

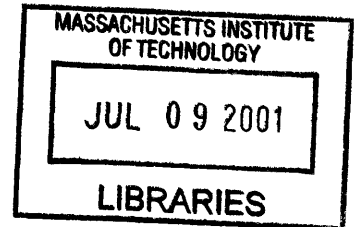
Optimizing Engineering Analysis Resource Allocation

By
Quang Nguyen

BARKER

B.S. Mechanical Engineering, University of Washington, 1995

Submitted to the Sloan School of Management
and the Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degrees of

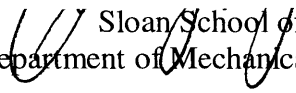


MASTERS OF BUSINESS ADMINISTRATION and MASTER OF SCIENCE IN MECHANICAL ENGINEERING


In conjunction with the Leaders for Manufacturing Program at the
Massachusetts Institute of Technology
June 2001

© 2001 Massachusetts Institute of Technology. All rights reserved.

Signature of Author: _____

 Sloan School of Management
Department of Mechanical Engineering
May 2, 2001

Certified by: _____

 Professor Steven D. Eppinger, Thesis Advisor
Professor of Management

Certified by: _____

 Professor James M. Utterback, Thesis Advisor
Professor of Management

Accepted by: _____

Professor Ain A. Sonin, Chairman, Graduate Committee
Department of Mechanical Engineering

Accepted by: _____

Margaret C. Andrews, Executive Director of Master's Program
MIT Sloan School of Management

This page intentionally left blank.

Optimizing Engineering Analysis Resource Allocation

By
Quang Nguyen

Submitted to the Department of Mechanical Engineering
and to the Sloan School of Management on May 2, 2001
in Partial Fulfillment of the Requirements for the Degrees of

Master of Business Administration
and
Master of Science in Mechanical Engineering

ABSTRACT

In recent years, automotive companies have become increasingly competitive. With margins already thin, automotive manufactures look to streamlining operating and development cost. One particular area of interest is the increased usage of math analysis tools to reduce development time and resources.

By using math analysis tools to predict product design behavior and validate requirements, the need for costly testing hardware is reduced. This thesis addresses the question of where to focus math analysis development efforts in order to optimize hardware reduction, through the development of a decision-support optimization model. In addition, the thesis addresses the cultural and organizational barriers associated with implementing new engineering technology.

Providing engineers with the best math analysis tools does not ensure that they will use or trust it. Successful implementation of a new technology requires careful planning from both the technical and societal perspectives.

Thesis Advisors:

Professor James M. Utterback, Sloan School of Management
Professor Steven D. Eppinger, Sloan School of Management

ACKNOWLEDGMENTS

This thesis is dedicated my mother who was recently afflicted with cancer. My mom has always placed her children first and has been there for me through the many years of growing up. From fleeing a war-torn Vietnam with little in our pockets, to dressing, feeding and getting me ready for pre-school, to calling me at graduate school to see how I was doing, I would not be where I am today if not for her love and encouragement.

I would also like to acknowledge my family, my co-workers, my advisors, and my friends in the LFM program. My family has always lent me their support and devotion to my schooling. My GM co-workers, Dave Kim, Susan Owen, Bernice Jung, Julie Irwin, Alex Lin, Dan Wingerdan, CC Hsu, Eileen Mutiso, Cara Melch, and Laura Wing for readily adopting me into the group. My academic advisors, Steven Eppinger, James Utterback, and Steve Graves, who have taken interest in my project and have provided valuable insight as well as expertise. Finally, I would like to acknowledge my friends in the LFM program for their friendship and support.

I am very thankful to all those who assisted me and supported my efforts in this project.

The author also wishes to note that the host company edited the contents of this thesis for proprietary information before releasing it for final publication.

Table of Contents

1.0 Introduction	8
1.1 Background	8
1.2 Project Definition	9
1.3 Organization	9
2.0 Project Background	10
2.1 Vehicle Product Development Process	10
2.1.1 Architecture Exploration	11
2.1.2 Styling Freeze	11
2.1.3 Start of System Fill	11
2.2 Analysis Development Validation (ADV) Process	11
2.3 Capability Matrix	13
3.0 Analysis of Problem	16
3.1 Customer Requirements	16
3.2 Mapping Requirements to Development Hardware	17
3.3 Relationship between Requirement and Hardware	19
3.4 Challenges to Optimizing Resource Allocation	20
3.4.1 Dynamic Variable Relationship	20
3.4.2 Linking New Math Analysis Software to the Solution	21
3.4.3 Incorporating the Capability Matrix	21
3.4.4 Cultural and Organizational Issues	22
3.5 Two Development Areas of Change	23
4.0 Technology Development	24
4.1 Model Scope	24
4.2 Optimization Modeling Algorithm	24
4.3 Modeling Assumption	26
4.4 Modeling Formulation	27
4.4.1 Inputs/Parameters	27
4.4.2 Decision Variables	29
4.4.3 Objective Function	29
4.4.4 Constraints	30
4.4.5 Model Output	30
4.4.5.1 Requirement and Test Procedure Output Matrix	30
4.4.5.2 Model Objective Output	31
4.5 Analysis Procedure	31
4.6 Validation and Preliminary Results	33
4.7 Data Regression	36
4.8 Optimization Heuristics	36
4.9 Modeling Issues	41
4.10 Extensions to the Model	43

4.10.1 Create Heuristic Optimization Model	43
4.10.2 Incorporate Capability Matrix into Model	43
4.10.3 Linking Software Initiatives to Vehicle Reduction	44
4.11 Technology Development Conclusions	46
5.0 Socio-Technical Development	48
5.1 Organizational and Cultural Development Goal	48
5.2 Identifying Causal Factors	48
5.2.1 Survey Results	50
5.3 Math Analysis Usage System Dynamics Model	54
5.3.1 Correlation Loop	55
5.3.2 Trust Loop	57
5.3.3 Culture Loop	60
5.3.4 Cost Pressures Loop	61
5.3.5 Incentives Loop	63
5.3.6 Systemic Model	65
5.4 Policy Implementation Opportunity	66
5.5 Project Extensions	67
5.5.1 Identify Key Policies to Increase Math Analysis Usage	67
5.5.2 Modify into Decision-making Tool	67
5.5.3 Create System Dynamics Model for Hardware Cost	68
5.6 Socio-technical Development Summary	68
6.0 Conclusions	69
6.1 Create System Dynamics Model of Hardware Cost	69
6.2 Develop Heuristic Optimization Program	69
6.3 Improve Capability Matrix	69
6.4 Increase Efforts to Improve Correlation	70
6.5 Integrate Heuristic Optimization Program with Capability Matrix	70
6.6 Create Goals to Limit Hardware Allocation	70
Bibliography	72
Appendix A: Sample Model	74
Appendix B: Qualitative Surveys	77
Appendix C: Quantitative Surveys	83
Appendix D: Math Analysis Usage System Dynamics Model	87

List of Figures

Figure 1. Vehicle Product Development Process	10
Figure 2. Requirements and the ADV Process.....	12
Figure 3. Mapping Decision Variables to the Objective	18
Figure 4. Two Development Areas of Change	23
Figure 5. Optimization Model Algorithm	25
Figure 6. Modeling Analysis process flow diagram for Program Q.....	32
Figure 7. Solver Model vs. Baseline Scenario.....	35
Figure 8. Heuristics Optimization Algorithm.....	44
Figure 9. Mapping Software Initiatives to Vehicle Reduction	45
Figure 10. Decision-Making Hierarchy	49
Figure 11. Position on Whether Math Analysis Improves Quality	51
Figure 12. Confidence in Math Analysis vs. Years of Experience.....	52
Figure 13. Confidence in Math Analysis vs. Job Function	53
Figure 14. Correlation Loop.....	56
Figure 15. Trust loop.....	59
Figure 16. Culture Loop.....	60
Figure 17. Cost Pressures Loop.....	62
Figure 18. Incentives Loop.....	64

List of Tables

Table 1. Example of Capability Matrix with Hypothetical Data.....	14
Table 2. Example of the Capability Matrix	15
Table 3. Sample of Requirements and Test Procedures.....	19
Table 4. Heuristic Optimization Example	40
Table 5. Heuristic vs. Solver Model	41

1.0 Introduction

1.1 Background

In recent years, automotive companies have become increasingly competitive. With margins already thin, automotive manufactures look to streamlining operating and development cost. One particular area of interest is the hardware stage, which is the stage during the new product development process where designs are prototyped and tested on full prototype vehicles. The prototyping stage of the product development cycle consumes considerable amount of time and both human and economic resources.

One way to reduce time and resources is to use computer analysis tools or math analysis tools, rather than prototyped hardware, to predict design behavior and validate requirements. Examples of computer analysis tools include finite element analysis and stress analysis. By eliminating the need to perform physical tests, the number of required tests are reduced and thus the number of prototype vehicles needed for testing is also reduced. Many manufacturers recognize the fact that math analysis will play a critical role in reducing development cost as well as development time, and have included the growth of math analysis capability in their strategic planning. However the math analysis tools vary in the level of confidence of their capability to predict design behavior. There are many underdeveloped analysis tools which do not allow engineers to eliminate hardware with acceptable risk. This risk prevents the replacement of hardware testing with math analysis.

Another factor preventing the increased use of math analysis in lieu of hardware testing has to do with cultural and organizational barriers. Providing engineers with the best math analysis tools does not ensure that they will use or trust them. The introduction of new technology also introduces change and this is met usually with resistance.

Successful implementation of a new technology requires careful planning from both the technology and socio-technical aspects.

1.2 Project Definition

To further reduce development cost, the General Motor Vehicle Product Development group needs to increase the use of analysis tools. Given that management wants to proceed in this direction the question then becomes:

*Given constrained resources, where can we concentrate
math analysis development for maximized reduction of test vehicles.*

The math analysis group has a limited budget that they are able to spend on developing new analysis tools, as well as limited human resources. Currently their growth strategy is to distribute the development budget over all the different analytical areas. The math analysis tools can be procured either as packaged software, developed by an internal group, or developed through sponsorship of research. Several things need to be considered such as maximum benefit to cost and end quality of the product. The question of benefit is also complicated by the fact that it is difficult to map the math analysis tools to hardware. Because of this, developing analysis capabilities for particular requirements may have no impact on hardware reduction. This thesis addresses the development of a mathematical model or framework that was used to determine where to optimally focus development of analysis tool capability.

The second part of this thesis addresses the cultural aspects of introducing new math analysis tools and procedures to the product development groups. By understanding how different groups interact and the dynamics between different factors we can start to understand how to affect change.

1.3 Organization

This thesis is organized similar to the approach used in analyzing the resource allocation problem. After understanding the background, an in-depth analysis was performed on the process involved and a flow diagram was created to map the objective to the variables. The analysis provided understanding into the problem and spawned two topic areas, technology development and organizational and cultural development. Finally, the findings and results are discussed as well as future extensions to the model.

2.0 Background Information

The first step to understanding the problem is to understand some of the key processes involved. The following briefly summarizes the processes related to the resource optimization problem.

2.1 Vehicle Product Development Process

Most automobile manufacturer follow a similar Vehicle Product Development (VPD) process, with the common goals of reducing product development time and cost. Although the VPD process itself is very complex, for our purposes we will simplify it into three distinct phases. Of particular importance is the concept of learning cycles, where prototype vehicles are constructed to test design concepts for both developmental and validation purposes. These learning cycles occur throughout the VDP whenever there are prototype builds. The three phases are shown in Figure 1.

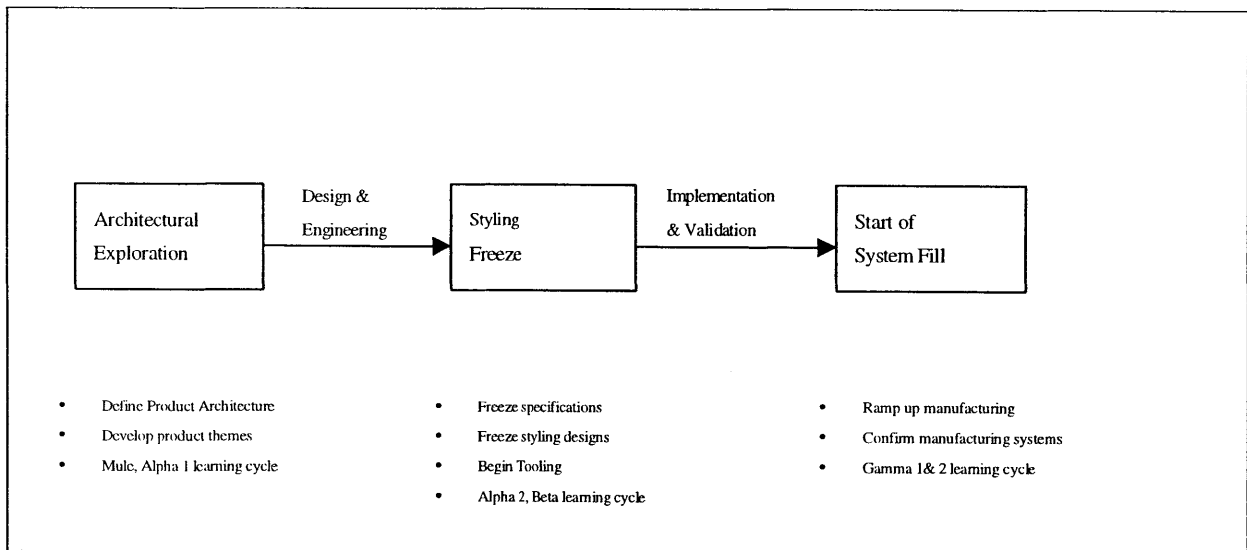


Figure 1. Vehicle Product Development Process

2.1.1 Architecture Exploration

The architectural exploration phase allows for the definition and exploration of the vehicle's underbody and structural components, such as the frame, sheet metal body, and critical vehicle dimensions. During this phase the architecture is approved early on and designs for integration of the vehicle subsystems are initiated. Mule and Alpha 1 builds are also completed during this phase if necessitated. The Mule stage is used to investigate architectural alternatives before architecture selection complete and/or to develop selected architectural subsystems by approved architecture. The Alpha 1 stage is used to confirm underbody structure design.

2.1.2 Styling Freeze

The styling freeze phase signifies that all interior and exterior surfaces are frozen, with the exception that the changes do not impact critical paths. In this phase vehicle designs are completed and validated to make sure they meet requirements. The requirements are completed at all vehicle levels (component, sub-systems, and full vehicle). Alpha 2 and Beta builds are also completed during this phase if needed. Alpha 2 is used to confirm crashworthiness and occupant protection. Beta is used to confirm total vehicle technical specifications.

2.1.3 Start of System Fill

In this phase the product development team shifts to getting manufacturing ready for production. The Gamma builds are used to test out the manufacturing systems.

2.2 Analysis Development Validation (ADV) Process

The first step in understanding the model is understanding the related processes. The ADV process is a sub-process within the VDP process that supports the validation of vehicle, subsystem, and component requirements. Given a set of vehicle requirements, the ADV process identifies the development and validation work that needs to be executed, the order in which the work needs to be done, and the resources required (including time and pre-production hardware). The ADV process is complex and requires careful planning to fully utilize available resources. An ADV plan is produced by several cross-functional sub-teams and is optimized by both a computer program and/or by team members.

