium

Function	Filters all SSB Approved (value = -1) records from table tbl-Fleet Approval Information [First Biennium -> (03-05)] into a new table tblTurnInEquipment_First_Biennium. For convenience, the three most recent biennium are referred to as First Biennium, Second Biennium and Third biennium respectfully. Since the biennium values are always changing with every additional biennium, it is better to use the generic names.

SQL code	SELECT IIf(InStr([turn_in equipment number],"-")=3,Mid([turn_in equipment number],1,2) & Mid([turn_in equipment number],4,4),Null) AS [T-I EquipID], "03-05" AS Plan, "" AS PlanNote, [tbl-Fleet Approval Information 03-05].[Appd ID], [tbl-Fleet Approval Information 03-05].Crew, [tbl-Fleet Approval Information 03-05].[Turn_In Equipment Number], [tbl-Fleet Approval Information 03-05].[Turn-In Equipment Second], [tbl-Fleet Approval Information 03-05].[Turn-In Received], [tbl-Fleet Approval Information 03-05].[SSB Approved:], [tbl-Fleet Approval Information 03-05].[Completion Code:], [tbl-Fleet Approval Information 03-05].[New Equipment No:], Now() AS FileCreationDate INTO tblTurnInEquipment_First_Biennium
	FROM [tbl-Fleet Approval Information 03-05]
	WHERE (((IIf(InStr([turn_in equipment number],"-")=3,Mid([turn_in equipment number],1,2) & Mid([turn_in equipment number],4,4),Null)) Is Not Null) AND (([tbl-Fleet Approval Information 03-05].[SSB Approved:])=-1));

qry_create_TurnInEquipment_Second_Biennium

Function	Filters all SSB Approved (value = -1) records from table tbl-Fleet Approval Information [Second Biennium -> (05-07)] into a new table tblTurnInEquipment_Second_Biennium

SQL code	SELECT IIf(InStr([turn_in equipment number],"-")=3,Mid([turn_in equipment number],1,2) & Mid([turn_in equipment number],4,4),Null) AS [T-I EquipID], "05-07" AS Plan, "" AS PlanNote, [tbl-Fleet Approval Information 05-07].[Appd ID], [tbl-Fleet Approval Information 05-07].Crew, [tbl-Fleet Approval Information 05-07].[Turn_In Equipment Number], [tbl-Fleet Approval Information 05-07].[Turn-In Equipment Second], [tbl-Fleet Approval Information 05-07].[Turn-In Received], [tbl-Fleet Approval Information 05-07].[SSB Approved:], [tbl-Fleet Approval Information 05-07].[Completion Code:], [tbl-Fleet Approval Information 05-07].[New Equipment No:], Now() AS FileCreationDate INTO tblTurnInEquipment_Second_Biennium
	FROM [tbl-Fleet Approval Information 05-07]
	WHERE (((IIf(InStr([turn_in equipment number],"-")=3,Mid([turn_in equipment number],1,2) & Mid([turn_in equipment number],4,4),Null)) Is Not Null) AND (([tbl-Fleet Approval Information 05-07].[SSB Approved:])=-1));

qry_create_TurnInEquipment_Third_Biennium

Function	filter all SSB Approved (value = -1) records from table tbl-Fleet Approval Information [Second Biennium -> (07-09)] into a new table tblTurnInEquipment_Third_Biennium

SQL code	SELECT IIf(InStr([turn_in equipment number],"-")=3,Mid([turn_in equipment number],1,2) & Mid([turn_in equipment number],4,4),Null) AS [T-I EquipID], "07-09" AS Plan, "" AS PlanNote, [tbl-Fleet Approval Information 07-09].[Appd ID], [tbl-Fleet Approval Information 07-09].Crew, [tbl-Fleet Approval Information 07-09].[Turn_In Equipment Number], [tbl-Fleet Approval Information 07-09].[Turn-In Equipment Second], [tbl-Fleet Approval Information 07-09].[Turn-In Received], [tbl-Fleet Approval Information 07-09].[SSB Approved:], [tbl-Fleet Approval Information 07-09].[Completion Code:], [tbl-Fleet Approval Information 07-09].[New Equipment No:], Now() AS FileCreationDate INTO tblTurnInEquipment_Third_Biennium
	FROM [tbl-Fleet Approval Information 07-09]
	WHERE (((IIf(InStr([turn_in equipment number],"-")=3,Mid([turn_in equipment number],1,2) & Mid([turn_in equipment number],4,4),Null)) Is Not Null) AND (([tbl-Fleet Approval Information 07-09].[SSB Approved:])=-1));

qry_create_TurnInEquipment_All_Bienniums_Merged

Function	Copies the qry_create_TurnInEquipment_First_Biennium query tblTurnInEquipment_All_Bienniums_Merged and all the data is deleted in the subsequent step to retain *only* the table structure.
SQL code	SELECT tblTurnInEquipment_First_Biennium.[T-I EquipID], tblTurnInEquipment_First_Biennium.Plan, tblTurnInEquipment_First_Biennium.PlanNote, "" AS [Plan Note], tblTurnInEquipment_First_Biennium.[Appd ID], tblTurnInEquipment_First_Biennium.Crew, tblTurnInEquipment_First_Biennium.[Turn_In Equipment Number], tblTurnInEquipment_First_Biennium.[Turn-In Equipment Second], tblTurnInEquipment_First_Biennium.[Turn-In Received], tblTurnInEquipment_First_Biennium.[SSB Approved:], tblTurnInEquipment_First_Biennium.[Completion Code:], tblTurnInEquipment_First_Biennium.[New Equipment No:], tblTurnInEquipment_First_Biennium.FileCreationDate INTO tblTurnInEquipment_All_Bienniums_Merged FROM tblTurnInEquipment_First_Biennium;

qry_delete_all_records_TurnInEquipment_All_Bienniums_Merged

Function	Deletes all the data in the subsequent step to retain only the table structure.
SQL code	DELETE * FROM tblTurnInEquipment_All_Bienniums_Merged;

qry_append_TurnInEquipment_First_Biennium

Function	Append data from tblTurnInEquipment_First_Biennium to merger table tblTurnInEquipment_All_Bienniums_Merged.
SQL code	INSERT INTO tblTurnInEquipment_All_Bienniums_Merged ([T-I EquipID], Plan,

	PlanNote, [Appd ID], Crew, [Turn_In Equipment Number], [Turn-In Equipment Second], [Turn-In Received], [SSB Approved:], [Completion Code:], [New Equipment No:], FileCreationDate) SELECT tblTurnInEquipment_First_Biennium.[T-I EquipID], tblTurnInEquipment_First_Biennium.Plan, tblTurnInEquipment_First_Biennium.PlanNote, tblTurnInEquipment_First_Biennium.[Appd ID], tblTurnInEquipment_First_Biennium.Crew, tblTurnInEquipment_First_Biennium.[Turn_In Equipment Number], tblTurnInEquipment_First_Biennium.[Turn-In Equipment Second], tblTurnInEquipment_First_Biennium.[Turn-In Received], tblTurnInEquipment_First_Biennium.[SSB Approved:], tblTurnInEquipment_First_Biennium.[Completion Code:], tblTurnInEquipment_First_Biennium.[New Equipment No:], tblTurnInEquipment_First_Biennium.FileCreationDate FROM tblTurnInEquipment_First_Biennium;

qry_append_TurnInEquipment_Second_Biennium

Function	Append data from tblTurnInEquipment_First_Biennium to merger table tblTurnInEquipment_All_Bienniums_Merged.
SQL code	INSERT INTO tblTurnInEquipment_All_Bienniums_Merged ([T-I EquipID], Plan, PlanNote, [Appd ID], Crew, [Turn_In Equipment Number], [Turn-In Equipment Second], [Turn-In Received], [SSB Approved:], [Completion Code:], [New Equipment No:], FileCreationDate) SELECT tblTurnInEquipment_Second_Biennium.[T-I EquipID], tblTurnInEquipment_Second_Biennium.Plan, tblTurnInEquipment_Second_Biennium.PlanNote, tblTurnInEquipment_Second_Biennium.[Appd ID], tblTurnInEquipment_Second_Biennium.Crew, tblTurnInEquipment_Second_Biennium.[Turn_In Equipment Number], tblTurnInEquipment_Second_Biennium.[Turn-In Equipment Second], tblTurnInEquipment_Second_Biennium.[Turn-In Received], tblTurnInEquipment_Second_Biennium.[SSB Approved:], tblTurnInEquipment_Second_Biennium.[Completion Code:], tblTurnInEquipment_Second_Biennium.[New Equipment No:], tblTurnInEquipment_Second_Biennium.FileCreationDate FROM tblTurnInEquipment_Second_Biennium;

qry_append_TurnInEquipment_Third_Biennium

Function	Append data from tblTurnInEquipment_First_Biennium to merger table tblTurnInEquipment_All_Bienniums_Merged
SQL code	INSERT INTO tblTurnInEquipment_All_Bienniums_Merged ([T-I EquipID], Plan, PlanNote, [Appd ID], Crew, [Turn_In Equipment Number], [Turn-In Equipment Second], [Turn-In Received], [SSB Approved:], [Completion Code:], [New Equipment No:], FileCreationDate)