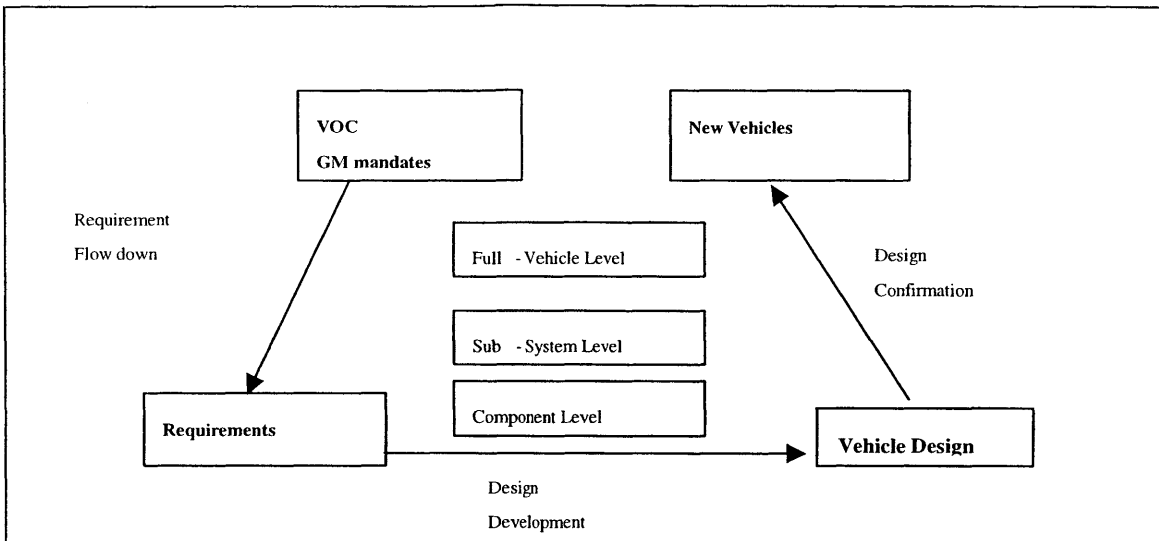


Figure 2. Requirements and the ADV Process.

The concept of requirements and an overview of the ADV process are shown in Figure 2. Requirements are produced from several sources such as the voice of the customer, GM mandates, engineering constraints, design constraints, and regulatory mandates. Once requirements are refined they are used describe the performance standards which component, sub-system, or full vehicle designs must meet. An example of this concept could be the mandate from the Federal Motor Vehicle Safety Standards & Regulation, which stipulates that burn resistance requirements of 4 inches per minute must be met for seat belts. This then becomes one of the specifications or requirements for the seat belt that must be validated during the ADV process.

A Design Release Engineer (DRE) would then design the part to meet the requirement, prototype the part, and then test the part during one of the hardware builds. If the seatbelt design performs to specification, the design is finalized, validated to confirm adherence to specification, and released. This must be done for requirements at all levels of the vehicle (component, sub-system, and full vehicle) prior to going into production.

The ADV process consists of several different activities centered around performing tests to validate requirements. Tests in this context can be defined as math-based analyses,

demonstrations, inspections, or hardware tests. *Development* is the structured process of comparing, modifying, selecting, and optimizing a design through tests. *Validation* is the formal process of confirming that a requirement is met through tests. Finally, *Analysis* is the calculation of the design behavior under specified conditions by representing mathematically (building mathematical models, simulations, algorithms, equations), examining data, and generating design conclusions.

It is the final activity, analysis, with which our interest lies. Although the ADV process encompasses both math-based and hardware-based methods, there is a push by management to migrate towards math-based methods because they hold the potential to reduce both development time and resources.

Although much work has been done in developing math analysis tools, as well as math analysis processes, math analysis is still in its early stages and more work needs to be done to streamline its use. One area of development is to identify the capability of the math analysis tools to satisfy each requirement in the ADV process.

2.3 Capability Matrix

The analysis capability differs for each analysis tool based on confidence in its ability to eliminate hardware. A matrix has been developed by the Global Analysis group to identify the capability levels and is shown in Table 1.

Level	Capability	Hardware Reduction	Hardware Need
1	Inadequate analysis capability	No hardware reduction	Yes
2	Capability to rank design alternatives	Eliminate some hardware and hardware development	Some
3	Capability to do development after one math model validation	No hardware for development	Some
4	Results as good as test data after one math model validation	Only one early hardware test necessary. No development or validation hardware	No
5	Analytical results as good as test data	No hardware necessary	No

Table 1. Example of Capability Matrix with Hypothetical Data

The capability matrix was intended as a decision support tool. ADV sub-teams could reference the capability matrix to determine how confident they can be in using math analysis for specific requirements. The capability level is determined by a focus group familiar with the math analysis tool, its usage, and the requirement it addresses.

The capability matrix is created by a committee of managers, mechanical designers, and analytical designers, under the following assumptions:

- Manpower is not a constraint (when commercial or in-house tools exist, the programs would be installed and licensed)
- Program timing is not a constraint (if we can do it now, but cannot currently speed it up with more people, tool licenses, process enhancements, etc.)
- Organizational boundaries are not considered within the campus site (the rating is of Computer Aided Engineering (CAE) capability at the campus, not just the analysis department's CAE capability at the campus).

- To have a given level of capability, someone at the site must have done the CAE task at least once at that level, or done it for a very similar application.
- Availability of model validation data (from in-house labs and suppliers) will not be a problem.

The committee looks at the capability on a build-by-build basis (each hardware stage), determines if a hardware test is needed to exit build stage, and decide if all build decisions can be made using math. They then identify the build stage during which math decision involves highest risk to set the overall capability. An example of the capability matrix is shown in Table 2.

Requirement ID	Title	# of Req	Level	Focus Group
CR1	Coarse Road Isolation	12	2	Controls Contributes
SR1	Smooth Road Isolation	12	2	Vehicle Dynamics
RF1	Ride Frequency	4	4	Vehicle Dynamics
RF1	Rear/Front Ride Frequency Ratio	2	4	Vehicle Dynamics

Table 2. Example of the Capability Matrix

3.0 Analysis of Problem

3.1 Customer Requirements

Before we even start the analysis we must ask what is the customer requirement? The customer or user of the analysis in this case is the CAE software portfolio shareholder. This includes the Information Technology Group, the Analysis group, and Engineering Operations Management. These three groups have the onus of determining the CAE development in terms of where to spend the money as well as where to focus the human resources. This is important because the way we build the model and perform the analysis must be in a usable format for the decision-maker. Questions about preferences, decision criteria, and answer format need to be discussed. The customer requirements for the problem can be simplify to two main questions:

- “Where do we put our software development dollars to reduce the most prototype hardware?” (development budget allocation)
- “Given that we know what we need to develop, who do we put on what development effort?” (human resource allocation)

Since those questions are two completely different optimization projects, and because there are no existing data or estimates of human resource allocation, only the first question will be focused on in this thesis. The next question is what form should the answer be in, and how detailed should it be? Mapping the math analysis tools to the requirement and then optimizing to see which math analysis tools would reduce the most hardware would be ideal, however it would be very difficult not only to map the tool to the requirement, but also to create a data source for all the potential CAE software would be unfeasible. Grouping the requirements by certain testing characteristics that are common to a given math analysis area was also another consideration, however we would lose some resolution to the optimization model. To keep things simple we will focus on the low hanging fruit: which top requirements, if replaced with math-based testing would eliminate the highest percentage of hardware. The constraints for this approach are discussed in later sections.

3.2 Mapping Requirements to Development Hardware

The migration towards math-based design and testing methods leaves the question of where do we best focus resources to develop our math analysis capability to maximize cost reduction. Math-analysis can reduce cost in several different ways such as time, quality, or utilization, however, this thesis will only focus on a measurable area, hardware testing. To understand how math-analysis can affect hardware we first need to map requirements to hardware.

The next step in understanding the model is mapping the objective (prototype vehicle reduction shown below shaded in blue) to the decision variables (requirements shown below shaded in gray). The objective is to find which set of requirements, if “removed” (i.e., replaced with math analysis) would result in the lowest vehicle count. It is here that we discover several complexities. Not only do the requirements map to hardware through several different paths, but also these paths are also dynamic from vehicle program to vehicle program and across hardware stages within a program. A visual representation of this mapping is shown in Figure 3.

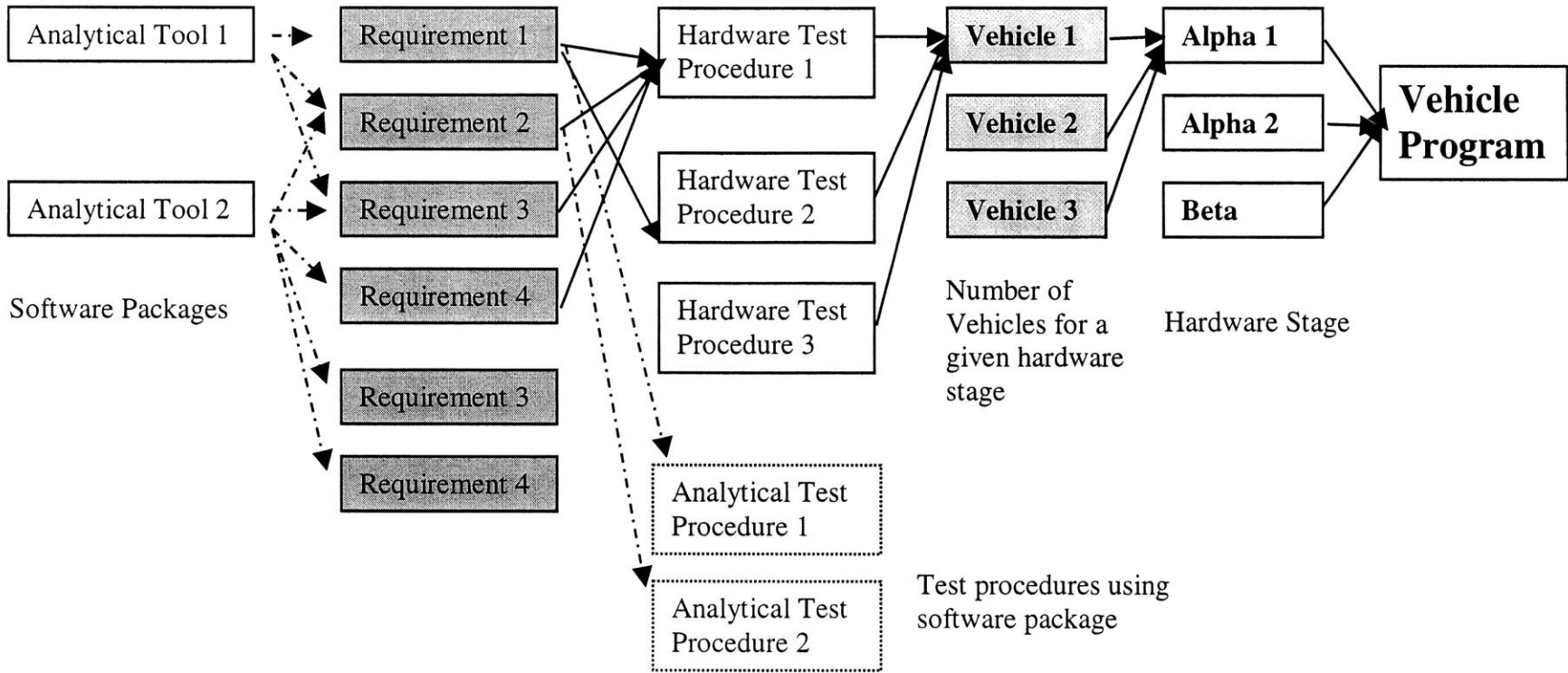


Figure 3. Mapping Decision Variables to the Objective

For a specific vehicle program there may be several different hardware build stages depending on the complexity of the new vehicle design. In Figure 3, there are three hardware stages: Alpha 1, Alpha 2, and Beta. A given hardware stage will be allocated a certain number of vehicles for testing purposes, and each vehicle has a given number of weeks to be used for testing. To make efficient use out of each vehicle, it must be packed with as many test procedures as possible for its given testing life. An optimization program, called Rainier, has been developed to configure the minimum number of vehicles needed for each hardware stage, given the test durations, requirements, and procedures involved. Each test procedure may be validating several different requirements. A sample of requirements and test procedures is shown below:

Requirement Description	Requirement ID	Test Procedure ID
Maximum Single Occupant Weight	R1	TP1 /TP4
Bumpers & Fascias	R2	TP2
Steering Control Effort	R3	TP3
Seat Integrity/Seat Chuck Performance	R4	TP4
Static Steering Effort	R5	TP3

Table 3. Sample of Requirements and Test Procedures

A requirement can be assigned to more than one test procedure and several different requirements can also be assigned to one test procedure. For example, Maximum Single Occupant Weight could be tested on TP1 as well as TP4. Likewise several requirements (Steering Control Effort and Static Steering Effort) can be assigned to a single test procedure, TP3. The goal is to move the testing of requirements from hardware test procedures to analytical test procedures.

3.3 Relationship between Requirement and Hardware

One of the key lesson from the mapping exercise is that there is no clear relationship between requirement to hardware. Not only does the relationship change from program to program, but also within the program relationships can change; finding a static relationship could be difficult and creating a dynamic model might be unrealistic. One also has to ask if optimizing data from past programs will be useful in planning for the future since the solution to the optimization

model will be specific to only that particular data set. However, we can make some assumptions that would simplify the model.

The first assumption is that requirements are generally mapped to the same test procedures, and this relationship will be provided by past ADV plans. The second assumption is that the relationship between test procedure and vehicles can be provided by Rainier, since it identifies which test procedures is tested on which vehicles. A representative vehicle development program will be used to provide the ADV plan and the Rainier data that map the relationship of requirement to vehicles. Once the model has been developed and a solution set is determined, we can regress the data to determine if any useful conclusions can be found. By generalizing the solution set we can learn what drives the hardware count.

3.4 Challenges to Optimizing Resource Allocation

After initial analysis and model formulation, several issues with modeling this issue came up, forcing re-evaluation of the model scope.

3.4.1 Dynamic Variable Relationship

One of the key modeling issues is that the relationships between the different modeling variables are dynamic not only from program to program, but also within the vehicle program. We are trying to plan for the future by studying data from the past, but each year the data change, so is the solution from the past relevant to future planning? However this is the only data we have available and to truly make any progress the solution to the resource allocation problem must look at the specific data. Earlier studies have been done with a macro-level perspective or strategic level, but this only addresses the obvious factors. Only by looking at the problem from the micro-level perspective can we uncover new answers. This issue forces us to think about how we build the model, what inputs we put in, what data comes out, and how do we use these data? There has been some effort to create standards or templates for vehicle program, which would help establish a consistent relationship between requirements and test procedure,

but because of the wide variety of automobiles and requirement needs, these templates have not been utilized.

3.4.2 Linking New Math Analysis Software to Solution

Knowing which requirements to replace with math-based testing does not necessarily tell you which new math analysis tools should be developed. There are two areas where math analysis tools can be developed: increasing the current capability of existing tools to a level where it can replace hardware testing and acquiring new math analysis software. You can increase the current capability of existing tools by running correlation experiments, and documenting analysis procedures. Of the two development areas, increasing current capability is the preferable choice because fewer resources are needed than with bringing in new software tools. The question then becomes how do you link the math analysis software development with the requirements, as the requirements are not generally linked to any of the tools.

3.4.3 Incorporating the Capability Matrix

After more in-depth analysis of the capability matrix, there were several issues that made incorporating the capability matrix into the model more difficult. However, there is value to showing the benefits of incremental analysis capability in the model. The issues with the capability matrix are identified below:

- From the interviews with different people in the organization and from research of documentation on the intranet, there seems to be no well-defined vision, definition, or intention of use for the capability matrix. Everyone had his or her own idea of what the capability matrix was, but there was not a central source documenting the essence of the capability matrix, thus creating inconsistency on scope definition.
- Lack of transference throughout corporation. Most interviewees were unfamiliar with the capability matrix and its usage. Although a lot of effort has been expended, due to low awareness, no one really uses the matrix.