	SELECT tblTurnInEquipment_Third_Biennium.[T-I EquipID], tblTurnInEquipment_Third_Biennium.Plan, tblTurnInEquipment_Third_Biennium.PlanNote, tblTurnInEquipment_Third_Biennium.[Appd ID], tblTurnInEquipment_Third_Biennium.Crew, tblTurnInEquipment_Third_Biennium.[Turn_In Equipment Number], tblTurnInEquipment_Third_Biennium.[Turn-In Equipment Second], tblTurnInEquipment_Third_Biennium.[Turn-In Received], tblTurnInEquipment_Third_Biennium.[SSB Approved:], tblTurnInEquipment_Third_Biennium.[Completion Code:], tblTurnInEquipment_Third_Biennium.[New Equipment No:], tblTurnInEquipment_Third_Biennium.FileCreationDate FROM tblTurnInEquipment_Third_Biennium;

qry_update_TempReassignmen

Function	Update PlanNote field in merged table tblTurnInEquipment_All_Bienniums_Merged to "Tmp Reasignment" where records from tblTurnIn-TempReassignment (external table, not part of original input files) match, i.e. same EQUIP_ID.

SQL code	UPDATE tblTurnInEquipment_All_Bienniums_Merged INNER JOIN [tblTurnIn-TempReassignment] ON tblTurnInEquipment_All_Bienniums_Merged.[Turn_In Equipment Number] = [tblTurnIn-TempReassignment].EQUIPMENT
	SET tblTurnInEquipment_All_Bienniums_Merged.PlanNote = "Tmp ReAssgmt"
	WHERE ((([tblTurnIn-TempReassignment].EQUIPMENT) Is Not Null));

tblTurnInEquipment_All_Bienniums_Merged_Crosstab

Function	Create a crosstab to make each biennium a separate field and match corresponding equipsment (EQUIP_ID) to corresponding biennium when they were marked in Fleet Approval Information (Ed's linked files).

SQL code	UPDATE tblTurnInEquipment_All_Bienniums_Merged INNER JOIN [tblTurnIn-TempReassignment] ON tblTurnInEquipment_All_Bienniums_Merged.[Turn_In Equipment Number] = [tblTurnIn-TempReassignment].EQUIPMENT SET tblTurnInEquipment_All_Bienniums_Merged.PlanNote = "Tmp ReAssgmt"
	WHERE ((([tblTurnIn-TempReassignment].EQUIPMENT) Is Not Null));

qry_create_ActiveTurn-Ins

Function	Create the ActiveTurn-Ins table used for zero ranking equipment that should not be considered for replacement in the final report.
SQL code	SELECT DISTINCT CREW.CREW_ID, CREW.CREW_NAME, EQUIP_MASTER.EQUIP_ID AS [Equip ID (Turn-In)], EQUIP_MASTER.STATUS, EQUIP_MASTER.CLASS, EQUIP_MASTER.EQUIP_DESC, IIf([03-05]>0,"03-05") AS [Plan03-5], IIf([05-07]>0,"05-07") AS [Plan05-7], IIf([07-09]>0,"07-09") AS [Plan07-9], tblTurnInEquipment_All_Bienniums_Merged.PlanNote, Trim(Trim([plan03-05]) & " " & Trim([plan05-07]) & " " & Trim([plan07-09]) & " " & Trim([PlanNote])) AS [Plan Note],

	tblTurnInEquipment_All_Bienniums_Merged_Crosstab.[Total Of Appd ID] AS TotalOnPlan INTO [ActiveTurn-Ins]
	FROM tblTurnInEquipment_All_Bienniums_Merged_Crosstab INNER JOIN ((EQUIP_MASTER INNER JOIN CREW ON EQUIP_MASTER.CREW_ID=CREW.CREW_ID) INNER JOIN tblTurnInEquipment_All_Bienniums_Merged ON EQUIP_MASTER.EQUIP_ID=tblTurnInEquipment_All_Bienniums_Merged.[T-I EquipID]) ON tblTurnInEquipment_All_Bienniums_Merged_Crosstab.[T-I EquipID]=EQUIP_MASTER.EQUIP_ID
	WHERE (((EQUIP_MASTER.STATUS)="16" Or (EQUIP_MASTER.STATUS)="17"));

qry__create_EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP

Function	Creates final report with zero ranked equip as well as regular rank, but NOT ordered
SQL code	SELECT IIf((EQUIP_CONDITION_MODEL_REPORT_RANKED.[Plan Note] Like (Select SQLQryBiennium FROM BIENNIUM WHERE CurrentBiennium IS NOT NULL) Or EQUIP_CONDITION_MODEL_REPORT_RANKED.[Plan Note] Like "*Tmp*"),0,(EQUIP_CONDITION_MODEL_REPORT_RANKED.RANK)) AS RANK, EQUIP_CONDITION_MODEL_REPORT_RANKED.CONDITION_INDEX, EQUIP_CONDITION_MODEL_REPORT_RANKED.CONDITION, EQUIP_CONDITION_MODEL_REPORT_RANKED.EQUIP_ID, EQUIP_CONDITION_MODEL_REPORT_RANKED.EQUIP_CATEGORY, EQUIP_CONDITION_MODEL_REPORT_RANKED.CLASS, EQUIP_CONDITION_MODEL_REPORT_RANKED.CLASS_DESC, EQUIP_CONDITION_MODEL_REPORT_RANKED.EQUIP_DESC, EQUIP_CONDITION_MODEL_REPORT_RANKED.EQUIP_STATUS, EQUIP_CONDITION_MODEL_REPORT_RANKED.MANUFACTURER, EQUIP_CONDITION_MODEL_REPORT_RANKED.MODEL, EQUIP_CONDITION_MODEL_REPORT_RANKED.CURRENT_DATE, EQUIP_CONDITION_MODEL_REPORT_RANKED.DELIVER_DATE, EQUIP_CONDITION_MODEL_REPORT_RANKED.ASSIGN_DATE, EQUIP_CONDITION_MODEL_REPORT_RANKED.AGE, EQUIP_CONDITION_MODEL_REPORT_RANKED.Std_Age, EQUIP_CONDITION_MODEL_REPORT_RANKED.[%AgeStd], EQUIP_CONDITION_MODEL_REPORT_RANKED.[>AgeStd], EQUIP_CONDITION_MODEL_REPORT_RANKED.Std_Measure, EQUIP_CONDITION_MODEL_REPORT_RANKED.LIFE_METER, EQUIP_CONDITION_MODEL_REPORT_RANKED.Std_UseLifeMeter, EQUIP_CONDITION_MODEL_REPORT_RANKED.[%UseStd], EQUIP_CONDITION_MODEL_REPORT_RANKED.[>UseStd], EQUIP_CONDITION_MODEL_REPORT_RANKED.LTD_EQUIP_REPAIR_OPER_COSTS, EQUIP_CONDITION_MODEL_REPORT_RANKED.ACQ_COST, EQUIP_CONDITION_MODEL_REPORT_RANKED.[>CostStd], EQUIP_CONDITION_MODEL_REPORT_RANKED.Std_ReplacmentCost, EQUIP_CONDITION_MODEL_REPORT_RANKED.[%CostStd], EQUIP_CONDITION_MODEL_REPORT_RANKED.PRIOR_12MONTH_EQUIP_FIXED_COSTS,

	EQUIP_CONDITION_MODEL_REPORT_RANKED.PRIOR_12MONTH_EQUIP_OPER_COSTS, EQUIP_CONDITION_MODEL_REPORT_RANKED.PRIOR_12MONTH_EQUIP_REPAIR_COSTS, EQUIP_CONDITION_MODEL_REPORT_RANKED.[1YrTotalCost], EQUIP_CONDITION_MODEL_REPORT_RANKED.CREW_ID, EQUIP_CONDITION_MODEL_REPORT_RANKED.CREW_NAME, EQUIP_CONDITION_MODEL_REPORT_RANKED.AQC_APPN, EQUIP_CONDITION_MODEL_REPORT_RANKED.[Plan Note] INTO EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP FROM EQUIP_CONDITION_MODEL_REPORT_RANKED;

qry_create_EQUIP_CONDITION_MODEL_REPORT_FINAL

Function	Creates final report (EQUIP_CONDITION_MODEL_REPORT_FINAL) with zero ranked equip as well as regular rank, ordered by RANK and EQUIP_ID, from EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP table.
SQL code	SELECT EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.RANK, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.CONDITION_INDEX, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.CONDITION, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.EQUIP_ID, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.EQUIP_CATEGORY, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.CLASS, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.CLASS_DESC, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.EQUIP_DESC, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.EQUIP_STATUS, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.MANUFACTURER, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.MODEL, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.CURRENT_DATE, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.DELIVER_DATE, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.ASSIGN_DATE, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.AGE, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.Std_Age, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[%AgeStd], EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[>AgeStd], EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.Std_Measure, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.LIFE_METER, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.Std_UseLifeMeter, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[%UseStd], EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[>UseStd], EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.LTD_EQUIP_REPAIR_OPER_COSTS, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.ACQ_COST, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[>CostStd], EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.Std_ReplacmentCost, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[%CostStd], EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.PRIOR_12MONTH_EQUIP_FIXED_COSTS, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.PRIOR_12MONTH_

	EQUIP_OPER_COSTS, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.PRIOR_12MONTH_ EQUIP_REPAIR_COSTS, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[1YrTotalCost], EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.CREW_ID, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.CREW_NAME, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.AQC_APPN, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.[Plan Note] INTO EQUIP_CONDITION_MODEL_REPORT_FINAL FROM EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP ORDER BY EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.RANK, EQUIP_CONDITION_MODEL_REPORT_FINAL_TEMP.EQUIP_ID;

MACROS

A macro is a list of one or more actions that work together to carry out a particular task in response to an event.[2] In this application, macros consist mainly of queries representing event procedures whose sequence must be preserved and hence the need to encapsulate them in a macro. No changes should be made to the macros. If the need arises to make adjustments, this should be made to the individual query in question.

The table below provides a summary of the macros run in this application (Note: the queries are referenced by corresponding numbers from sub-section 0):

Macros included in the condition model

Macro name	Task	Queries (actions)
createNewTables	Convert input data (flat files) into MS Access tables in the database	0 to 0
addPrimaryKeys	Creates primary and foreign keys in the new tables, which creates a relational database	0 to 0
alter_ECMReport_Column_datatypes	Alters condition model report field datatypes for compatibility with parent datatypes. (Source data contains different datatypes for similar fields in different files, hence the necessity for this conversion).	0 to 0
compute_Condition_Index	Computes the condition index used to rank equipment for replacement	0 to 0
update_ECMReport_Stds_Yes_No_Columns	Updates greater than/less than age, cost and use standard values from	0 to 0

[2] Virginia Anderson (2003), *How to Do Everything with Microsoft Office 2003*, p. 358

	“binary” (-1/0) to Yes/No strings	
create_ActiveTurn-Ins	Creates the ActiveTurn-Ins table, which is used to determine equipment that should be zero-ranked in the final report, i.e. equipment that should NOT be considered for replacement in the current biennium	0 to 0
createConditionModelReport	Creates final condition model report, ranked and zero-ranked where required	0 to 0

MODULES

The application uses only one module named *modAccessDBToolBox* (see Appendix 0). This module is identical to that used in the old Fleet Replacement Model. It performs various functions, including enabling connectivity to the database (DB) by defining the DB path, which is a global variable applied by other classes in the application.

The existing module was left intact to enable further development of the application in terms of security restrictions such as passwords, usernames and remote access to the application. Table 2 provides a summary of the functions contained in the *modAccessDBToolBox* module.

(Note: these functions and their definitions are reproduced verbatim as per the original author of the module in the old Fleet Replacement Plan. See Appendix 0 for complete VB code).

Functions included in the module *modAccessDBToolBox*

Function: GetCurrentDBPath
Purpose: Return the current Access database path

Function: GetCurrentDBName
Purpose: Return the current Access database name

Function: UpdateDBLinks
Purpose: This function updates all of the current database's links with a given database, including those links already established. It will link all tables except for system tables. Will only affect those tables that match those of the given database. If the table names are the same they will will be deleted an then linked. Those tables that do not match those table names from the given database will be left

untouched.	
Function: **Purpose:**	CheckFileExists Checks if a file already exists in the database
Function: **Purpose:**	CreateDBLink This function creates a link to the current database
Function: **Purpose:**	DisplayStatusMessage This updates the Access status bar with a given message
Function: **Purpose:**	ChangeProperty Allows change of database properties

SOFTWARE MAINTENANCE (ADMINISTRATIVE GUIDE)

This section provides guidance on how to update the static input files. This function will ONLY be available to authorized administrative users.

UPDATING STATIC INPUT FILES

This refers to necessary updates to static input files described in Section 5.1.3. There are two ways to perform the updates:

1. Import the new data set into the condition model database. As a precaution, rename the existing table in the database, e.g. by adding a prefix “x” to the filename. This ensures that the data is not lost. Import the new data file as follows: *File→Get External Data→Import*, browse for the file with the updated data. Go through the wizard steps and make sure to name the imported table **exactly** the same as the original name of the one being replaced. This is the preferred option since the administrator does not handle the data itself and therefore chances of data corruption are minimal, if any.
2. Change the data directly in the relevant database table. This may be risky as the administrator may inadvertently change the wrong data, thus corrupting the database.