- In its current state the capability matrix is a guideline for ADV planners, not a precept. There should be more definitive rules on when a requirement should be tested using hardware or math analysis. The capability matrix needs to have more useful definable/measurable metrics. For example, when a math analysis tool is able to correlate to hardware within a certain percentage on three different occasions, it is given a capability rating of three and can subsequently be used in lieu of hardware testing.
- It was hard to actually find the data source for the requirements. The data were not in a user-friendly format, such as Excel. A user guide should be created for the Capability matrix documenting the history, scope, intent, assumptions, usage, contacts, etc.
- Data on capability levels are incomplete or outdated. Only 20% of the requirements for Q program were found.
- Determination of capability level is subjective.
- Capability level can change from program to program and stage to stages.
- Capability level needed to reduce hardware is different for different stages.

3.4.4 Cultural and Organizational Issues

One of the major issues uncovered during the analysis did not have to do with technical modeling obstacles, but rather the cultural and organizational issues surrounding math analysis usage. *It was discovered that there were very few instances of actually math-based testing, but many instances of cultural or organizational reasons for not using math-based testing, when the capability for math-based testing was available.* Also during the initial interviews, it was increasingly evident that cultural differences existed between the different vehicle product development groups.

The focus of this project was to optimally allocate resources for the purpose of reducing development and validation hardware. This reduction will be achieved through increased math-based testing, however, providing engineers with analytical tools does not always guarantee that they will be used.

3.5 Two Development Areas of Change

Organizations are resistant to not only technology, but also the change that technology introduces, so there are two sides to integrating technology into an organization: the technology and socio-technical development.

This fact led to increase the scope of the project to not only provide an optimization model that answers the question where to focus resource development, but also try to understand how to change the organizational culture to use math-based testing in lieu of hardware testing. The socio-technical development side of the project deals with trying to gain insight into the cultural and organizational barriers that prevent GM from utilizing the full potential of their modeling capability.

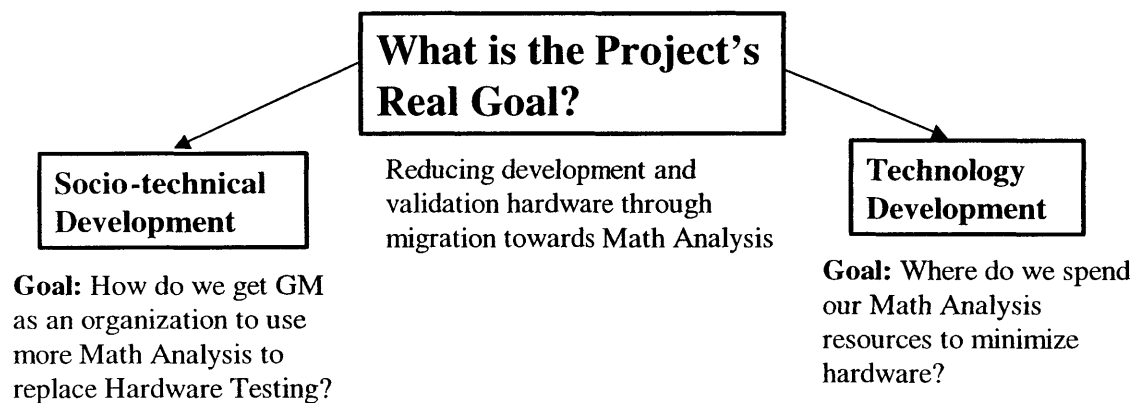


Figure 4. Two Development Areas of Change

The thesis addresses both aspects, starting with the technology development.

4.0 Technology Development

The goal of the model is to develop a decision support tool that can be used to help identify where math analysis resource development should be focused that would have the most impact on reducing development and validation hardware. The reduction of development and validation hardware will be accomplished through increased math-based testing.

4.1 Model Scope

The model will be used to analyze past program data to identify trends that affect hardware counts. This can be accomplished by using the optimization model to determine which combination of requirements that, if tested with math analysis, would result in the greatest vehicle count reduction, given a constraining number of requirements that can be replaced via math.

The model will only answer the question of which requirements, if “eliminated”, would have the greatest impact on vehicle count for a specific program and hardware stage. The model results should then be analyzed to identify any trend in which requirements are affecting vehicle counts. This information can then be used to answer the resource allocation question.

- Solving the problem of where to allocate limited human resources for math analysis is not within scope of the project.
- Solving the problem of where to allocate limited current project resources for a vehicle program are not within the scope, but will be discussed in Section 12.2.

4.2 Optimization Modeling Algorithm

Once the relationship between decision variables and the objective has been identified, as discussed in Section 3.3, the optimization modeling algorithm can be determined. As shown in Figure 5, the decision variables in the model are vehicle program requirements. These variables are binary to describe whether or not the associated requirements will be tested using hardware

or math-based methods. The requirements in question can include full vehicle, sub-system, and component-level requirements, since all are tested on full vehicles.

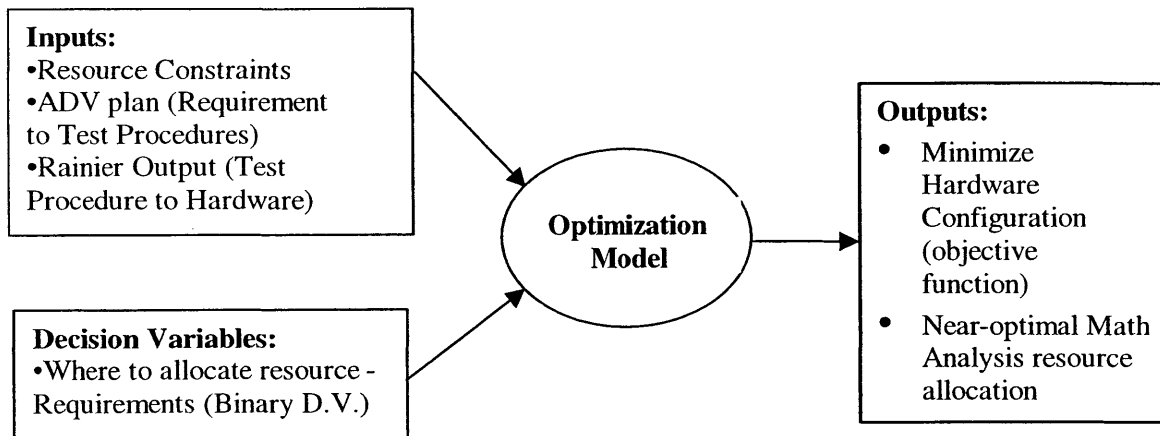


Figure 5. Optimization Model Algorithm

The model has 3 main inputs:

1. ADV plan that provides the relationship between requirements and test procedures.
2. Rainier output that provides the relationship between test procedures and vehicles.
3. Analysis group resource constraints, given as the number of requirements that can be replaced.

Rainier output is used since the model will not interface with Rainier to determine vehicle configurations. This means that the model will be determining math analysis resource allocations from an optimized vehicle configuration.

Using Excel Solver Premium, the model will maximize the number of vehicles reduced given a constrained resource. The output from the model will be the number of vehicles reduced and which combination of requirements are validated using analysis to achieve that reduction.

4.3 Modeling Assumptions

This model was created with the assumptions listed below:

- Requirements “removed” from an ADV plan are assumed to be tested using math analysis and no additional hardware testing is needed.
- The requirements “removed” will have the greatest impact on vehicle count for only that specific program and hardware stage. Removing these requirements on other programs may not have the same impact on hardware. However, there is some assumption of continuity, or that requirements have a tendency to be matched to the same test procedures and thus mapped to the same vehicles.
- A unique model needs to be generated for each hardware stage in a vehicle program. Although the model follows the same format, the data set is different and thus the model form factor will be different.
- The model solution may not be unique. There could be other solutions resulting in the same amount of vehicle reduction.
- Because of the exponential number of possible answers, Premium Solver Plus uses several different solving methods (Simplex Linear, GRG Nonlinear, Mixed-Integer, Evolutionary), which does not guarantee optimal solutions.
- In order for the requirements to be “turned off” it means that the math analysis capability is at a minimum capability level for a given hardware stage, i.e. For Alpha 1 the math analysis capability needs to be 3 or above to be able to eliminate hardware. The required capability level differs for each hardware stage.
- Requirements are only mapped to specific test procedures, via past ADV plans. (See section 4.0.)

- Rainier output provides the mapping from test procedures to specific vehicles. The model does not re-optimize the hardware configuration after determining which requirements to remove. To do so would require integration with Rainier (see section 4.0).
- The analysis capability matrix is not taken into account in the current model, so the model does not account for incremental benefits of math analysis development. For example, it may be less expensive to move a level 2-analysis tool to a level 3 than it is to bring in a brand new level 3-analysis tool. The implication of not taking into account the analysis capability is that the model ignores cost and feasibility, whereas cost and feasibility could be a major decision-making factor. The importance of the capability matrix is acknowledged, and will address its future integration into the model (see section 12.2).

4.4 Modeling Formulation

With the objective and variables identified, we can start formulating the model. The model was created using Excel because it allows visual understanding of the model and most users are familiar with it. The following sections describe the model.

4.4.1 Inputs/Parameters

The model can be used on any past vehicle development programs that have followed the ADV process. The following data is required to create and run the model:

- “Clean” vehicle program ADV data, i.e. the data have been filled out completely and each requirement is matched to a legitimate test procedure and has an associated duration. ADV data provides relationships between requirements and test procedures.
- Rainier output. The model starts with an optimized solution set from Rainier, which provides the relationships between test procedures and test vehicles.
- Number of requirements you want to remove. This is equivalent to a budget constraint, for example GM may only be able to focus development on 30 math

analysis initiatives. This again ignores that eliminating requirements costs may be different between different requirements.

- Feasibility data. Feasible requirements are those that should be included in the model. Infeasible requirements are not feasible to replace with math analysis. After running the model with all requirements, the infeasible ones should be excluded as decision variables from the model or permanently turned “on”.

R_t = Total number of Requirements in given hardware stage

T_t = Total number of Test Procedures in a given hardware stage

R_c = Total number of requirements Analysis GROUP wants to eliminate

I = Set of requirements

J = Set of test procedures

K = Set of Vehicles

Mapping Matrix

From the ADV plan, the mapping of requirement to test procedures is put in a matrix format. (see Appendix: Sample Model)

$RT_{ij} = \{ 1, \text{ if requirement } i \in I \text{ is associated with test } j \in J, 0, \text{ otherwise} \}$

From the Rainier output, the mapping of test procedure to vehicles is also put in matrix format. (see Appendix: Sample Model)

$TV_{jk} = \{ 1, \text{ if test procedure } j \in J \text{ is associated with vehicle } k \in K, 0, \text{ otherwise} \}$

4.4.2 Decision Variables

There are three primary decision variables sets: requirements, test procedures, and vehicles. Although we are mostly interested in which requirements affect the vehicle count, the model requires that the test procedure and vehicle also be binary decision variables, so that a forcing constraint can be imposed. (See 4.4 Constraints for details.) The decision variables are formulated as follows:

$$\mathbf{R}_i = \{ 1, \text{ if requirement } i \in I \text{ is tested using hardware} \\ \{ 0, \text{ if requirement } i \in I \text{ is tested using math analysis.} \}$$

$$\mathbf{T}_j = \{ 1, \text{ if hardware test procedure } j \in J \text{ is required} \\ \{ 0, \text{ if hardware test procedure } j \in J \text{ is eliminated} \}$$

$$\mathbf{V}_k = \{ 1, \text{ if vehicle } k \in K \text{ is required} \\ \{ 0, \text{ if vehicle } k \in K \text{ is eliminated} \}$$

4.4.3 Objective Function

As discussed above, the objective is to minimize the vehicle count for a given hardware stage. So the objective function is the sum of the prototype vehicles required for a hardware stage.

$$\text{Minimize } \sum_{k=K} \mathbf{V}_k$$

4.4.4 Constraints

The first constraint of the model is how many requirements we can eliminate. Currently the model is set up only to put a numerical constraint on \mathbf{R}_c ; however; cost and development time constraints could be added to the model later.

$$\sum_{i \in I} (1 - \mathbf{R}_i) \leq \mathbf{R}_c$$

Forcing constraints were also added to the model to force the \mathbf{T}_j and \mathbf{V}_k decision variables to be 1 or 0, based on the summation of requirements and test procedures.

$$\mathbf{T}_j * \mathbf{R}_i - \sum_{i \in I} \mathbf{R}_i \geq 0$$

(Forces \mathbf{T}_j to be 1 if requirements are greater than 0).

Note that the summation is over requirements i that are tested on test procedure j ($RT_{ij} = 1$)

$$\mathbf{V}_k * \mathbf{T}_i - \sum_{i \in I} \mathbf{T}_i \geq 0$$

(Forces \mathbf{V}_k to be 1 if test procedures are greater than 0).

Note that the summation is over tests i that are assigned to vehicle j ($TV_{ij} = 1$)

4.4.5 Model Outputs

4.4.5.1 Requirement and Test Procedure Output Matrix

The model outputs the following information in a matrix format, based on requirement and test procedure binary decision variables. Only the model uses this information.

$Q_{ij} = R_i * RT_{ij}$ Whether requirement $i \in I$ is tested by hardware test procedure $j \in J$.

$Y_{jk} = T_j * TV_{ij}$ Whether test procedure $j \in J$ is configured to vehicle $k \in K$.

4.4.5.2 Model Objective Output

Two outputs of particular importance to the user are the number of minimized vehicles and requirements “removed”. Given the constraints, the model will choose which requirements should be “removed” to achieve the minimized number of vehicles, and it is these requirements that we should scrutinize for trends in resource allocation. The minimized vehicles will give us an indication of how many vehicles could be eliminated with improved math analysis.

The solution may include both feasible and non-feasible requirements. We will refer to these as impact requirements, or requirements that have the greatest impact on reducing hardware.

4.5 Analysis Procedure

For the initial analysis, data from a small derivative car program was used, including three hardware stages. This derivative program, called Q program, has only a fourth of the requirements of a full program. This initial data set will provide insight to how the model works and characteristics of the impact requirements. The Q program was chosen because it was a small data set, easy to work with, and had readily available complete data.

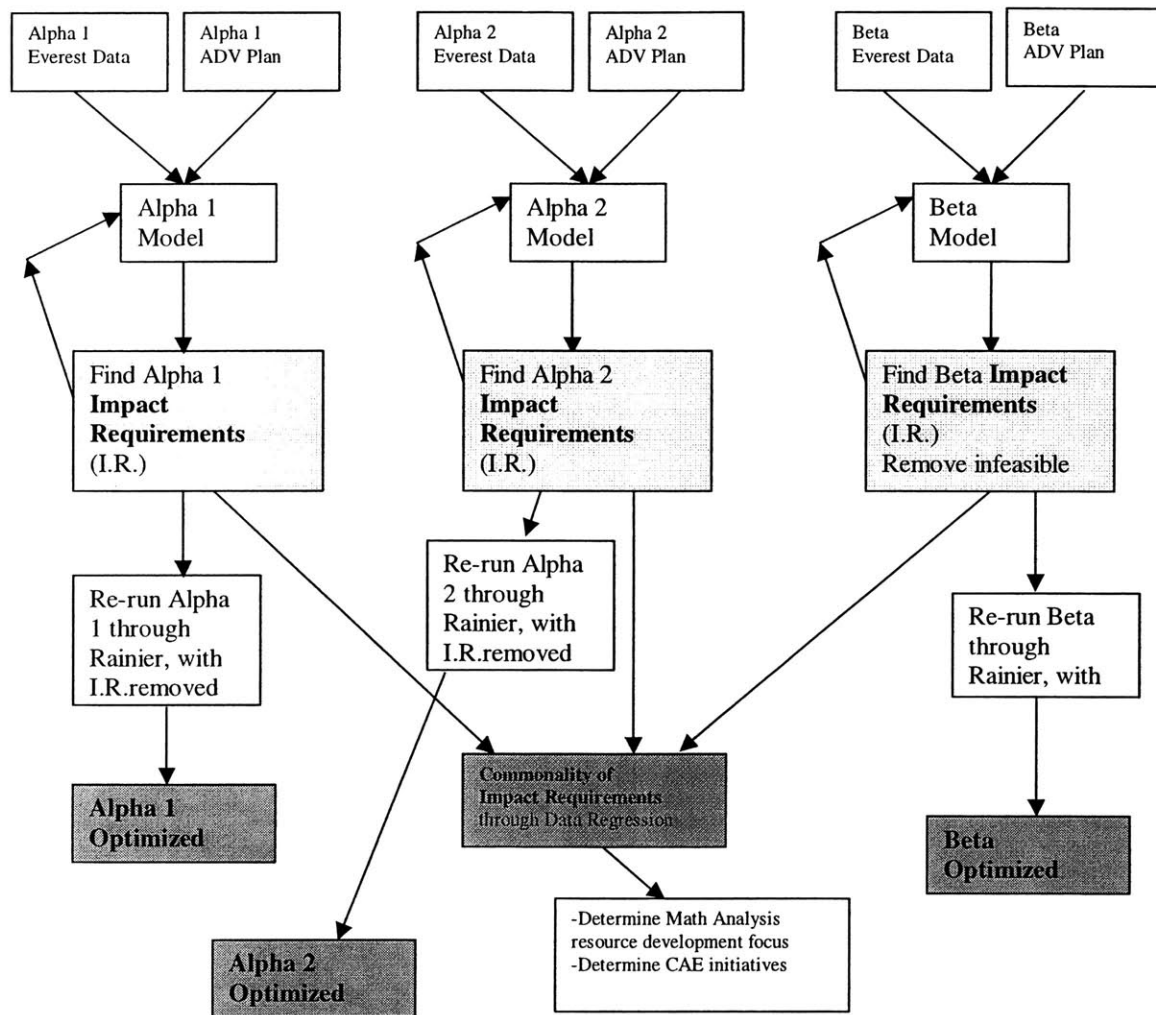


Figure 6. Modeling Analysis process flow diagram for Program Q.