DATABASE CONNECTION MODULE (MODACCESSDBTOOLBOX)

```
Option Compare Database
Option Explicit
```

```
'****************************************************************
'****************************************************************
'*****  Function: GetCurrentDBPath
'*****  Purpose:  Return the current Access database path
'*****  Inputs:     None
'*****  Outputs:    None.
'*****  Return Value: String that includes
'*****  Errors:     None
'*****  Author:     Linda M. Parets
'*****  Created:    05-16-2001
'*****************************************************************
'*****************************************************************
Function GetCurrentDBPath() As String

   Dim strDBPath As String
   Dim strDBFile As String

   strDBPath = CurrentDb.Name
   strDBFile = Dir(strDBPath)

   GetCurrentDBPath = Left(strDBPath, Len(strDBPath) - Len(strDBFile))

End Function

'****************************************************************
'****************************************************************
'*****  Function: GetCurrentDBName
'*****  Purpose:  Return the current Access database name
'*****  Inputs:     None
'*****  Outputs:    None.
'*****  Return Value: The database name
'*****  Errors:     None
'*****  Author:     Linda M. Parets
'*****  Created:    05-16-2001
'*****************************************************************
'*****************************************************************

Function GetCurrentDBName() As String

   Dim intIdx As Integer
   Dim strReturnValue As String

   On Error GoTo GetCurrentDB_Err

   For intIdx = Len(CurrentDb.Name) To 1 Step -1
      If Mid$(CurrentDb.Name, intIdx, 1) = "\" Then
         strReturnValue = Right$(CurrentDb.Name, Len(CurrentDb.Name) - intIdx)
         Exit For
      End If
   Next intIdx

GetCurrentDB_Exit:
   GetCurrentDBName = strReturnValue
   Exit Function
```

```
GetCurrentDB_Err:
   strReturnValue = ""
   Err.Raise Err.Number
   Resume GetCurrentDB_Exit

End Function

'******************************************************************
'******************************************************************
'*****  Function: UpdateDBLinks
'*****  Purpose:    This function updates all of the current database's
'*****              links with a given database, including those links
'*****              already established. It will link all tables except
'*****              for system tables.
'*****           Will only effect those tables that match those of the
'*****              given database. If the table names are the same they will
'*****              will be deleted an then linked.  Those tables that do not
'*****              match those table names from the given database will be
'*****              left untouched.
'*****  Inputs:
'*****           strDBPathAndName - the full name and path of database to link
'*****              the current database to.
'*****  Outputs:   None.
'*****  Return Value: None.
'*****  Errors:     Will reraise any unhandled error,
'*****              these will should be handled in the calling function
'*****           1101 + vbObjectError - Could not create link
'*****           1102 + vbObjectError - Could not delete link
'*****           1103 + vbObjectError - Database was not found
'*****  Author:     Linda M. Parets
'*****  Created:    05-15-2001
'*******************************************************************
'*******************************************************************

Public Function UpdateDBLinks(ByVal strDBPathAndName As String)

   On Error GoTo UpdateDBLinks_Exit

   Dim objWS1, objWS2 As Workspace
   Dim objDB_Client, objDB_Data As Database
   Dim tdData, tdClient As TableDef
   Dim blnCreateLink As Boolean
   Dim blnReturnValue As Boolean
   Dim lngErrorNum  As Long
   Dim strFieldName As String

   Const WORKSPACE2_NAME = "DBWorkspace2"
   Const LOGIN_USERID = "ADMIN"

   Const ERR_NO_ERROR = 0
   Const ERR_NO_CREATE = 1101 + vbObjectError
   Const ERR_NO_DELETE = 1102 + vbObjectError
   Const ERR_NO_DATABASE = 1103 + vbObjectError

   blnReturnValue = True
```

```
lngErrorNum = ERR_NO_ERROR

'Check if given database exists
If CheckFileExists(strDBPathAndName) Then

   Set objDB_Client = CurrentDb
   Set objWS2 = DBEngine.CreateWorkspace(WORKSPACE2_NAME, LOGIN_USERID, "")
   Set objDB_Data = objWS2.OpenDatabase(strDBPathAndName)

   ' Check each table in the given database
   For Each tdData In objDB_Data.TableDefs
      If Mid(tdData.Name, 1, 4) <> "~TMP" Then
         If Mid(tdData.Name, 1, 4) <> "MSYS" Then
           blnCreateLink = True

            ' Compare tables with the current database
            For Each tdClient In objDB_Client.TableDefs
               If Mid(tdClient.Name, 1, 4) <> "MSYS" Then
                  If tdClient.Name = tdData.Name Then
                     blnCreateLink = False
                     Exit For
                  End If
               End If
            Next tdClient

            'Refresh Link
            If blnCreateLink Then
               If Not (CreateDBLink(strDBPathAndName, tdData.Name)) Then
                  strFieldName = tdData.Name
                  lngErrorNum = ERR_NO_CREATE
                  Exit For
               End If
            Else
               'Remove old link and update link
               If Not (DeleteDBTable(objDB_Client, tdData)) Then
                  strFieldName = tdData.Name
                  lngErrorNum = ERR_NO_DELETE
                  Exit For
               Else
                  If Not (CreateDBLink(strDBPathAndName, tdData.Name)) Then
                     strFieldName = tdData.Name
                     lngErrorNum = ERR_NO_CREATE
                     Exit For
                  End If
               End If
            End If

         End If
     End If

      If Not (lngErrorNum = ERR_NO_ERROR) Then
         Exit For
      End If

  Next tdData
```

```
   Else
      'Database not found
      lngErrorNum = ERR_NO_DATABASE
   End If

UpdateDBLinks_Exit:
   Set tdData = Nothing
   Set tdClient = Nothing
   Set objDB_Data = Nothing
   Set objWS2 = Nothing
   Set objWS1 = Nothing
   Set objDB_Client = Nothing

   'Raise Errors
   If Err.Number Then
      blnReturnValue = False
      Err.Raise Err.Number
   Else
      Select Case lngErrorNum
      Case ERR_NO_CREATE
         Err.Raise ERR_NO_CREATE, "UpdateDBLinks", "Database Link could not be created: " & strFieldName
      Case ERR_NO_DELETE
         Err.Raise ERR_NO_DELETE, "UpdateDBLinks", "Database Table could not be deleted: " & strFieldName
      Case ERR_NO_DATABASE
         Err.Raise ERR_NO_DATABASE, "UpdateDBLinks", "Database could not be found: " &
strDBPathAndName
      Case Else
         ' No Error
      End Select
   End If
   Exit Function

End Function

Function CheckFileExists(strDBPathAndName As String) As Boolean
   Dim strout As String
   Dim blnreturn As Boolean
   strout = Dir(strDBPathAndName)
   If strout <> "" Then
      blnreturn = True
   Else
      blnreturn = False
   End If
   CheckFileExists = blnreturn

End Function

'****************************************************************
'****************************************************************
'*****  Function:   DeleteDBTable
'*****  Purpose:    This function deletes a give table from the
'*****              current database
'*****  Inputs:
'*****            objDB - the database to delete the table from
```

```
'*****          objtd - The table definition name to be deleted
'*****  Outputs:   none
'*****  Return Value:   True if link was succesfull, false if it was not
'*****  Errors:    None
'*****  Author:    Linda M. Parets
'*****  Created:   05-16-2001
'*****************************************************************
'*****************************************************************
Function DeleteDBTable(ByVal objdb As Database, ByVal objTD As TableDef) As Boolean

   On Error GoTo DeleteTable_Err

   Dim blnReturnValue As Boolean

   blnReturnValue = True

   objdb.TableDefs.Delete objTD.Name

DeleteTable_Exit:
   DeleteDBTable = blnReturnValue
   Exit Function

DeleteTable_Err:
   blnReturnValue = False
   Resume DeleteTable_Exit

End Function


'****************************************************************
'****************************************************************
'*****  Function:  CreateDBLink
'*****  Purpose:   This function creates a link to the current database
'*****  Inputs:
'*****          strDBNameAndPath - the database to be linked to the current,
'*****             full path and name
'*****          strTDName - The table definition name to be linked
'*****  Outputs:   None.
'*****  Return Value:   True if link was succesfull, false if it was not
'*****  Errors:    None
'*****  Author:    Linda M. Parets
'*****  Created:   05-15-2001
'*****************************************************************
'*****************************************************************
Function CreateDBLink(ByVal strDBNameAndPath As String, ByVal strTDname As String) As Boolean

   On Error GoTo CreateDBLink_Err

   Dim blnReturnValue As Boolean

   Const APP_APPLICATION_NAME = "Microsoft Access"

   blnReturnValue = True

   DoCmd.TransferDatabase acLink, APP_APPLICATION_NAME, _
```

```
    strDBNameAndPath, acTable, strTDname, strTDname

CreateDBLink_Exit:
   CreateDBLink = blnReturnValue
   Exit Function

CreateDBLink_Err:
   blnReturnValue = False
   Resume CreateDBLink_Exit

End Function

'******************************************************************
'******************************************************************
'*****  Function:   DisplayStatusMessage
'*****  Purpose:    This updates the Access status bar with a given message
'*****  Inputs:     strMessage - the message to display
'*****  Outputs:    None.
'*****  Return Value:   True if successfull, false if not
'*****  Errors:     None
'*****  Author:     Linda M. Parets
'*****  Created:    05-16-2001
'*******************************************************************
'*******************************************************************
Function DisplayStatusMessage(ByVal strMessage As String) As Boolean

   On Error GoTo DisplayStatus_Err

   Dim blnReturnValue As Boolean

   blnReturnValue = True

   Call SysCmd(acSysCmdSetStatus, strMessage)

DisplayStatus_Exit:
   DisplayStatusMessage = blnReturnValue
   Exit Function

DisplayStatus_Err:
   blnReturnValue = False
   Resume DisplayStatus_Exit

End Function

Function ChangeProperty(strPropName As String, varPropType As Variant, varPropValue As Variant) As Integer
' Allows you to change the database properties
'  Original code taken from a microsoft sample - LMP 10/27/2000

   Dim dbs As Database, prp As Property
   Const conPropNotFoundError = 3270

   Set dbs = CurrentDb
   On Error GoTo Change_Err
   dbs.Properties(strPropName) = varPropValue
   ChangeProperty = True
```

```
Change_Bye:
    Exit Function

Change_Err:
    If Err = conPropNotFoundError Then  ' Property not found.
        Set prp = dbs.CreateProperty(strPropName, varPropType, varPropValue)
        dbs.Properties.Append prp
        Resume Next
    Else
        ' Unknown error.
        ChangeProperty = False
        Resume Change_Bye
    End If
End Function
```