Using the Q program as an example, Figure 6 shows the flow diagram for our analysis methodology, which can be used subsequently on any other vehicle program that follows the ADV process. The results from different programs and hardware stages should be aggregated to determine if any commonality exists between different programs.

The following is a step-by-step description of the analysis methodology.

1. Generate model from vehicle program's ADV plan and Rainier output data.
2. Run model to determine impact requirements.
3. Analyze solution including infeasible requirements for trends or insight.
4. Rerun everything but feasible requirements through model to determine a revised set of impact requirements.
5. Run feasible impact through Rainier to determine optimal test fleet. This requires revising the ADV plan to reflect the use of analysis tools for the identified requirements and rerunning the remaining hardware testing data through Rainier.
6. Repeat steps 1-5 for the different hardware stages.
7. Determine commonality for impact requirements between different hardware stages, if any.
8. Repeat steps 1-7 for all the different vehicle programs under examination.

4.6 Validation and Preliminary Results

After creating the model and running preliminary scenarios, it was necessary to confirm the validity of the model in two different areas. The first was that the model produced near-optimal solutions. The second was to confirm that the model produced results that were at least as good as current resource allocation methods.

Addressing the first question was relatively simple, just run the model on a data set with a known optimal solution. This, however, could only be done with a small model due to the combinatorial nature of the problem. The results showed that the model worked well, since the solution matched the known optimal solution set. This suggests that the program optimizes for a small data set, but to test the robustness of the model and Solver, a larger data set with a known optimal or near optimal solution is needed, and this was not available.

For the second task of confirming that the model produced results as least as good as current resource allocation methods, we turned to the capability matrix and the experience of analysis engineers who worked on the Q program. We produced a baseline resource allocation scenario

which identified how the current resource allocation method (resource allocation determined by focus groups) would remove 30-40 requirements.

The enlisted engineers had been on the Q program and were also quite familiar with the capability matrix so they were able to choose “the best” 30-40 requirements that could be removed. After “removing” 30-40 requirements, the data were then re-run through Rainier. Figure 9 below shows the results of the three different methodologies. The first column shows the requirement and vehicle reduction results from the solver model. The second column shows what happened when the chosen requirements were re-run through Rainier to determine the re-optimized fleet size. The third column shows the results of removing the requirements chosen through the baseline methodology and then running the remaining test set through Rainier. The results show there is approximately a two-fold reduction in vehicles using the model and re-optimizing with Rainier.

Model results			Model through Rainier			Baseline through Rainier		
Remove 30 Requirements			Remove 30 Requirements			Remove 30 Requirements		
Count		%	Count		%	Count		%
Vehicle reduction	24	8%	Vehicle reduction	19	27%	Vehicle reduction	23	12%
Requirement reduction	30	15%	Requirement reduction	30	15%	Requirement reduction	30	15%
Remove 35 Requirements			Remove 35 Requirements			Remove 35 Requirements		
Count		%	Count		%	Count		%
Vehicle reduction	23	12%	Vehicle reduction	19	27%	Vehicle reduction	22	15%
Requirement reduction	35	17%	Requirement reduction	35	17%	Requirement reduction	35	17%
Remove 37 Requirements			Remove 37 Requirements			Remove 37 Requirements		
Count		%	Count		%	Count		%
Vehicle reduction	22	15%	Vehicle reduction	8	31%	Vehicle reduction	21	19%
Requirement reduction	37	18%	Requirement reduction	37	18%	Requirement reduction	37	18%
Remove 40 Requirements			Remove 40 Requirements			Remove 40 Requirements		
Count		%	Count		%	Count		%
Vehicle reduction	22	15%	Vehicle reduction	8	37%	Vehicle reduction	21	19%
Requirement reduction	40	19%	Requirement reduction	40	19%	Requirement reduction	40	19%

Figure 7. Solver Model vs. Baseline Scenario

We can see in the second column that after re-optimizing through Rainier the requirements chosen by the model produced results significantly better than through current resource allocation methods.

4.7 Data Regression

Because math analysis feasibility of the requirements was not incorporated into the model and the Q program was not representative of a major vehicle program, the results produced from the model were of limited use in answering the question of where to focus resource allocation. However, the data could be regressed to help create optimization heuristics to solve future versions of the resource allocation problem. With heuristics, you can add rules-of-thumb to avoid infeasible solutions. The heuristics were developed by tracing the requirements to which vehicles they were mapped to, and then identifying characteristics or trends of the requirements.

The first insight gained was by looking at the vehicles eliminated. Data regression showed that each vehicle eliminated had a low number of unique requirements for testing. Other important factors discovered were the number of test iterations for requirements, cross-referenced (requirements tested on several different vehicles) requirements, and test durations.

4.8 Optimization Heuristics

Solver uses algorithms that produce near-optimal solutions, which leaves us with the question; can we produce better answers than the Solver model? Since Solver uses mathematical means to arrive at the solutions, simple rules and patterns are ignored. These rules and pattern may produce better solutions, and is often referred to as heuristics. The data regression provides the means to developing heuristics.

The following heuristic was developed using the findings from data regression. This heuristic should produce similar hardware reduction results to the current Solver model, while eliminating the long computing time. The goal for this heuristic is the same as the Solver model: to reduce the largest number of vehicles by eliminating a given number of requirements. The capability matrix is not currently incorporated in the heuristic. The heuristics' steps are described below and an example of its use is shown in Figure 10. Arbitrary shadings differentiate the vehicles.

1. Sort vehicles by the least number of unique requirements they are validating. (i.e., requirements that are validated multiple times on a test vehicle are counted only once.)

2. Secondary sort list of vehicles by the number of replicate test requirements. Test replicates in this case refer to repeat tests on the same requirements.
3. Select vehicles with least number of unique requirements for “removal”, up to the given number of requirements to be removed.
4. In case of multiple vehicles with equal number of unique requirements, the priority goes to the vehicle with the greatest sum duration referenced to other vehicles.
5. For remaining requirements that can be “removed”, but cannot eliminate a whole vehicle, select requirements with largest sum test duration. This rule removes requirements that are on multiple vehicles and replicated many times that add up to large test time. This minimizes test time required for the hardware stage and when re-optimized allows additional vehicles to be eliminated.

A simple extension is to pick vehicles with capability levels 3 and above. This would add a feasibility factor to the model.

Veh #	Test Proc	Description	# Unique Requirements	Test Duration (days)	Iterations in ADV Plan	Cumulative Requirements Removed
14	TP9790	Chassis - Frame Assembly Target Life	1	120	0	1
26	TBD	Chassis - Tires Wheels Trim - Destructive Tire test	1	10	1	2
6	TP8335	Steering Control Effort	3	1	9	5
6	TP8335	Static Steering Effort	3	1	9	
6	TP0000	Ride/Vibration: Engineering-Level Requirements	3	7	33	
6	TP0000	Ride/Vibration: Engineering-Level Requirements	3	7	33	
6	TP0000	Ride/Vibration: Engineering-Level Requirements	3	20	33	
6	TP0000	Ride/Vibration: Engineering-Level Requirements	3	40	33	
6	TP0000	Ride/Vibration: Engineering-Level Requirements	3	40	33	
6	TP0000	Ride/Vibration: Engineering-Level Requirements	3	40	33	
8	TP8335	Steering Control Effort	3	1	9	8
8	TP8335	Static Steering Effort	3	1	9	
8	TP0000	Externally Induced Vibration	3	40	33	
8	TP0000	Externally Induced Vibration	3	40	33	
8	TP0000	Externally Induced Vibration	3	20	33	
8	TP0000	Externally Induced Vibration	3	7	33	

8	TP0000	Externally Induced Vibration	3	7	33	
7	TP0000	Low/Medium Coefficient Surface Stopping Distance Ratio	3	45	1	11
7	TP0000	Low/Medium Coefficient Surface Stopping Distance Ratio	3	45	1	
7	TP0000	Externally Induced Vibration	3	20	21	
7	TP0631	Road/Tire Induced Vibration	3	5	9	
9	TP0000	N/A	3	45	0	14
9	TP0000	N/A	3	45	0	
9	TP0000	Externally Induced Vibration	3	20	21	
9	TP0631	Road/Tire Induced Vibration	3	5	9	
1	TP9791	Reliability and Durability	5	126	0	19
1	TP9791	Corrosion Resistance	5	100	0	
1	TP9791	Reliability	5	100	1	
1	TP9791	Drive-line Leakage	5	50	0	
1	TP9791	Target Life	5	50	0	
12	TP4309	60.2 Bumpers & Fascias	6	7	0	25
12	TP4432	Corner Noise	6	100	9	
12	TP8318	Linear Range Handling	6	3	9	
12	TP8318	Linear Range Handling	6	3	9	
12	TP8318	Linear Range Handling	6	3	9	
12	TP8335	Steering Control Effort	6	1	9	
12	TP8335	Static Steering Effort	6	1	9	
12	TP0000	Externally Induced Vibration	6	3	21	
16	TP4040	Straight-Line Launch Maneuvers	9	15	0	34

16	TP4623	Accelerating in a Turn	9	15	0	
16	TP4623	Steering While Accelerating	9	15	0	
16	TP0000	Automatic Transmission TCC and Shift Modification	9	10	0	
16	TP4040	Acceleration Feel During Non-TAS Acceleration	9	10	0	
16	TP4040	Acceleration Feel During TAS Acceleration	9	1	0	
16	TP0000	Driveline Vibration	9	30	1	
16	TP8516	Mountain Descent Capability	9	15	1	9 Vehicles
16	TP4328	??	9	25	0	34 requirements
22	TP0000	Mounting	9	1	0	43
22	TP0000	Mounting	9	1	0	
22	TP4630	GVMR Stopping Distance	9	18	1	
22	TP0000	Low/Medium Coefficient Surface Stopping Distance Ratio	9	45	3	
22	TP4630		9	12	5	
22	TP4630		9	12	6	
22	TP4630		9	12	6	
22	TP8516		9	30	1	10 Vehicles
22	TP0000		9	5	0	43 requirements

Table 4. Heuristic Optimization Example

Using this heuristic we are able to achieve a reduction of 9 vehicles by eliminating 34 requirements, or a reduction of 10 vehicles by eliminating 43 requirements. A comparison of the heuristic result vs. the Solver model result is shown in Table 5.

Heuristic Model			Solver Model		
Remove 40 Requirements	#	%	Remove 40 Requirements	#	%
Vehicle reduction	9	35%	Vehicle reduction	7	27%
Requirement reduction	40	19%	Requirement reduction	40	19%
Remove 35 Requirements	#	%	Remove 35 Requirements	#	%
Vehicle reduction	9	35%	Vehicle reduction	5	19%
Requirement reduction	35	17%	Requirement reduction	35	17%

Table 5. Heuristic vs. Solver Model

4.9 Modeling Issues

The Solver model has its limitations, as discussed below, and the user should be aware of them. However, the model in its current form can still be used to help make planning decisions. The modeling and implementation issues are listed below:

- The biggest implementation issue encountered is how do you use the model to answer cross-program resource allocation problems? The issue lies in the fact that the solution is unique to the data set and there is no guarantee that the same vehicles will be eliminated on subsequent vehicle programs. Also, since the ADV plan is constantly changing, how do you plan for the dynamic future using the static past? Since the requirements and test procedures are usually mapped using past ADV plans as a template, there is a likelihood that eliminating all the requirements mapped to a certain test procedure will eliminate that test procedure on future programs. Rainier, on the other hand, has more variation in mapping test procedure to vehicles but, because of sharing restrictions on vehicles, the chances for a trend are increased. The past data is the best available resource for planning future math analysis development, but it is difficult to gauge the effectiveness of this method.
- Since each program and hardware stage within a program is unique, what is the most logical way to aggregate the results to ensure the overall modeling objective is achieved? Do we aggregate all the results from one program and compare them with another program? Do we aggregate all the results from a hardware stage for all the programs for a given and compare

with another hardware stages? Section 4.5 proposes a method, but its effectiveness still needs to be evaluated.

- As noted above, the current model does not incorporate the benefits of incremental analysis capability in the model. For example, the model does not consider that it may require less resources to bring an existing level 2 capability to a level 3 than it would to bring a new math analysis tool to level 3 capability. In its current state, the capability matrix has issues that need to be addressed prior to incorporating it into the model.
- Ideally you would want to link potential CAE initiatives to hardware, run the optimization, and then select the CAE initiatives to focus on. However, there is a disconnect between the requirements and the math analysis tools. Knowing which requirements reduce hardware does not mean we know which math analysis tool would migrate that requirement to Math-Based testing. The model specifies which requirements impact hardware, but it is up to resource planning to translate this information to determine the best math analysis development plan. (Linking CAE initiatives to hardware is further discussed in Section 4.10.3.)
- Specifying how many requirements to migrate towards math analysis is not an accurate way to depict resource constraints. Resources are best represented with a budget constraint or human resource constraints. However, no data source linking math analysis tools to money or human resource is currently available.
- Since a larger data set would require clean-up time that we did not have, the model was not tested for robustness. Also, because of constraints with the number of columns you can have in Excel, there could be complications with using the Solver model on large data sets. For this reason, solving complex problem with heuristic-based techniques is recommended. Not only would this address the technical issues, but it would also facilitate an interface between the front-end (requirement optimization, R.A.O.M) and the back-end (vehicle configuration, Rainier).

- Models are only as useful as the data input, and this certainly applies in this case. Since the ADV process is still under development, not all vehicle programs follow consistent data format. The vehicle programs that produce ADV plans in the correct format still require manual manipulation and scrubbing of the data. For example, there are many instances where test procedures are given default names (e.g. TP0000), thus diluting the effectiveness of the model.
- The solution set determined by the model may not be the only solution set. Another set of requirements may also yield the same or better reduction in hardware.

4.10 Extensions to Model

4.10.1 Create Heuristic Optimization Model

Since the optimization heuristic discussed in section 4.8 had better results and performance than the Solver model, development of a heuristic optimization program would be a logical first step. Not only would the heuristic model solve more quickly, but it would also facilitate the front-end (heuristic) connecting to back-end (Rainier). The heuristic model would have the same applications and assumptions as the Solver model.