VB CODE FOR MAIN MENU FORM (FORM_FRMIMPORTTABLES)

```
Option Compare Database
Dim maxProgressBar As Integer
Dim currBiennium As String

Private Sub btn_ConfirmBiennium_LostFocus()

    'Hide the combo box
    ComboBiennium.Visible = False


End Sub

Private Sub ComboBiennium_Change()


    'Display a new label with the selected bienium, align it to the combo box position
    lbl_SelectedBiennium.Caption = ComboBiennium.Value
    lbl_SelectedBiennium.Visible = True

    'Create a variable for the selected biennium
    currBiennium = ComboBiennium.Value

    ConfirmBienniumMsg = "Please confirm that ALL the input files are from the selected " & currBiennium & "
biennium?"

    answer = MsgBox(ConfirmBienniumMsg, vbExclamation + vbYesNo, "Confirm biennium")
    If answer = vbYes Then
        'show import file button and label
        lbl_Import_Source_Files.Visible = True
        Command0.Visible = True

        'Supress MS Access warnings
        DoCmd.SetWarnings False

        'set current biennium values in table BIENNIUM to "Null"
        sqlResetBiennium = "UPDATE BIENNIUM SET CurrentBiennium = NULL, SQLQryBiennium = NULL"

        'MsgBox (sqlResetBiennium)
        DoCmd.RunSQL (sqlResetBiennium)

        'identify current biennium in DB by updating CurrentBiennium & SQLQryBiennium fields in BIENNIUM
table
        'NB: chr(34) represents the double quote character within a string, chr(39) single quotes

        sqlUpdateBiennium = "UPDATE BIENNIUM SET CurrentBiennium = 'Yes', " _
        & "SQLQryBiennium = " & Chr(39) & "*" & currBiennium & "*" & Chr(39) & " WHERE Biennium like " &
"'" & currBiennium & "'"

        'MsgBox (sqlUpdateBiennium)
        DoCmd.RunSQL (sqlUpdateBiennium)
```

```
   End
   Else
      'DoCmd.Close acForm, "frmImportTables"
      DoCmd.OpenForm "frmImportTables"
   End If


End Sub

Private Sub Command18_Click()

'Open Reports form
   DoCmd.OpenForm ("frmReports")

End Sub

Private Sub Form_Load()


   'Show control buttons
   ComboBiennium.Visible = True
   Command0.Visible = False
   Command2.Visible = False
   Command18.Visible = False
   btn_Refresh.Visible = False
   boxProgressBar.Visible = False
   rectProgressBar.Visible = False


   'Hide labels until prior process is complete
   lbl_Biennium.Visible = True
   lbl_SelectedBiennium.Visible = False
   lbl_Import_Source_Files.Visible = False
   lbl_Process_Data.Visible = False
   lbl_view_condition_report.Visible = False
   lbl_run_model_details.Visible = False


   'Clear labels and/or put default captions
   lbl_Biennium.Caption = "Select Biennium: "
   lbl_Import_Source_Files.Caption = "Click to import source files from source directory"
   lbl_Process_Data.Caption = "Click to create normalized database from input files"
   lbl_view_condition_report.Caption = "Click to view equipment condition model reports"
   lbl_New_Source_Files_Path.Caption = ""
   lbl_Progress_Bar.Caption = ""
   lbl_run_model_details.Caption = ""


   'set progress bar width to zero
   Me.rectProgressBar.Width = 0

   'position progress bar to align with first button (import data button)

   Me.rectProgressBar.Top = Me.Command0.Top
```

```
    Me.boxProgressBar.Top = Me.Command0.Top
    Me.lbl_Progress_Bar.Top = Me.Command0.Top

    'show login date and time
    lbl_DateTime.Caption = "Login time:" & " " & Time & "   " & Date


    'hide all the import file labels
    lbl_List_Import_Filename1.Caption = ""
    lbl_List_Import_Filename2.Caption = ""
    lbl_List_Import_Filename3.Caption = ""
    lbl_List_Import_Filename4.Caption = ""
    lbl_List_Import_Filename5.Caption = ""
    lbl_List_Import_Filename6.Caption = ""
    lbl_List_Import_Filename7.Caption = ""
    lbl_List_Import_Filename8.Caption = ""
    lbl_List_Import_Filename9.Caption = ""

    'define forcolr of  labels as yellow
    lbl_List_Import_Filename1.ForeColor = 65535
    lbl_List_Import_Filename2.ForeColor = 65535
    lbl_List_Import_Filename3.ForeColor = 65535
    lbl_List_Import_Filename4.ForeColor = 65535
    lbl_List_Import_Filename5.ForeColor = 65535
    lbl_List_Import_Filename6.ForeColor = 65535
    lbl_List_Import_Filename7.ForeColor = 65535
    lbl_List_Import_Filename8.ForeColor = 65535
    lbl_List_Import_Filename9.ForeColor = 65535


End Sub

Private Sub Command0_Click()

On Error Resume Next

'define file path
    filepath = GetCurrentDBPath & "Reports\"


'hide all the import file labels
    lbl_List_Import_Filename1.Caption = ""
    lbl_List_Import_Filename2.Caption = ""
    lbl_List_Import_Filename3.Caption = ""
    lbl_List_Import_Filename4.Caption = ""
    lbl_List_Import_Filename5.Caption = ""
    lbl_List_Import_Filename6.Caption = ""
    lbl_List_Import_Filename7.Caption = ""
    lbl_List_Import_Filename8.Caption = ""
    lbl_List_Import_Filename9.Caption = ""
    lbl_run_model_details.Caption = ""

'define forcolr of  labels as yellow
    lbl_List_Import_Filename1.ForeColor = 65535
    lbl_List_Import_Filename2.ForeColor = 65535
```

```
   lbl_List_Import_Filename3.ForeColor = 65535
   lbl_List_Import_Filename4.ForeColor = 65535
   lbl_List_Import_Filename5.ForeColor = 65535
   lbl_List_Import_Filename6.ForeColor = 65535
   lbl_List_Import_Filename7.ForeColor = 65535
   lbl_List_Import_Filename8.ForeColor = 65535
   lbl_List_Import_Filename9.ForeColor = 65535


'Change the label caption to "selected biennium"
   lbl_Biennium.Caption = "Selected Biennium:"

'Hide the combo box
   ComboBiennium.Visible = False

'Hide next steps button
   Command2.Visible = False

'Show refresh button
   btn_Refresh.Visible = True

'Clear label caption
   lbl_Progress_Bar.Caption = ""

'show progress bar
   boxProgressBar.Visible = True
   rectProgressBar.Visible = True

'initialize progress bar

'Dim maxProgressBar As Integer
   maxProgressBar = Me.boxProgressBar.Width 'set default progress at 100% = full length of background box

   Me.rectProgressBar.Width = ((0 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "0% complete"
   Me.Repaint


'Deleting Import Error Files that already existed
'MsgBox ("Deleting existing Import Error files")
   lbl_Import_Source_Files.Caption = ""
   lbl_Import_Source_Files.Caption = "Deleting existing Import Error files ..."

'Delete the error tables that exist in the database
   If IsObject(CurrentDb.TableDefs("DOWN45_ImportErrors")) Then
      lbl_List_Import_Filename1.Caption = "deleting >>> DOWN45_ImportErrors"
      DoCmd.DeleteObject acTable, "DOWN45_ImportErrors"
   End If

   If IsObject(CurrentDb.TableDefs("EQMSTR_ImportErrors")) Then
      lbl_List_Import_Filename1.Caption = "deleting >>> EQMSTR_ImportErrors"
      DoCmd.DeleteObject acTable, "EQMSTR_ImportErrors"
   End If

   If IsObject(CurrentDb.TableDefs("Excel_KRL_E21666_ImportErrors")) Then
```

```
        lbl_List_Import_Filename1.Caption = "deleting >>> Excel_KRL_E21666_ImportErrors"
        DoCmd.DeleteObject acTable, "Excel_KRL_E21666_ImportErrors"
    End If

    If IsObject(CurrentDb.TableDefs("Excel_KRL_E21667_ImportErrors")) Then
        lbl_List_Import_Filename1.Caption = "deleting >>> Excel_KRL_E21667_ImportErrors"
        DoCmd.DeleteObject acTable, "Excel_KRL_E21667_ImportErrors"
    End If


'Initiate progress bar
    Me.rectProgressBar.Width = ((1 / 100) * maxProgressBar)
    lbl_Progress_Bar.Caption = "1% complete"
    Me.Repaint


'Delete existing source data tables
'MsgBox ("Deleting existing source tables from database")
    lbl_Import_Source_Files.Caption = ""
    lbl_Import_Source_Files.Caption = "Deleting existing source data files ..."

    lbl_List_Import_Filename1.Caption = "deleting >>> DOWN45_ImportErrors"
    DoCmd.DeleteObject acTable, "tblImportExcel_KRL_E21666-Prev12MoEquipCosts"


    lbl_List_Import_Filename1.Caption = "deleting >>> DOWN45_ImportErrors"
    DoCmd.DeleteObject acTable, "tblImportExcel_KRL_E21667-LTDEquipCosts"


    lbl_List_Import_Filename1.Caption = "deleting >>> DOWN45_ImportErrors"
    DoCmd.DeleteObject acTable, "tblImportHPEMS37R-TEAMS ELINE"


    lbl_List_Import_Filename1.Caption = "deleting >>> tblImportCEClass"
    DoCmd.DeleteObject acTable, "tblImportCEClass"


    lbl_List_Import_Filename1.Caption = "deleting >>> tblImportEQMSTR"
    DoCmd.DeleteObject acTable, "tblImportEQMSTR"


    lbl_List_Import_Filename1.Caption = "deleting >>> tblImportTEAMSActHrs05-07"
    DoCmd.DeleteObject acTable, "tblImportTEAMSActHrs05-07"


    lbl_List_Import_Filename1.Caption = "deleting >>> tblImportTEAMSActHrs03-05"
    DoCmd.DeleteObject acTable, "tblImportTEAMSActHrs03-05"


    lbl_List_Import_Filename1.Caption = "deleting >>> tblImportDOWN45"
    DoCmd.DeleteObject acTable, "tblImportDOWN45"


    lbl_List_Import_Filename1.Caption = "deleting >>> tblImportEMSEquipUseMtr"
    DoCmd.DeleteObject acTable, "tblImportEMSEquipUseMtr"
```

```
'Initiate progress bar
   Me.rectProgressBar.Width = ((5 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "5% complete"
   Me.Repaint


'Deleting NORMALIZED tables that already existed
'MsgBox ("Deleting New tables created during normalization")
   lbl_Import_Source_Files.Caption = ""
   lbl_Import_Source_Files.Caption = "Deleting existing normalized tables ..."


   lbl_List_Import_Filename1.Caption = "deleting >>> EQUIP_CLASS"
   DoCmd.DeleteObject acTable, "EQUIP_CLASS"


   lbl_List_Import_Filename1.Caption = "deleting >>> EQUIP_MASTER"
   DoCmd.DeleteObject acTable, "EQUIP_MASTER"


   lbl_List_Import_Filename1.Caption = "deleting >>> EQUIP_CREW_ASSN"
   DoCmd.DeleteObject acTable, "EQUIP_CREW_ASSN"


   lbl_List_Import_Filename1.Caption = "deleting >>> CREW"
   DoCmd.DeleteObject acTable, "CREW"


   lbl_List_Import_Filename1.Caption = "deleting >>> PRIOR_24MONTH_EQUIP_USE_MTR"
   DoCmd.DeleteObject acTable, "PRIOR_24MONTH_EQUIP_USE_MTR"


   lbl_List_Import_Filename1.Caption = "deleting >>> TEAMS_ACT_HRS_03-05"
   DoCmd.DeleteObject acTable, "TEAMS_ACT_HRS_03-05"