4.10.2 Incorporate Capability Matrix into Model

After creating a heuristic optimization model, the next step would be to integrate the Capability Matrix into the model to address the feasibility of each requirement being replaced by math analysis and capture the benefits of incremental analytical improvements. The methodology for this model is shown in Figure 8.

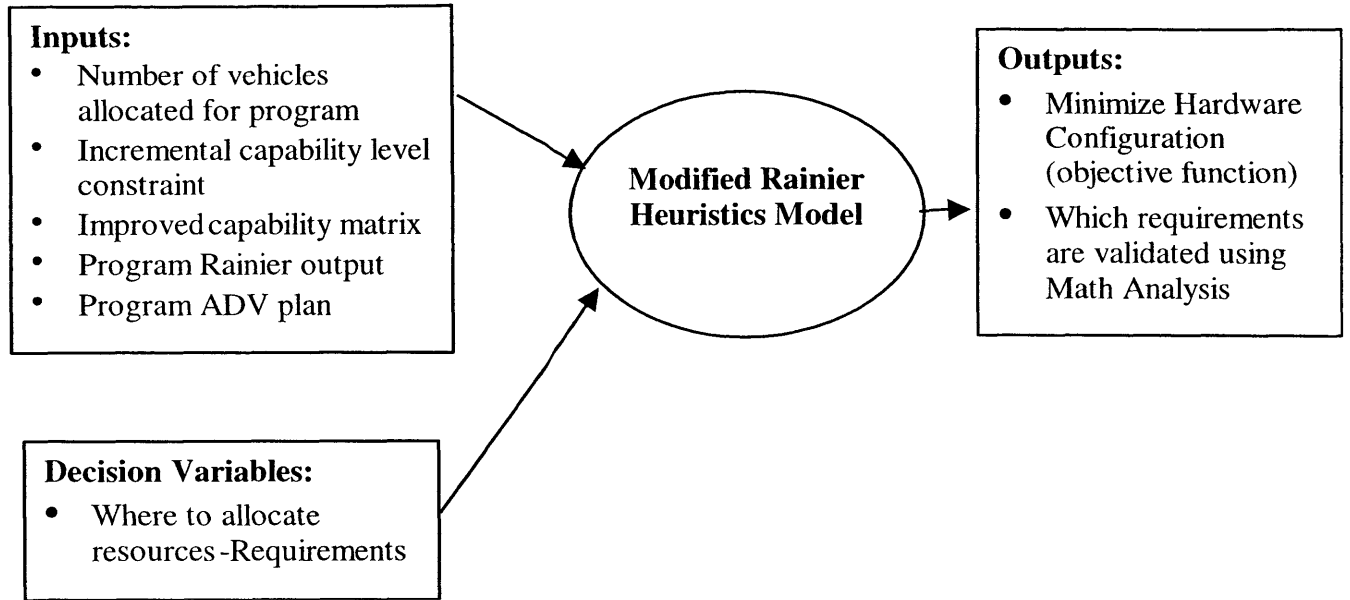


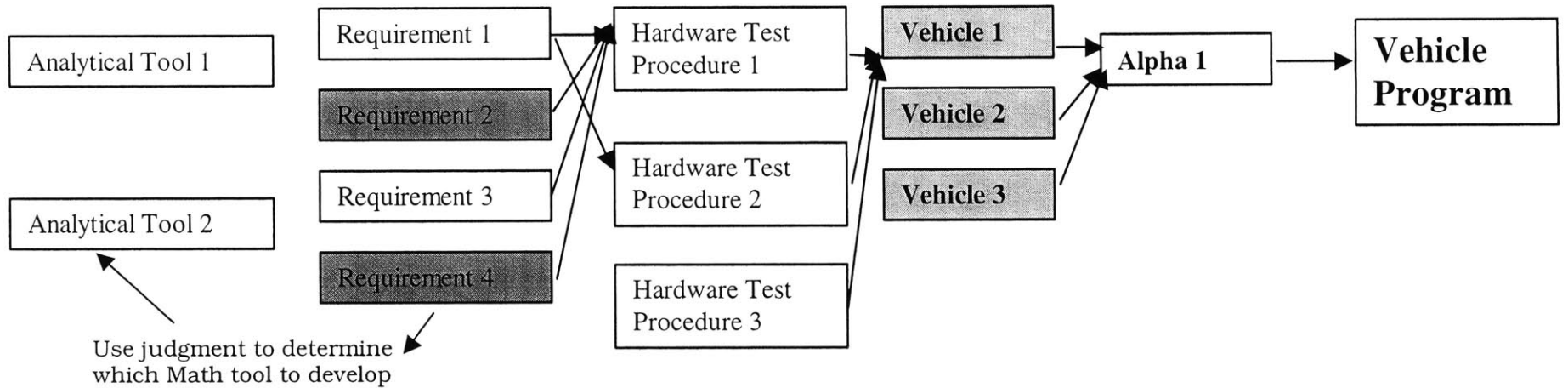
Figure 8. Heuristics Optimization Algorithm

One application for this model would be to address current program allocation issues, especially if vehicle allocation is a constraint. For example, if a given vehicle program was limited to 40 vehicles, the new heuristic model would be able to identify the minimal number of requirements that need to be validated analytically to meet the vehicle allocation. The model would also be able to discriminate whether or not it would be feasible for the requirements to be validated analytically or with hardware. However, this model is dependent on the capability matrix issues being addressed as discussed in Section 3.4.3.

4.10.3 Linking Software Initiatives to Vehicle Reduction

Another extension to the model is to link the yearly software initiatives directly to the vehicle reduction. In the current model, only the requirements are linked to hardware reduction; it is up to management to determine where to allocate resources for tool development. Linking software initiatives to vehicle reduction would be a more accurate way of determining the impact of resource allocation. The following figure shows the difference between the current Solver model and the proposed model.

Current Model tells us which requirements minimizes vehicle count



Extended Model tells us which CAE initiatives minimizes vehicle count (includes cost constraints on CAE initiatives)

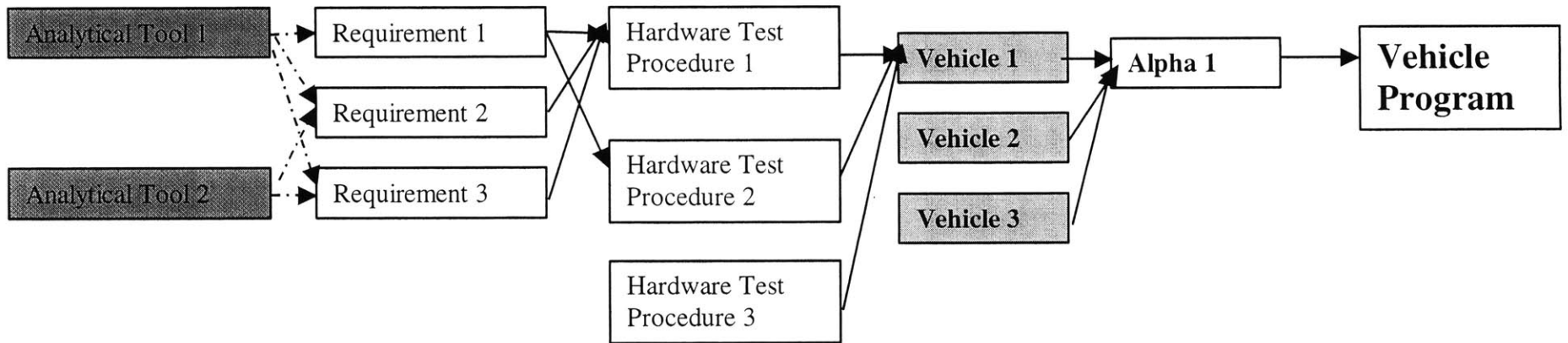


Figure 9. Mapping Software Initiatives to Vehicle Reduction

The following tasks need to be completed to create the extended model:

- 1) Identify the yearly CAE initiatives and which requirements they map to
- 2) Have focus groups identify the cost of each initiative
- 3) Have focus groups identify the capability increase of each CAE initiative
- 4) Improve Capability Matrix metrics
- 5) Create new model to solve using modified algorithm incorporating new cost constraints and capability matrix
- 6) Estimate Analysis group's resource budget

4.11 Technology Conclusions

The deliverables produced during the development of the solver model include the model scope, a working Solver model, and heuristics that can be used to develop future optimization models. Although the model has its share of technical and data acquisition issues, developing the model has had several latent benefits.

The first benefit was an increased understanding of the math analysis resource allocation problem. The project originally started with a broad problem statement, but as we increased our understanding of the problem and the customer requirements, we learned what we can and cannot answer. A model scope was created to give the project boundary. Working on the model also allowed us to identify what data is required and areas within the ADV process that needed improvement, such as a need for consistent ADV data format, or the usage of the Capability matrix.

Another benefit from creating the model was that the increased understanding of the problem could be used to help focus future cost reduction efforts. This project was focused on proof of concept and identifying how to use the solution to determine resource allocation. The next step is applying the learning to steer the direction of future projects, as well as increase efforts to address the issues identified during the project.

Finally, the project brought out organizational and cultural development issues. Providing the design and development community with the best math analysis tool does not ensure that they will use it, and without that, the company won't benefit from their investments. It is with this final point that motivates the excursus of socio-technical development.

5.0 Socio-Technical Development

5.1 Organizational and Cultural Development Goal

The second goal of this project is to understand how to affect change and to increase math analysis usage. Before answering the question of how to get the organization to move to math, we have to know the right questions to ask. Some of these questions include:

- What are the different enablers or factors that facilitate change? These enablers should not only identify why people want change, but also what indirect factors cause change.
- What are the different disrupters or factors that cause people to resist change? To identify the disrupters we must identify some of the changes that people are afraid of. Examples of changes might include: loss of job security, loss of expertise or identity, a need to learn new skills, shifts in influence or authority, shifts in communication patterns, and re-organization.
- Finally after identifying what some of the enablers and disrupters, what policies can we create to address them?

By answering these questions we will gain a better understanding of how to affect change with math analysis usage.

5.2 Identifying Causal Factors

The first step in this study was to conduct qualitative interviews with key decision makers regarding the decision whether to use analysis or hardware testing. The decision-making hierarchy is shown in Figure 10. At the top of the matrix organization, the cross-functional project team ultimately reports directly to their Engineering Director and indirectly to the Vehicle Line Executive (VLE). The Engineering Director and the VLE usually are not involved in engineering decisions but ultimately have veto power over any decision. Next in the hierarchy are functional managers. The program director is responsible for the vehicle program; the Total

Vehicle Integration Engineer (TVIE) is responsible for integrating all the engineering systems; and the Chief Engineer is responsible engineering designs. They are closer to the engineering design, but do not actively involved in engineering decisions, but do participate in design conflict resolution.

The next tier consists of the key decision makers. These decision makers are ranked in order of influence on the decision-making process and they include Validation Engineers (VE), Analysis Engineers, Design Release Engineers (DREs), and Development Engineers. The Validation Engineer is responsible for the activities related to testing the requirements to ensure they meet specification; the Analysis Engineer provides the math analysis for the car design; the DRE is responsible for the design of their part, and the Development engineer represents the customer and is responsible for the feel of the designed components. There was some confusion between the different interviewees about who has the most influence over the use of math analysis; however, most agreed that the DRE and the Development Engineer have the most influence.

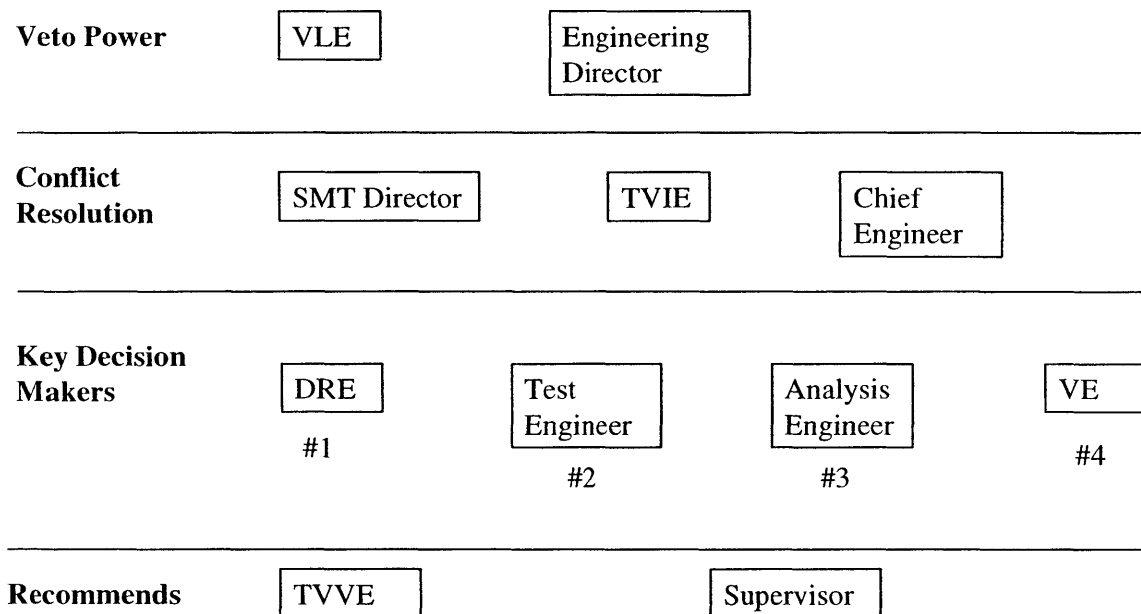


Figure 10. Decision-Making Hierarchy

Lastly, are the Total Vehicle Validation Engineers (TVVE) and Engineering Supervisor who do not affect the decision but provides input that might affect the decision.

The qualitative interviews allowed us to identify factors that disrupt and enable change, facilitate discussion on relevant topics, and get a feel for cultural and organizational differences among the decision-making parties.

The qualitative interviews were conducted using the three organization behavior lenses (strategic, political, and cultural lenses). This lens theory states that most people already have a schema or pre-conceived notion about a problem or issue. However, when we expand our perception we are able to understand and uncover the entire issue, the three lenses were developed to put us into alternate mind frames designed to expand our perceptions. The qualitative surveys using the three lenses can be found in appendix B.

Using the information from the qualitative interviews, a survey was carefully crafted to measure the relative strengths of 24 factors, the correlation between different factors, and on their influence on the decision to use math analysis. (Quantitative survey can be found in appendix C).

5.2.1 Survey Results

The survey included three filter questions to separate the different demographics of job function, issue position, and job experience. The anonymous survey was then mailed to functional engineering group managers of Analysis Engineers, DREs, Validation Engineers, and Development Engineers to cascade through their teams. A total of 76 surveys were sent, to which only 27 people responded. Because of the small sample size the results have limited statistical significance. Using the 24 different factors, 24 charts were plotted for each of the three filters, and 24 charts for the cumulative response. There were also two charts plotting the filters against each other, for example a chart plotting the experience vs. the job function. There was a grand total of 98 charts. Some of these charts can be found in appendix D. The survey data also allowed us to determine the influence factors have on each other, which allowed us to create a system dynamics

model of math analysis in the ADV process. The following examples illustrate how the surveys were used to determine the relationships for the system dynamics model.

In Figure 11, the graph shows overall responses to the question of whether math analysis improves product quality. The results indicate that most respondents felt math analysis does help improve product quality.

How has the quality of the product or design been impacted by Math Analysis ?