   lbl_List_Import_Filename1.Caption = "deleting >>> TEAMS_ACT_HRS_05-07"
   DoCmd.DeleteObject acTable, "TEAMS_ACT_HRS_05-07"


   lbl_List_Import_Filename1.Caption = "deleting >>> PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS"
   DoCmd.DeleteObject acTable, "PRIOR_12MONTH_EQUIP_REPAIR_OPER_COSTS"


   lbl_List_Import_Filename1.Caption = "deleting >>> LTD_EQUIP_REPAIR_OPER_COSTS"
   DoCmd.DeleteObject acTable, "LTD_EQUIP_REPAIR_OPER_COSTS"


   lbl_List_Import_Filename1.Caption = "deleting >>> CREW_APPROPRIATION"
   DoCmd.DeleteObject acTable, "CREW_APPROPRIATION"


   lbl_List_Import_Filename1.Caption = "deleting >>> DOWN45"
   DoCmd.DeleteObject acTable, "DOWN45"
   lbl_List_Import_Filename1.Caption = "deleting >>> EQUIP_CONDITION_MODEL_REPORT"
   DoCmd.DeleteObject acTable, "EQUIP_CONDITION_MODEL_REPORT"


   lbl_List_Import_Filename1.Caption = "deleting >>> EQUIP_CONDITION_MODEL_REPORT_RANKED"
   DoCmd.DeleteObject acTable, "EQUIP_CONDITION_MODEL_REPORT_RANKED"


   lbl_List_Import_Filename1.Caption = "deleting >>> %AgeStd"
   DoCmd.DeleteObject acTable, "%AgeStd"


   lbl_List_Import_Filename1.Caption = "deleting >>> %CostStd"
   DoCmd.DeleteObject acTable, "%CostStd"


   lbl_List_Import_Filename1.Caption = "deleting >>> %UseStd"
   DoCmd.DeleteObject acTable, "%UseStd"


   lbl_List_Import_Filename1.Caption = "deleting >>> CONDITION_INDEX"
   DoCmd.DeleteObject acTable, "CONDITION_INDEX"


'Initiate progress bar
   Me.rectProgressBar.Width = ((10 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "10% complete"
   Me.Repaint

'Catch any errors that occur during sorce file import
   On Error GoTo Err_Command0_Click

'Begin importing of the datafiles
'MsgBox ("Importing new tables from:" & filePath)
   lbl_Import_Source_Files.Caption = ""
   lbl_Import_Source_Files.Caption = "Importing new source files ...              [~ 1.5 mins]"
   lbl_New_Source_Files_Path.Caption = "New input files location ->" & filepath


   lbl_List_Import_Filename1.Caption = "importing >>> tblImportExcel_KRL_E21666-Prev12MoEquipCosts"
   DoCmd.TransferText acImportFixed, "Excel_KRL_E21666 Import Specification",
"tblImportExcel_KRL_E21666-Prev12MoEquipCosts", filepath & "Excel_KRL_E21666.csv"
   lbl_List_Import_Filename1.ForeColor = 16777088
   lbl_List_Import_Filename1.Caption = "imported >>> tblImportExcel_KRL_E21666-Prev12MoEquipCosts"
   Me.Repaint

'Initiate progress bar
   Me.rectProgressBar.Width = ((20 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "20% complete"
   Me.Repaint

   lbl_List_Import_Filename2.Caption = "importing >>> tblImportExcel_KRL_E21667-LTDEquipCosts"
   DoCmd.TransferText acImportFixed, "Excel_KRL_E21667 Import Specification",
"tblImportExcel_KRL_E21667-LTDEquipCosts", filepath & "Excel_KRL_E21667.csv"
   lbl_List_Import_Filename2.ForeColor = 16777088
   lbl_List_Import_Filename2.Caption = "imported >>> tblImportExcel_KRL_E21667-LTDEquipCosts"
   Me.Repaint

'Initiate progress bar
   Me.rectProgressBar.Width = ((30 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "30% complete"
   Me.Repaint

   lbl_List_Import_Filename3.Caption = "importing >>> tblImportHPEMS37R-TEAMS ELINE"
   DoCmd.TransferText acImportFixed, "HPEMS37R Import Specification", "tblImportHPEMS37R-TEAMS
ELINE", filepath & "HPEMS37R.txt"
   lbl_List_Import_Filename3.ForeColor = 16777088
   lbl_List_Import_Filename3.Caption = "imported >>> tblImportHPEMS37R-TEAMS ELINE"
   Me.Repaint

   'Initiate progress bar
   Me.rectProgressBar.Width = ((40 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "40% complete"
   Me.Repaint

   lbl_List_Import_Filename4.Caption = "importing >>> tblImportCEClass"
   DoCmd.TransferText acImportFixed, "Class Import Specification", "tblImportCEClass", filepath &
"CECLASS.TXT"
   lbl_List_Import_Filename4.ForeColor = 16777088
   lbl_List_Import_Filename4.Caption = "imported >>> tblImportCEClass"
   Me.Repaint

'Initiate progress bar
   Me.rectProgressBar.Width = ((50 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "50% complete"
   Me.Repaint

   lbl_List_Import_Filename5.Caption = "importing >>> tblImportEMSEquipUseMtr"
   DoCmd.TransferText acImportFixed, "HPEMS62R (MtrUse) Import Specification",
"tblImportEMSEquipUseMtr", filepath & "HPEMS62R.TXT"
   lbl_List_Import_Filename5.ForeColor = 16777088
   lbl_List_Import_Filename5.Caption = "imported >>> tblImportEMSEquipUseMtr"
   Me.Repaint

   'Initiate progress bar
   Me.rectProgressBar.Width = ((60 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "60% complete"
   Me.Repaint

   lbl_List_Import_Filename6.Caption = "importing >>> tblImportTEAMSActHrs05-07"
   DoCmd.TransferText acImportDelim, "UsageactivityHrs-conditionmodel Import Specification",
"tblImportTEAMSActHrs05-07", filepath & "Usageactivityhrs-conditionmodel05-07-update.txt"
   lbl_List_Import_Filename6.ForeColor = 16777088
   lbl_List_Import_Filename6.Caption = "imported >>> tblImportTEAMSActHrs05-07"
   Me.Repaint

   'Initiate progress bar
   Me.rectProgressBar.Width = ((70 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "70% complete"
   Me.Repaint


   lbl_List_Import_Filename7.Caption = "importing >>> tblImportTEAMSActHrs03-05"
   DoCmd.TransferText acImportDelim, "UsageactivityHrs-conditionmodel Import Specification",
"tblImportTEAMSActHrs03-05", filepath & "Usageactivityhrs-conditionmodel03-05update.txt"
   lbl_List_Import_Filename7.ForeColor = 16777088
   lbl_List_Import_Filename7.Caption = "imported >>> tblImportTEAMSActHrs03-05"
   Me.Repaint

'Initiate progress bar
   Me.rectProgressBar.Width = ((80 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "80% complete"
   Me.Repaint


   lbl_List_Import_Filename8.Caption = "importing >>> tblImportDOWN45"
   DoCmd.TransferText acImportFixed, "DOWN45 (OASQ) Import Specification", "tblImportDOWN45", filepath
& "DOWN45.TXT"
   lbl_List_Import_Filename8.ForeColor = 16777088
   lbl_List_Import_Filename8.Caption = "imported >>> tblImportDOWN45"
   Me.Repaint

'Initiate progress bar
   Me.rectProgressBar.Width = ((90 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "90% complete"
   Me.Repaint


   lbl_List_Import_Filename9.Caption = "importing >>> tblImportEQMSTR"
   DoCmd.TransferText acImportFixed, "EQMSTR Import Specification", "tblImportEQMSTR", filepath &
"EQMSTR.TXT"
   lbl_List_Import_Filename9.ForeColor = 16777088
   lbl_List_Import_Filename9.Caption = "imported >>> tblImportEQMSTR"
   Me.Repaint

   lbl_Import_Source_Files.Caption = ""
   lbl_Import_Source_Files.Caption = "Completed file import!!!"
'lbl_List_Import_Filename1.Caption = ""


'Show control buttons for next step
   Command2.Visible = True
   btn_Refresh.Visible = True

'Hide labels until prior process is complete
   lbl_Import_Source_Files.Visible = True
   lbl_Process_Data.Visible = True


'Initiate progress bar
   Me.rectProgressBar.Width = ((100 / 100) * maxProgressBar)
   lbl_Progress_Bar.Caption = "100% complete"
```

```
    Me.Repaint


'Beep when completed
    Beep


Exit_Command0_Click:
    Exit Sub

Err_Command0_Click:
    MsgBox Err.Description
    Resume Exit_Command0_Click

End Sub
Private Sub Command2_Click()
On Error GoTo Err_Command2_Click

'Show control buttons
Command0.Visible = False
Command18.Visible = False
btn_Refresh.Visible = True
lbl_run_model_details.Visible = True


'set label caption
lbl_Process_Data.Caption = ""
lbl_Process_Data.Caption = "Running condition model ...           [~ 4.5 mins]"
lbl_run_model_details.Caption = "Creating database from input files"

'Initialize progress bar
Me.rectProgressBar.Width = 0
lbl_Progress_Bar.Caption = "0% complete"

'position progress bar to align with second button (process data button)
Me.rectProgressBar.Top = Me.Command2.Top
Me.boxProgressBar.Top = Me.Command2.Top
Me.lbl_Progress_Bar.Top = Me.Command2.Top


'Initiate progress bar
Me.rectProgressBar.Width = ((25 / 100) * maxProgressBar)
lbl_Progress_Bar.Caption = "25% complete"
Me.Repaint


'Frist suppress MS Access create table warnings
DoCmd.SetWarnings False


'Run the macro that creates the new tables
DoCmd.RunMacro "createNewTables"


'Initiate progress bar
```

```
Me.rectProgressBar.Width = ((50 / 100) * maxProgressBar)
lbl_Progress_Bar.Caption = "50% complete"
Me.Repaint


'run macro to add primary keys to the new normalized tables
DoCmd.RunMacro "addPrimaryKeys"
'lbl_Process_Data.Caption = ""


'Initiate progress bar
Me.rectProgressBar.Width = ((70 / 100) * maxProgressBar)
lbl_Progress_Bar.Caption = "70% complete"
Me.Repaint


'MsgBox ("Primary keys added to new tables created")
'lbl_Process_Data.Caption = "Completed processing new database!!!"
lbl_run_model_details.Caption = "Completed processing new database"


'change label caption when process is completed
'lbl_Process_Data.Caption = "Creating equipment condition report...    [~ 1 min]"
lbl_run_model_details.Caption = "Creating equipment condition report"


'Initiate progress bar
Me.rectProgressBar.Width = ((80 / 100) * maxProgressBar)
lbl_Progress_Bar.Caption = "80% complete"
Me.Repaint


'run macros to compute %Age, %Cost and %Use, Condition and Rank of Equipment
DoCmd.RunMacro "createConditionModelReport"


'Initiate progress bar
Me.rectProgressBar.Width = ((100 / 100) * maxProgressBar)
lbl_Progress_Bar.Caption = "100% complete"
Me.Repaint


'change label caption when process is completed
lbl_Process_Data.Caption = "Completed running equipment condition model!!!"


'hide lable lbl_run_model_details
lbl_run_model_details.Visible = False


'show view report button and label
Command18.Visible = True
lbl_view_condition_report.Visible = True
```

```
'Beep when completed
Beep


Exit_Command2_Click:
    Exit Sub

Err_Command2_Click:
    MsgBox Err.Description
    Resume Exit_Command2_Click

End Sub
Private Sub btn_Refresh_Click()
    DoCmd.Close acForm, "frmImportTables"
    DoCmd.OpenForm "frmImportTables"

End Sub
```

VB CODE FOR REPORTS FORM (FORM_FRMREPORTS)

```
Option Compare Database
Public filepath As String
Public saveFilepath As String

Private Sub btn_Exit_Reports_Click()

    ConfirmExitMsg = "Are you sure you want to close and exit the application?"

    exitApp = MsgBox(ConfirmExitMsg, vbExclamation + vbYesNo, "Confirm Exit")
    If exitApp = vbYes Then

        'Exit application
        DoCmd.Quit

    End If

End Sub

Public Sub Form_Load()
    'define file path -> by default, this will be the same location where the the application is located
    'filepath = GetCurrentDBPath & "Reports\FinalReports\"
    filepath = GetCurrentDBPath

    'Message informing user of default save location andhow to change the destination path
    msgChangeFilePath = "IMPORTANT: The label below indicates the DEFAULT location where files will" _
            & " be saved. To select a different location, delete the default path indicated in the" _
            & " label, then copy and paste (or type) the desired destination directory path in the label."

    lbl_FilePathMessage.Caption = msgChangeFilePath

    'set default save path same as the location where the application resides.
    txt_SaveFileLocation.Value = filepath
    saveFilepath = filepath

End Sub

Public Sub txt_SaveFileLocation_LostFocus()
    'set default save path same as the location where the application resides.
        saveFilepath = txt_SaveFileLocation.Value & "\"

End Sub

Private Sub btn_ResetReportsForm_Click()
    'set default save path same as the location where the application resides.
    'txt_SaveFileLocation.Value = filepath
       DoCmd.Close acForm, "frmReports"
    DoCmd.OpenForm ("frmReports")
End Sub

Private Sub btn_view_rpt_1_Click()
```

```
    'Open equipment condition report in readOnly mode
    DoCmd.OpenTable "EQUIP_CONDITION_MODEL_REPORT_FINAL", acViewNormal, acReadOnly
End Sub

Private Sub btn_save_rpt_1_Click()

    'Save as spreadsheet
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9,
"EQUIP_CONDITION_MODEL_REPORT_FINAL", saveFilepath &
"EQUIP_CONDITION_MODEL_REPORT_FINAL.xls"

    'save messagebox
    MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_2_Click()
    'Open repair history report (query)
    DoCmd.OpenQuery "qry_create_repair_history_report", acViewNormal, acReadOnly
End Sub

Private Sub btn_save_rpt_2_Click()

    'Save as spreadsheet
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_repair_history_report",
saveFilepath & "REPAIR_HISTORY_REPORT.xls"

    'save messagebox
    MsgBox "File Saved"

End Sub

Private Sub btn_view_rpt_3_Click()
    'Open repair history report (query)
    DoCmd.OpenQuery "qry_create_cost_history_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_3_Click()

    'Save as spreadsheet
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_cost_history_report",
saveFilepath & "COST_HISTORY_REPORT.xls"

    'save messagebox
    MsgBox "File Saved"

End Sub

Private Sub btn_view_rpt_4a_Click()
    'Open Light Fleet report (query)
    DoCmd.OpenQuery "qry_create_lightfleet_report", acViewNormal, acReadOnly
End Sub
```

```
Private Sub btn_save_rpt_4a_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_lightfleet_report", saveFilepath
& "LIGHT_FLEET_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_4b_Click()
   'Open Heavy Equipment report (query)
   DoCmd.OpenQuery "qry_create_heavy_equip_report", acViewNormal, acReadOnly
End Sub

Private Sub btn_save_rpt_4b_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_heavy_equip_report",
saveFilepath & "HEAVY_EQUIPMENT_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub

Private Sub btn_view_rpt_4c_Click()
   'Open Attachments report (query)
   DoCmd.OpenQuery "qry_create_attachments_report", acViewNormal, acReadOnly
End Sub

Private Sub btn_save_rpt_4c_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_attachments_report",
saveFilepath & "ATTACHMENTS_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub

Private Sub btn_view_rpt_5a_Click()
   'Open Attachments report (query)
   DoCmd.OpenQuery "qry_create_crews_report", acViewNormal, acReadOnly
End Sub

Private Sub btn_save_rpt_5a_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_crews_report", saveFilepath &
"CREWS_REPORT.xls"
```

```
    'save messagebox
    MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_6a_Click()
    'Open Appropriation 010 report (query)
    DoCmd.OpenQuery "qry_create_appropriation_010_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6a_Click()

    'Save as spreadsheet
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_010_report",
saveFilepath & "APPROPRIATION_010_CONDITION_REPORT.xls"

    'save messagebox
    MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_6b_Click()
    'Open Appropriation 050 report (query)
    DoCmd.OpenQuery "qry_create_appropriation_050_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6b_Click()

    'Save as spreadsheet
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_050_report",
saveFilepath & "APPROPRIATION_050_CONDITION_REPORT.xls"

    'save messagebox
    MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_6c_Click()
    'Open Appropriation 80 report (query)
    DoCmd.OpenQuery "qry_create_appropriation_080_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6c_Click()

    'Save as spreadsheet
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_080_report",
saveFilepath & "APPROPRIATION_080_CONDITION_REPORT.xls"

    'save messagebox
```

```
   MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_6d_Click()
   'Open Appropriation 110 report (query)
   DoCmd.OpenQuery "qry_create_appropriation_110_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6d_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_110_report",
saveFilepath & "APPROPRIATION_110_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_6e_Click()
   'Open Appropriation 120 report (query)
   DoCmd.OpenQuery "qry_create_appropriation_120_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6e_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_120_report",
saveFilepath & "APPROPRIATION_120_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_6f_Click()
   'Open Appropriation 130 report (query)
   DoCmd.OpenQuery "qry_create_appropriation_130_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6f_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_130_report",
saveFilepath & "APPROPRIATION_130_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"
```

```
End Sub


Private Sub btn_view_rpt_6g_Click()
   'Open Appropriation 170 report (query)
   DoCmd.OpenQuery "qry_create_appropriation_170_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6g_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_170_report",
saveFilepath & "APPROPRIATION_170_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub


Private Sub btn_view_rpt_6h_Click()
   'Open Appropriation 200 report (query)
   DoCmd.OpenQuery "qry_create_appropriation_200_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6h_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_200_report",
saveFilepath & "APPROPRIATION_200_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub

Private Sub btn_view_rpt_6i_Click()
   'Open Appropriation 200 report (query)
   DoCmd.OpenQuery "qry_create_appropriation_160_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_6i_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_160_report",
saveFilepath & "APPROPRIATION_160_CONDITION_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub
```

```
Private Sub btn_view_rpt_7a_Click()
   'Open Appropriation 200 report (query)
   DoCmd.OpenQuery "qry_create_metered_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_7a_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_metered_report", saveFilepath &
"METERED_EQUIPMENT_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub

Private Sub btn_view_rpt_7b_Click()
   'Open Appropriation 200 report (query)
   DoCmd.OpenQuery "qry_create_non-metered_report", acViewNormal, acReadOnly
End Sub


Private Sub btn_save_rpt_7b_Click()

   'Save as spreadsheet
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_non-metered_report",
saveFilepath & "NON-METERED_EQUIPMENT_REPORT.xls"

   'save messagebox
   MsgBox "File Saved"

End Sub


Private Sub btn_Save_All_Reports_Click()
   MsgBox "Are you sure you want to save all the reports?", vbQuestion + vbYesNo


   'Save as spreadsheets

   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9,
"EQUIP_CONDITION_MODEL_REPORT_FINAL", saveFilepath &
"EQUIP_CONDITION_MODEL_REPORT_FINAL.xls"
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_repair_history_report",
saveFilepath & "REPAIR_HISTORY_REPORT.xls"
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_cost_history_report",
saveFilepath & "COST_HISTORY_REPORT.xls"
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_lightfleet_report", saveFilepath
& "LIGHT_FLEET_CONDITION_REPORT.xls"
   DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_heavy_equip_report",
saveFilepath & "HEAVY_EQUIPMENT_CONDITION_REPORT.xls"
```

```
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_attachments_report",
saveFilepath & "ATTACHMENTS_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_metered_report", saveFilepath &
"METERED_EQUIPMENT_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_non-metered_report",
saveFilepath & "NON-METERED_EQUIPMENT_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_010_report",
saveFilepath & "APPROPRIATION_010_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_050_report",
saveFilepath & "APPROPRIATION_050_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_080_report",
saveFilepath & "APPROPRIATION_080_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_110_report",
saveFilepath & "APPROPRIATION_110_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_120_report",
saveFilepath & "APPROPRIATION_120_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_130_report",
saveFilepath & "APPROPRIATION_130_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_130_report",
saveFilepath & "APPROPRIATION_160_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_170_report",
saveFilepath & "APPROPRIATION_170_CONDITION_REPORT.xls"
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel9, "qry_create_appropriation_200_report",
saveFilepath & "APPROPRIATION_200_CONDITION_REPORT.xls"



    'save messagebox
    MsgBox "Files Saved"


End Sub
```