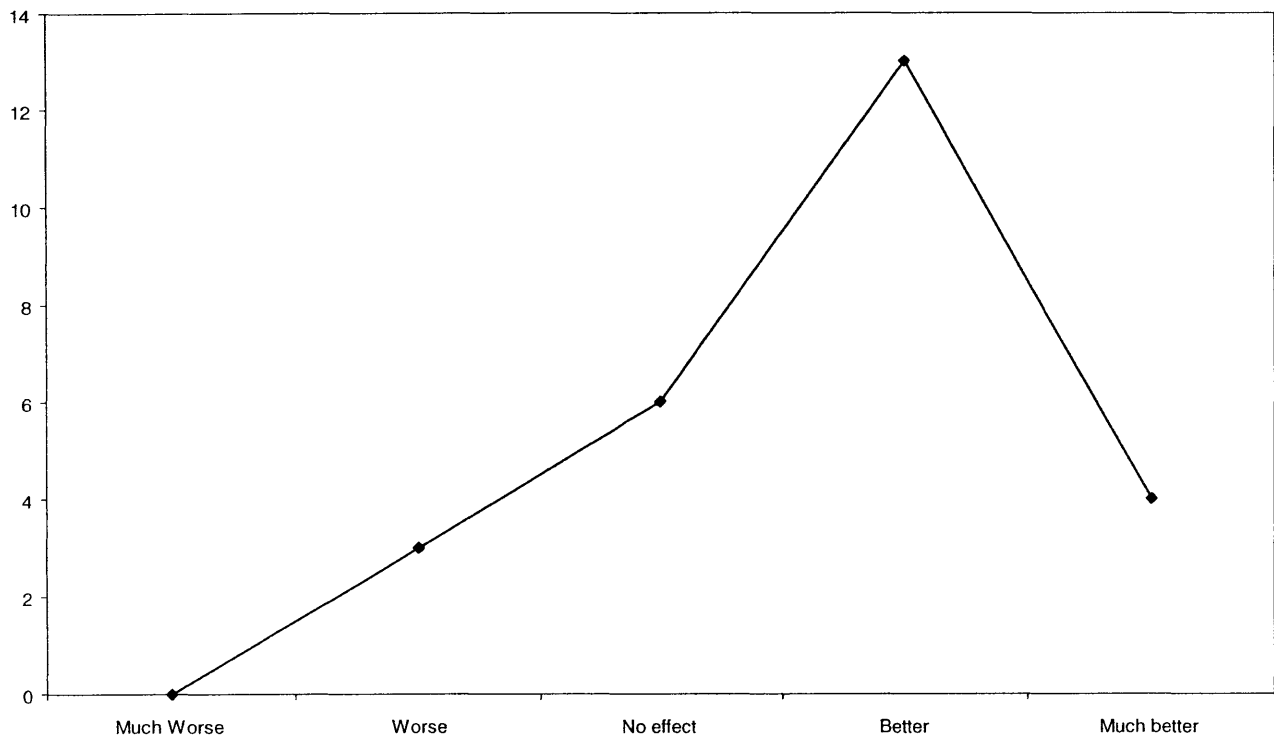


Figure 11. Position on Whether Math Analysis Improves Quality

Using the experience filter, we can see how the “years of experience” demographics differed in their confidence in the math tools.

Do you generally feel confident with the results from Math Analysis?

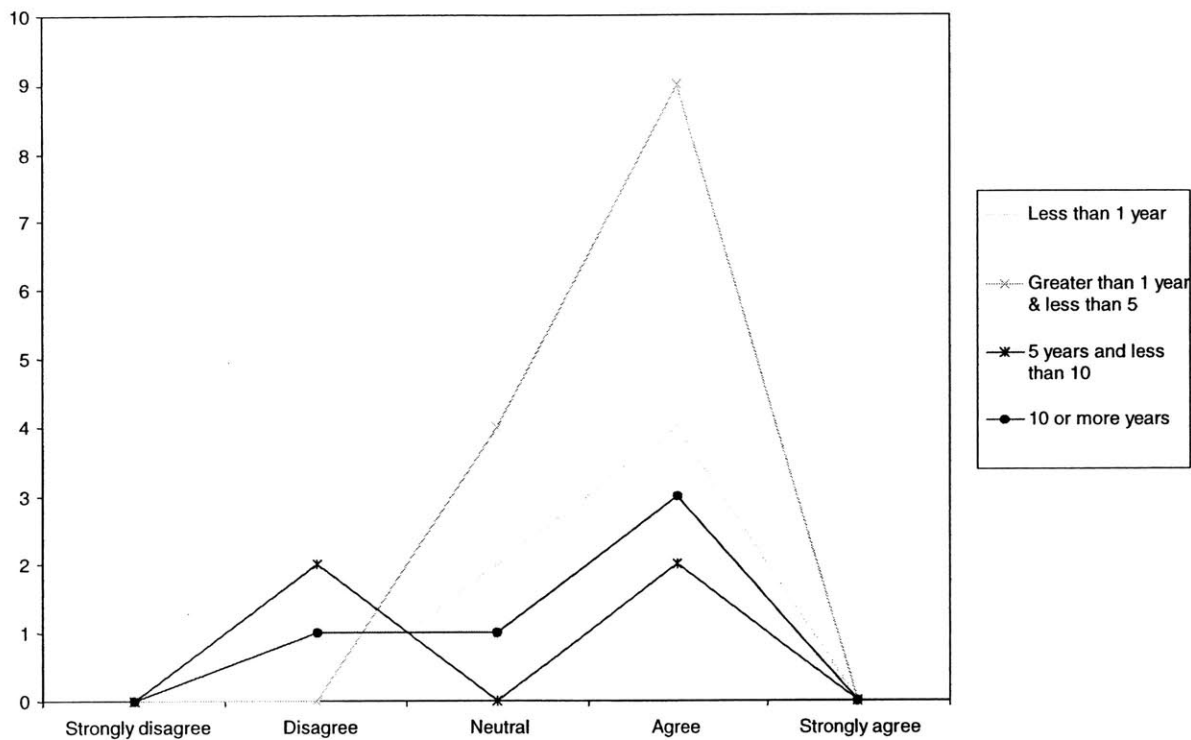


Figure 12. Confidence in Math Analysis vs. Years of Experience

Figure 12 shows that people with less experience were mostly neutral or confident in math analysis. We can draw the assumption that people with less experience begin with the disposition to trust math analysis. However, the results showed that the more experienced demographic has a more diverse distribution. For example, the results of the respondents with 5-10 years of experience were bi-modal. We can conclude from this graph that people who have more experience have a culture less trusting of math analysis than people with less experience, although we cannot conclude the cause of this mistrust from the graphs.

Do you generally feel confident with the results from Math Analysis?

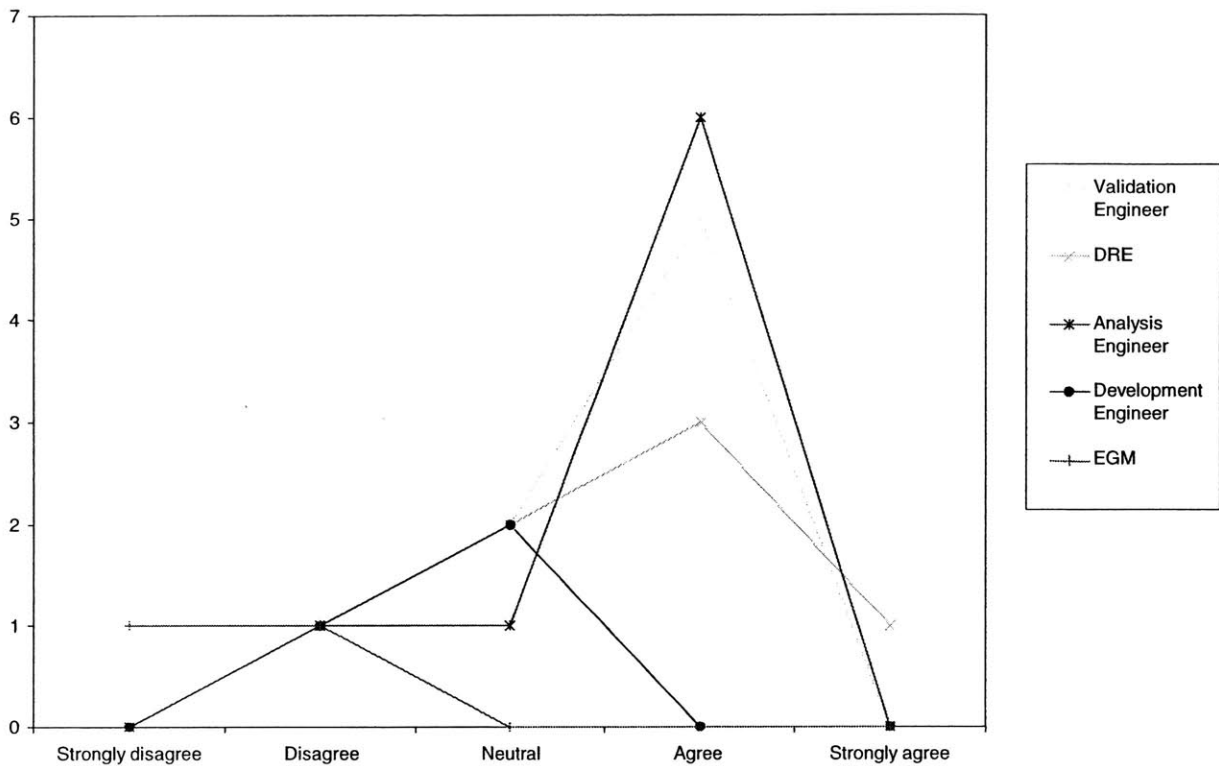


Figure 13. Confidence in Math Analysis vs. Job Function

The graph in Figure 13 plots confidence in Math Analysis by job function. These results will help to quantify the difference in culture between the different work groups. The above graph indicates that Analysis and Validation Engineers were generally confident in math analysis. Disparity starts with DREs, and Development Engineers (based only on a sample size of 3), who were more divided in their confidence in math analysis.

Using all the findings from the quantitative data, we extracted the factors that had a high correlation or that revealed trends relevant to the system. For example, knowing that employees felt there was a gap between management’s perception of math analysis capability and actual capability, we would incorporate this factor into the system dynamics model. The following describes the system dynamics model of math analysis usage is based on the findings from the interviews and surveys.

5.3 Math Analysis Usage System Dynamics Model

Most managers are aware that culture-related issues are difficult to identify, and even harder to address. Numerous researches have been done on change management; however, this one-size-fits-all approach does not take into account the differences in organizations, and the uniqueness of each problem. System dynamics modeling was chosen for this project because it customizes to the problem, captures cultural issues, organizational issues, and is excellent for illustrative purposes. System dynamics allows us to address a subjective topic, culture, in an objective manner by analyzing the cause and effects of certain factors. Many of us are aware of the cultural differences in groups, but rarely are they addressed, instead we tend to focus on organizational changes or technology improvements. We also tend to focus on our own area and how the surrounding environment affects our group, but we rarely consider how our actions may affect the whole. System dynamics integrates all the systems as a whole.

System dynamics utilizes causal loop analysis, feedback control, and can be used for simulations. Causal loop analysis shows how certain factors in a system affect other factors, how these factors feed back into the system, and ultimately how these factors affect the end goal. If we wish to simulate the system under different scenarios, quantitative relationships need to be added to the model.

System dynamics modeling can also identify how enablers and disrupters affect the whole system. Identifying the disrupters and enablers provides opportunities to implement or modify existing policies to suppress the disrupters and facilitate the enablers.

The goal of the system dynamics model is to increase the percentage of tests that are math-based. In other words, given a set number of requirements that need to be tested for a program, how can we increase the percentage of requirements validated using math-based testing? We represent the goal as a stock called “% Math-Based Testing”. The flow going in will be the adoption rate; to keep the diagram clean, no outflow is used. Each loop is named, identified as balancing or reinforcing, and has different shadings. A lightly shaded loop means that, in its current state, it is

not helping to increase the adoption rate (disrupters). A dark shaded loop is actively increasing the math analysis adoption rate (enablers).

The data that was used as a basis for the model was limited due to lack of responses. In addition, the model does not include every variable or factor that is affecting the system. However, the major components of the system are included and should help us gain a better understanding of the math analysis usage problem.

5.3.1 Correlation Loop

To prove that math analysis tools work, there needs to be data correlating the math test to the actual physical test. During development both a math model and a physical model is created. The two models are compared to see how close the tested parameters were. The correlation of the hardware to the math model is affected by two factors: how accurate the math model is and how accurate the hardware model is. The math model can be inaccurate because of the Analysis Engineering's expertise in creating the model and effort put into creating the model. The hardware model can be inaccurate because it is built with prototype parts or non-production manufacturing processes.

given by management. Analysis resources and priorities are exogenous variables in this system.

We can see from this loop that hardware accuracy may be an issue if resources are not sustained. Likewise it is difficult to sustain math model accuracy because of limited capacity, such as time and human and computer resources. Only the “Analysis Capability” loop is helping to improve correlation. The general outcome from these three loops is that correlation results are neutral, but slowly improving.

The insight that we can draw from these loops is that in order to increase correlation, analysis resources need to be increased and correlation effort needs to be a priority. The correlation loop is important since it feeds many downstream loops.

5.3.2 Trust Loop

Having highly correlated data doesn't necessarily mean that the math analysis adoption rate will increase. Another facet to getting people to use math analysis is trust. Correlation is data or fact driven, but trust is intrinsic and driven by culture and experience.

As modeling experience increases, so does understanding of the actual math analysis capability, which builds confidence in the decision to use math analysis. This confidence helps to increase trust in math analysis. However, as the math analysis technology grows, so does the gap in understanding math analysis capability. This gap, which lessens over time for the users due to learning and experience, also causes management to overestimate the capability of math analysis. This is because as analysis capability increases, the disconnect between management and technology increases. Managers only know what the software suppliers tell them, but without using the math tools, they do not have full understanding. This perception gap undermines the employees' confidence in the decision to use math analysis, which ultimately affects trust.

Another factor that affects trust, which was much more difficult to uncover, was hardware culture, or bias against using hardware. Two things created this culture: experience and personal bias. Experience refers to how long the employee has been working with hardware, as well as their experience with math modeling. Unfortunately, without the actual experience of replacing hardware with math analysis, experience cannot be acquired. It's similar to purchasing on the Internet. Until you've purchased several times you would not have the experience to make a judgment on Internet security. People also have formed personal biases against math analysis for various reasons. A common personal bias was that users felt hardware revealed things that math analysis could not, such as noises, rattles, or sensations.

The current status of trust is very low since, according to the interviewees, correlation is neutral, while "Hardware Culture" is a deterrent to building trust in math analysis. Management should try to focus on the factors affecting trust, such as improving hardware culture or correlation, and trust employees' assessments on which math analysis is best suited for the job. Examples of these policies include, cross training employees in both analysis, design, and development to improve culture and increasing analysis resources to improve correlation efforts.

5.3.3 Culture Loop

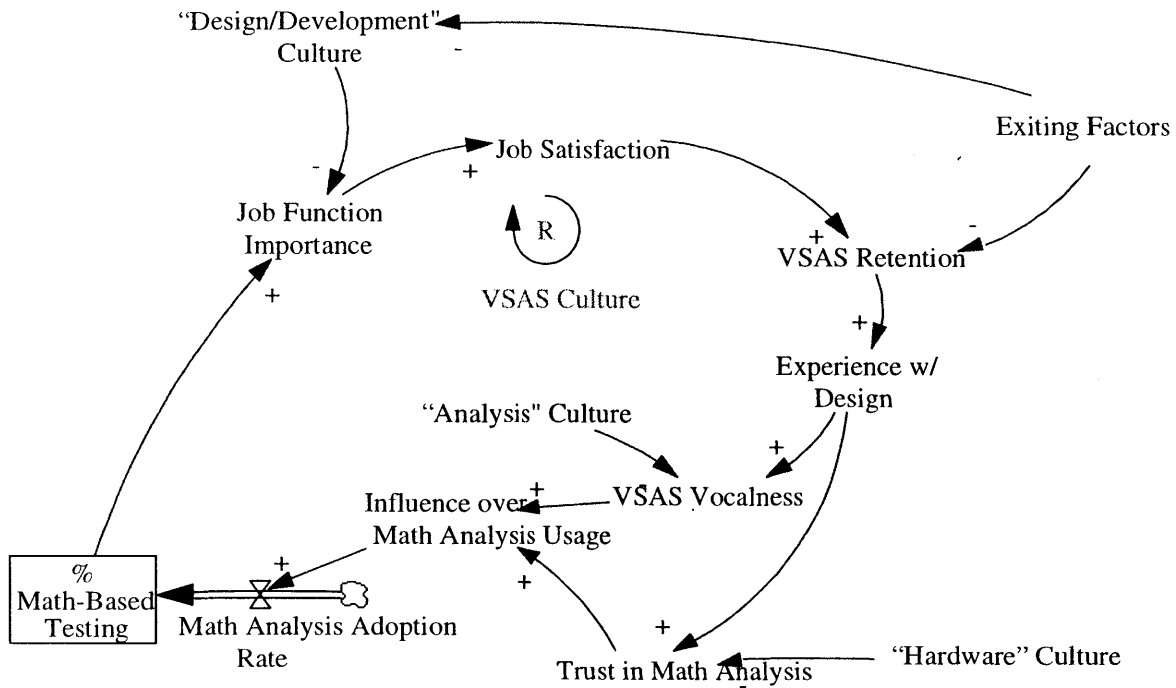


Figure 16. Culture Loop

The loop in **Figure 16** is positive towards increasing math analysis, particularly since it is reinforcing. As math-based testing increases, Analysis Engineers start to feel that their job is more important, which leads to job satisfaction and job retention. The longer the engineer is working in the group, the more design experience he/she gains. Design experience in this case refers to both creating accurate math models and experience with physical design, such as intuitively knowing where to put fasteners or welds. This experience also helps to strengthen the trust in math analysis. As an Analysis Engineer gains more experience, the DREs and Development Engineers they work with also gain more confidence in their modeling ability (similar to finding a good mechanic). Also, as their experience increases, they are more apt to speak up and vocalize their opinion, since they are more confident in the model. This should work to increase their influence over math analysis usage.

Two factors suppress this loop. The first factor is the lack of job importance or influence that the Analysis Engineers have due to the organizational culture. Since Analysis is seen as a service organization to the design and development community, they have little influence on math analysis usage. The other factor is that Analysis engineers do not work on the same parts long enough to build experience. There are numerous factors causing this, which are lumped into “exiting factors”. Factors that affect Analysis retention include limited career mobility, redundant work, and lack of job satisfaction. There is one latent benefit from the low retention rate; as analysis engineers leave, they generally move to a design or development position, alleviating the cultural barriers. These people are better able to relate the experiences they had as Analysis Engineer to their design and development colleagues.

To get the full benefit from this loop, we need to remove or weaken the externalities. Policies to change the hiring practices in order to bring in more vocal Analysis Engineers, while increasing job retention would help build analysis influence. There are also cultural barriers that cause a lack of empathy and understanding between the groups. These barriers can be taken down through cross training, co-locating, and team building.

5.3.4 Cost Pressure Loop

One of the main factors currently driving math analysis is cost pressure. As seen in Figure 17, as math-based testing increases, fewer hardware tests need to be performed and thus hardware cost decreases. As hardware decreases, development time and resources also decrease. Math-based testing reduces program costs through either hardware reduction, decreased development time, and/or decreased hardware test resources (cost associated with building and testing hardware), which results in reduced GM operating expenses. This leaves more available budget, resulting in less pressure to reduce VDP cost. This decrease in pressure will cause management to shift focus to other areas and relaxing the pressure to limit vehicle allotment.

5.3.5 Incentives Loop

This loop in Figure 18 describes the incentives for Validation Engineers, DREs, and Development Engineers to use math analysis. There are three incentive loops that ultimately affect whether or not they have a positive experience with math analysis. These three incentives are time savings, product quality, and positive experience. If correlation is positive then hardware testing does not need to be done, decreasing development time, which increases personal work time, creating a positive experience. Also, as development time goes down, all program schedule goals are met, creating a positive experience. Finally, correlation can affect the design risk. If correlation is positive, engineers are better able to predict design behavior and design a better product. Design risk is reduced and the quality of the vehicle is consequentially improved, creating a positive experience with math analysis.

Design risk and the quality of the vehicle can also affect management's perception of math analysis. Without good correlation, design risks increase which eventually affects the vehicle quality, causing managers to have differing opinions on the usefulness of math analysis. Coupled with a lack of accountability for math analysis usage, managers can become ambivalent about driving math analysis.

The insight gained from this loop is that most of the incentives are spawned from correlation and end with positive experience. However, since correlation currently is weak, there are no strong incentives for engineers to use math analysis for validation. However, even with weak correlation, there is still incentive to use Math

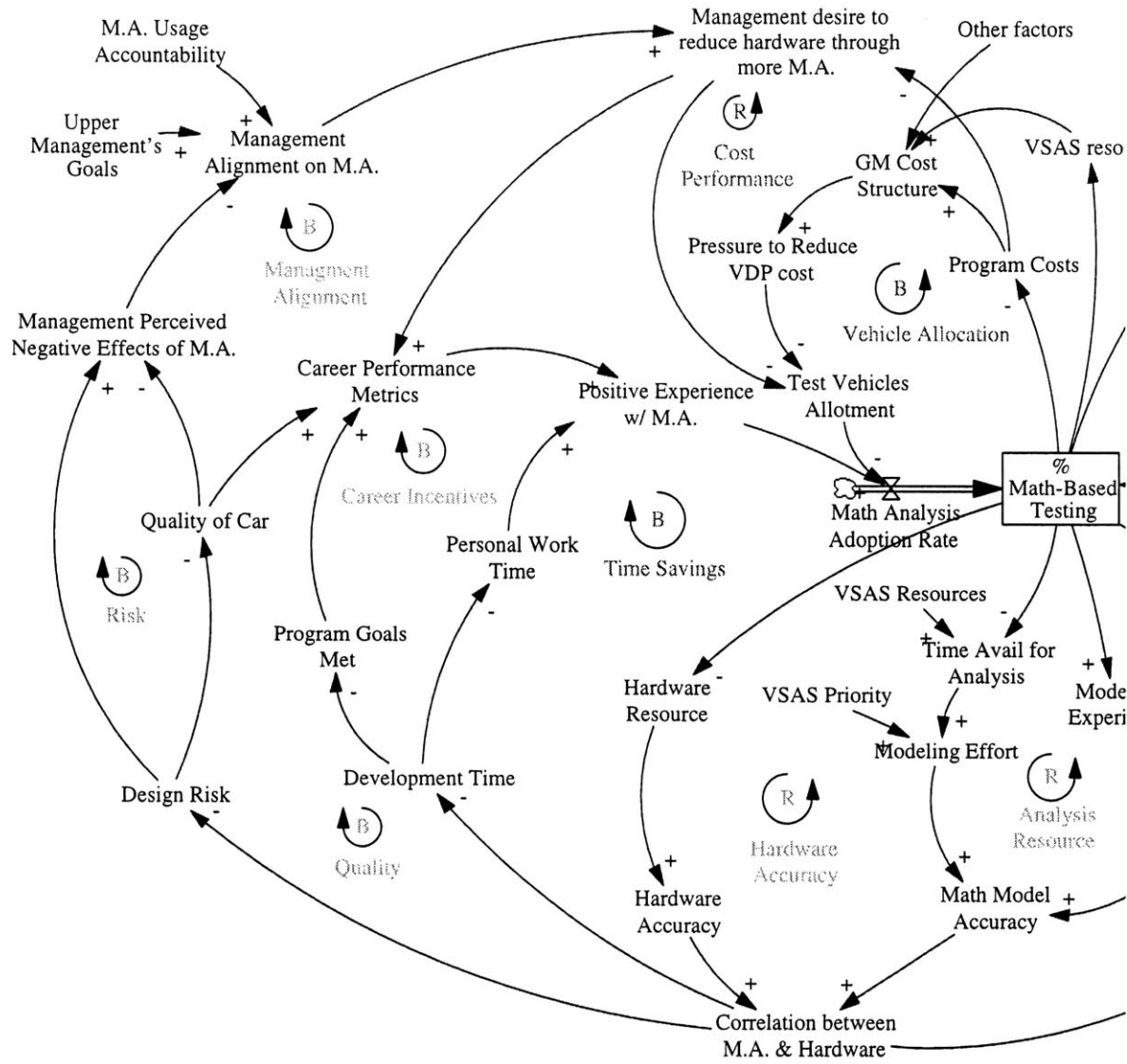


Figure 18. Incentives Loop

Analysis for designing purposes, such as choosing between designs, because this reduces development time.

Since we have already addressed policies to improve correlation, we can investigate other policies such as basing performance goals for both middle management and employees on more math analysis usage, and to reduce risk associated with bad correlation. These policies would introduce exogenous factors that could make the engineers' experience with math analysis more positive.

5.3.6 Systemic Model

When you integrate the loops together it gives you a holistic view of the math analysis usage system. The model can be used to facilitate discussions on how current or future policies, re-organizations, and projects might affect math-based testing. Some of the insights gained from the math analysis model on a systemic level are discussed below:

- **Change Disrupters**

Disrupters of math analysis are caused by the lack of: incentives to use math analysis, math analysis human resources, trust due to sociological reasons, math analysis knowledge transfer, correlation data, and management alignment.

- **Change Agents**

Enablers of math analysis are: growth of math analysis capability, General Motor's increasing sensitivity to cost, and the increasing importance of Analysis in VDP.

- **Direct factors affecting math analysis usage**

Only three factors have a direct effect on math analysis adoption rate: positive experience with math analysis, direct influence on math analysis, and test vehicle allotment constraints. However, it is very difficult to affect these factors directly with policies or exogenous variables; change must be approached through indirect factors.

- **In-direct factors affecting math analysis usage**

Correlation was the root of several of the loops associated with positive experience with math analysis as well as with trust, so special emphasis should be placed on correlation efforts. Cost constraint was another indirect factor that has a large effect on test vehicle allotment and management direction, but increasing cost sensitivity can both help and hinder the goal to increase the percentage of Math-Based testing. Budget cuts would reduce the test vehicles allotted, forcing testing to be done mathematically. However, if the budget cuts also impact Analysis resources, math modeling accuracy and hardware accuracy, correlation would suffer.

5.4 Policy Implementation Opportunity

After interviewing key decision makers, analyzing math analysis in the GM VDP, and creating the system dynamics model, the following observations might help to facilitate math analysis acceptance and alter the organizational culture:

Management at all levels needs to be aligned on the direction they want to guide math analysis development. When engineers get conflicting agendas from different managers, the chances are good they will also have conflicting perceptions of math analysis. This alignment can be measured with small achievable goals such as having a certain percentage of tests, within reason, for a vehicle program done with math analysis. This would signal to the employees that GM is unified on moving to math.

Creating more incentives would be a catalyst to math analysis acceptance. These incentives should focus on creating a positive experience with math analysis and also building more experience with math analysis. As the experience base grows, so will people's willingness to accept and trust math analysis.

Metrics should also be used to measure and provide feedback on increasing math analysis usage as well as correlation. A policy is only good if it works, and the only way to see if it works is to

measure it. Feedback also allows us to improve and communicate policy improvements. Examples of metrics on math analysis usage would be what percentage of a given program is Math-Based Testing. Example of correlation metrics might be what percentage of math analysis models built in a given year show positive correlation.

Although many find it difficult to discuss the cultural differences among different work groups, most people acknowledge that this disparity exists, and that it can be a huge disrupter of change. Bridging this gap can be done with re-organizing, cross training, and building trust. However, cultural shifts can take time to eliminate, causing delays in the systems. Trust in math analysis can be built by both improving correlation, cultural dynamics, and building the experience base. The difficult part is seeding the math analysis experience base for growth.

Finally, to facilitate math analysis adoption the positive loops need to be strengthened, and the negative loops weakened. This can usually be done by introducing exogenous variables or policies into the system. This concept can help strengthen enablers and turn disrupters into change agents.

5.5 Project Extensions

5.5.1 Identify key policies to increase math analysis usage

Using the math analysis usage model as a guideline, the next step would be further analysis of the model to identify additional variables and key policies or projects that would have the greatest impact on increasing math analysis usage.

5.5.2 Modify into decision-making tool by adding simulation to model

By adding quantitative/qualitative relationships to the variables in the model it is possible to run a simulation. Simulations of the model allow us to change certain variables to determine the impact on the stock, and these scenarios would be used to evaluate certain decisions.

5.5.3 Create system dynamics model using hardware count or cost as the stock

System dynamics can be used to model the growth of math analysis usage, as well as to model hardware cost or hardware count. Following the same procedures, we can determine what factors play a role in affecting hardware count or cost, and consequently these factors can be modeled using system dynamics. The model would serve as an excellent visual tool to understand how the factors interact. In addition the model would be easier to quantify and simulate. This would aid in decision-making.

5.6 Socio-technical Development Summary

To successfully sustain change, emphasis should be placed on the need to integrate both technical and socio-technical factors. However, people must be taken into account to sustain change because, according to research [Russell Hanson, “Managing the Human Aspects of Implementation”, SME, 1987.] the socio-technical factors are the stronger of the two forces. The technology is easy to change, but people are not because they are all unique in their perceptions and actions.

Policies can be made to change the system, but there will always be some form of policy resistance. Therefore change induced by this method is difficult to sustain. The most permanent way to sustain change is to start at the grass-roots level. Policies should be directed towards altering cultures and attitudes. Creating strong incentives for engineers to want to use math analysis permeates into the culture, and math analysis has a better chance of becoming deep-rooted in the organization.

The math analysis Adoption model was created to help illustrate the major factors involved in affecting math analysis usage. In its current form, simulations or scenarios cannot be performed because no quantifiable relationships have been established between the different variables. The model was not intended to be a decision-making tool, but to provoke thought on how the different factors affect one another. The model is just the beginning of understanding and can be further refined as the organization shifts and new norms appear.

6.0 Conclusions

By combining the learning from the both the technology development and the socio-technical development the necessary steps to maximize the migration for the vehicle product development group to increased math analysis usage and ultimately reduction of hardware costs. These key steps are summarized below.

6.1 Create system dynamics model of hardware cost

Recognizing that increased math analysis usage may be one of several indirect factors leading to reduction in hardware cost, we have to ask ourselves, is increased math analysis what we are truly after? Perhaps a more appropriate use of system dynamics might be to model hardware cost and what factors are compounding it. A system dynamics model would allow an increased understanding of what is driving hardware both on direct and indirect levels and also capture the subjective as well as objective factors. A system dynamics model can also be used as a decision-making tool to evaluate the value of different projects or scenarios.

6.2 Develop heuristic optimization program

As discussed before, the next logical extension to the Solver model would be to develop a heuristic optimization program. This model would be integrated with Rainier and could be used as a decision-making support tool. This model would have better results and performance.

6.3 Improve capability matrix

Capability matrix will be critical to defining feasibility of any of the model's solution to reducing hardware. There is a great need to capture the incremental benefits of the math analysis' current capability. The following should be the focus of improvement

- Develop definitive metrics
- Complete centralized data source in useable format (Excel)
- Create user guide documenting the vision, scope, background, usage, definition, assumptions, issues, and contacts.

6.4 Increase efforts to improve correlation

This should be a major focus to address the socio-technical development. Correlation feeds into many loops, such as incentives, quality, and trust. So without good correlation data, it will be very difficult to enforce engineers to use math analysis. Currently, correlation is improving, but at a very slow pace, causing math analysis adoption to also grow at a very slow rate. Improving correlation as well as analysis retention will help accelerate the move to math. Several things can be done to improve correlation: making correlation a priority for analysis engineers, increasing Analysis resource, and creating incentives for analysis engineers to stay at their job functions.

6.5 Integrate heuristic optimization program with capability

Once the mechanisms to address the socio-technical issues are in place, the next step is to stimulate the math analysis stock. By creating a policy that forces people to use math analysis, the percentage of Math-based testing will increase, and people will become accustomed to using math analysis. According to the system dynamics model, this can be accomplished by limiting the hardware allocated for a vehicle program, forcing people to turn to math analysis to accomplish requirement validation.