VB CODE FOR GRAPHICAL USER INTERFACE

```
Private Sub ComboBiennium_Change()


    'Display a new label with the selected bienium, align it to the combo box position
    lbl_SelectedBiennium.Caption = ComboBiennium.Value
    lbl_SelectedBiennium.Visible = True

    'Create a variable for the selected biennium
    currBiennium = ComboBiennium.Value

    ConfirmBienniumMsg = "Please confirm that ALL the input files are from the selected " & currBiennium & "
biennium?"

    answer = MsgBox(ConfirmBienniumMsg, vbExclamation + vbYesNo, "Confirm biennium")
    If answer = vbYes Then
        'show import file button and label
        lbl_Import_Source_Files.Visible = True
        Command0.Visible = True

        'Supress MS Access warnings
        DoCmd.SetWarnings False

        'set current biennium values in table BIENNIUM to "Null"
        sqlResetBiennium = "UPDATE BIENNIUM SET CurrentBiennium = NULL, SQLQryBiennium = NULL"

        'MsgBox (sqlResetBiennium)
        DoCmd.RunSQL (sqlResetBiennium)

        'identify current biennium in DB by updating CurrentBiennium & SQLQryBiennium fields in BIENNIUM
table
        'NB: chr(34) represents the double quote character within a string, chr(39) single quotes

        sqlUpdateBiennium = "UPDATE BIENNIUM SET CurrentBiennium = 'Yes', " _
        & "SQLQryBiennium = " & Chr(39) & "*" & currBiennium & "*" & Chr(39) & " WHERE Biennium like " &
"'" & currBiennium & "'"

        'MsgBox (sqlUpdateBiennium)
        DoCmd.RunSQL (sqlUpdateBiennium)

    End
    Else
        'DoCmd.Close acForm, "frmImportTables"
        DoCmd.OpenForm "frmImportTables"
    End If

    End Sub
```