However, this policy needs to be married with technology development, in this case the heuristic optimization program integrated with the capability matrix. This tool would not only allow us to stimulate the math analysis stock, but it will also be a valuable decision-making tool that can be used to dictate/suggest math analysis usage in current vehicle program.

6.6 Create goals to limit hardware allocation

As discussed above, this policy needs to be a major focus once incentives are in place for math analysis usage. A policy to limit hardware allocation will signal that GM is committed, and that GM management is aligned to migrate to math analysis. This policy should also incorporate some sort of feedback system to actively measure the effects of the policy on math analysis usage.

Although most people are aware cultural issues exist, rarely is it given the same emphasis as technology development. One of the main conclusions of this project was the need to address both technical and socio-technical aspects of using math analysis. One without the other would hinder the move to math.

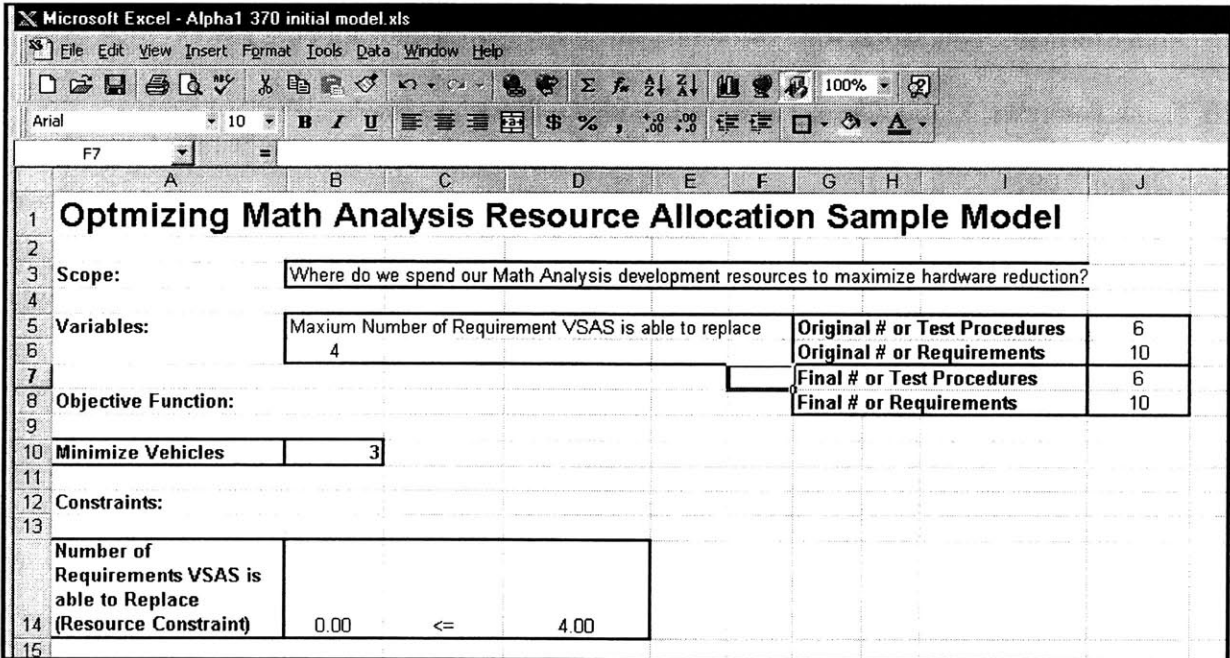
Bibliography

1. Derek Bunn, "Analysis for Optimal Decisions", John Wiley & Sons, 1982.
2. Edward Bursk, John Chapman, "New Decision-Making tools for Managers", 1963.
3. Wayne L. Winston, "Introduction to Mathematical Programming", Second Edition, 1995.
4. Geoffrey Vinning, "Statistical Methods for Engineers", 1998.
5. Thomas H. Davenport, "Process Innovation: Reengineering Work Through Information Technology", 1993.
6. Peter Senge, "The Dance of Change; The Challenges to Sustaining Momentum in Learning Organizations", 1999.
7. Raymond Manganelli, Mark Klein, "The Reengineering Handbook, A Step-by-Step Guide to Business Transformation", 1994.
8. Chris Argyris, "Knowledge for Action, a Guide to Overcoming Barriers to Organizational Change", 1993.
9. Edgar H. Schein, "Organizational Culture and Leadership", 1987.
10. Howard Eisner, "Reengineering Yourself and Your Company", 2000.
11. Shoji Shiba, Alan Graham, Dave Walden, "A new American TQM Four Practical Revolutions in Management", 1993.
12. Ancona D, Kochan T, Scully, M., Van Maanen, J., & Westney, D. E, "Organizational Behavior and Processes", South-Western College Publishing, 1999.
13. David Carr, Henry J. Johansson, "Best Practices in Reengineering", 1995.
14. Jerry Yorman Wind, Jeremy Main, "Driving Change, How the Best Companies are Preparing for the 21st Century", 1998.
15. Will McWhinney, "Creating Paths of Change", 2nd Edition, 1997.
16. Joe Tidd, Hohn Bessant, Keith Pavitt, "Managing Innovation, Integrating Technological Market and Organizational Change", 1997.
17. Karl Sabbagh, "21st Century Jet", 1996.
18. Gary Klein, Judith Orasanu, Roberta Calderwood, and Caroline Zsombok, "Decision Making in Action", 1993.

19. Katsuya Amako, "Object Modeling Techniques", 1995.
20. Russell L. Ackoff, "Scientific Method- Optimizing Applied Research Decisions", 1962.
21. Pamela L. Alreck, Robert B. Settle, "The Survey Research Handbook", Second Edition, 1997.
22. Herbert H. Hyman, "Secondary Analysis of Sample Surveys: Principles, Procedures, and Potentialities", 1972.
23. Ciriani Leachman, "Optimization in Industry- Mathematical Programming and Modeling Techniques in Practice", 1993.
24. Ciriani Leachman, "Optimization in Industry- Mathematical Programming and Modeling Techniques in Practice 2", 1994.
25. John Sterman, John Morecroft, "Modeling for Learning Organizations", 1994.
26. Frank Andrews, Laura Klem, Terrence Davidson, Patrick O'Malley, Willard Rodgers, "A Guide for Selecting Statistical Techniques for Analyzing Social Science Data", Second Edition, 1981.
27. Susan Owens, "Rainier: Environment for Vehicle Resource Optimization", 2000.
28. Susan Owens, "Analysis Replacing Hardware", 2000.
29. "Development of ADV Plan: Process Analysis and Improvement", General Motors R& D Contract Report, 1996.
30. Paola Nisonger, "ADV Process Overview", General Motors, 1997.

Appendix A: Sample Model

A small-scale sample model was initially created to illustrate the modeling concept, and to help the user create future resource allocation optimization models. Creating the model has not been automated, so for the time being the user has to manually build the model. The full formulation of the model is outlined in section 4.0. Solver runs through which combinations of requirements to turn “on” or “off” using heuristic algorithms, to find a near-optimal combination that minimizes hardware. In the figure below we can see the first part of the model contains the variables you need to enter (cell B6: requirement constraint), the objective function (cell B10), the requirements “turned off” (cell B14), and displays the original and final number of test procedures (cell J5, J7) and the original and final number of requirements (cell J6, J8) in the model.



The second part of the model puts the ADV data (relationship between requirement and test procedures), and Rainier output data (relationship between test procedures and vehicles) into a matrix format. This is shown in the next Figure.

The screenshot shows an Excel spreadsheet with two tables. The first table, titled 'Mapping Requirements to Test Procedures from ADV plan', maps 10 requirements (R1-R10) to 6 test procedures (TP1-TP6). The second table, titled 'Test Procedure Vehicle Configuration from Everest', maps 3 prototype vehicles (V1-V3) to the same 6 test procedures.

		Test Procedures					
		TP1	TP2	TP3	TP4	TP5	TP6
Requirements	R1	1	1	0	0	0	0
	R2	1	0	0	0	0	1
	R3	0	1	0	0	0	0
	R4	0	0	1	0	0	1
	R5	0	0	1	0	0	0
	R6	0	0	0	1	0	0
	R7	0	0	0	1	0	1
	R8	0	0	0	1	1	0
	R9	0	0	0	0	1	0
	R10	0	0	0	0	1	1

		Test Procedures					
		TP1	TP2	TP3	TP4	TP5	TP6
Prototype Vehicle	V1	1	1	0	0	0	0
	V2	0	0	1	1	0	0
	V3	0	0	0	0	1	1

The first matrix was created by extracting the requirement and test procedure data from the ADV plan and using the “pivot table” function. The matrix was then copied into the model. The same procedure was used to create the matrix for the test procedure and vehicles from the Rainier output.

The final part of the model can be seen in the figure below, and contains the decision variables and forcing constraints. The yellow highlighted cells indicate that Solver will change these boxes to arrive at a solution.

Microsoft Excel - Alpha1 370 initial model.xls

File Edit View Insert Format Tools Data Window Help

100%

Arial 10

F7

15

16 Decision Variables:

17

Requirement Decision Variables		Output Matrix Requirements	Test Procedures					
			TP1	TP2	TP3	TP4	TP5	TP6
R1	1	R1	1	1	0	0	0	0
R2	1	R2	1	0	0	0	0	1
R3	1	R3	0	1	0	0	0	0
R4	1	R4	0	0	1	0	0	1
R5	1	R5	0	0	1	0	0	0
R6	1	R6	0	0	0	1	0	0
R7	1	R7	0	0	0	1	0	1
R8	1	R8	0	0	0	1	1	0
R9	1	R9	0	0	0	0	1	0
R10	1	R10	0	0	0	0	1	1
Requirement Count		Total Requirement	2	2	2	3	3	4
		Test Procedure Needed?	1	1	1	1	1	1
		Forcing Constraints	8	8	8	7	7	6
			>=	>=	>=	>=	>=	>=
			0	0	0	0	0	0

30

31

32

33

34

35

36

Output Matrix	Test Procedures						Total Test Procedures	Vehicle Needed?	Forcing Constraints	
Prototype Vehicles	TP1	TP2	TP3	TP4	TP5	TP6				
V1	1	1	0	0	0	0	2	1	4	>= 0
V2	0	0	1	1	0	0	2	1	4	>= 0
V3	0	0	0	0	1	1	2	1	4	>= 0

37

38

39

40

41

42

ADY Data / ADY Matrix / Everest Data / Everest Matrix / Sample (0) / Sample (1) /

Draw AutoShapes

Ready NUM

Model Output (Decision Variables)

Because of large number of decision variables and constraints, the Premium Solver upgrade needs to be used. Premium Solver handles up to 10,000 variables, however when the Q Alpha one model was created it only required 300 decision variables. More robust testing could not be performed because of data availability and project scope issues.

Appendix B: Qualitative Survey

Goal:

Unbiased understanding of organizations, motivations, and factors affecting key stakeholders in the decision to use math analysis testing in lieu of Hardware testing

Math Analysis Survey Plan:

1. Pre-pre-survey meeting with Analysis Engineer:
2. Pre-survey meeting: Meet with Q program analysis, DRE, & Test Engineers
Goal: Identify driving factors affecting culture and organization.
3. E-mail Mass survey: Email to individuals involved with Q program.
Goal: Quantify factors, find correlation between factors.
4. Use Data to model causal loop analysis.

Survey Topics:

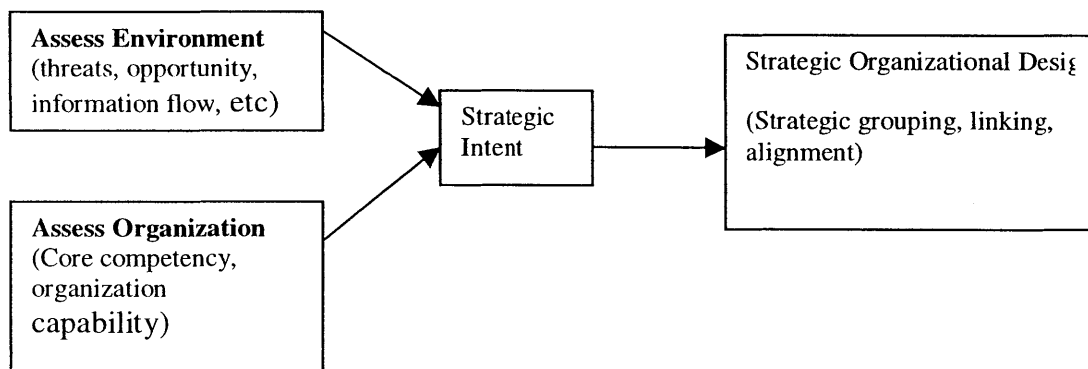
This section is meant to collect ideas, gain insight, share problems, and identify opportunities for improvement of math analysis or Computer Aided Engineering (CAE) in the VDP. Please answer the following questions as you best can. Also feel free to add additional comments or suggestions for areas we haven't listed.

The Three Lenses

Most people already have a schema or pre-conceived notion about a problem or issue, however it is only when we expand our perception that we are able to understand and uncover the entire issue. The three lenses puts us into alternate mind frames designed to expand our perceptions. The most difficult part in using the three lenses is the act of abandoning our personal schemas.

Strategic Lens

Analysis of a problem with a perspective look at how the flow of tasks and information is designed, how people are sorted into roles, and how these roles are related, and how the organization can be rationally optimized to achieve its goals. Strategic lens involves the ability to organize an organization and manage.



- What are the organizations involved with the usage math analysis?
- What is the strategy, goals, or mandate of the organizations involved?
- How are the organizations organized?
- How is the Product development team formed? What is the grouping schema (grouped by activity, output, customer, geography, etc.)

- How do the groups interact? (pooled, sequential, or reciprocal)
- How are the groups within the project rewarded? What are the incentives?
- How might the strategic intent of each organization be misaligned?
- Who are the current customers, who are the users, who are the current suppliers of CAE?
- What is the typical analysis functions during the different project phases?
- What could be done strategically to increase Math based testing to reduce testing hardware?
- Additional comments on Strategic lens:

HOW ORGANIZATIONS AND GROUPS INVOLVED WITH MATH ANALYSIS ARE STRUCTURED

ANALYTICAL COMMUNITY

Goal:

DESIGN COMMUNITY

Goal:

TEST COMMUNITY

Goal:

Goal:

Political Lens

Analysis of a problem with a perspective look at how power and influence are distributed and wielded, how multiple stakeholders express their different preferences and get involved or excluded from decisions, and how conflicts can be resolved.

- Who are the stakeholders or key decision-makers?
- What are their goals? What drives these goals? Is this consistent or does it wane depending on context? How malleable are their interests?
- How would you map the relationships among the different stakeholders?
- How much power does each stakeholder have?
- Where does their power come from (resources, technical skills, special circumstances, position, perception, etc)?
- What are the influences on the stakeholders? Are certain people or groups treated differently?
- Are the needed policies, processes, or systems in place to support conflict resolution?
- How do we get buy in for math analysis to replace a hardware test? What needs to happen? Who makes the final decision?
- What could be done to the politics to increase Math based testing to reduce testing hardware?
- Additional comments on Political lens:

**WHO ARE THE KEY DECISION-MAKERS IN DECIDING WHETHER TO USE MATH ANALYSIS OR HARDWARE,
AND HOW WOULD YOU CHARACTERIZE THEIR INTERACTIONS?**

DRE

Goal:

Influence:

Test Engineer

Goal:

Influence:

Analysis Engineer

Goal:

Influence:

Goal:

Influence:

Cultural Lens

Analysis of a problem with a perspective look at how history has shaped the assumptions and meanings of different people, how certain practices take on special meaningfulness and even become rituals, and how stories and other artifacts shape the feel of an organization.

- Are there cultural differences between the different stakeholders' groups? *Culture is defined as (1) a pattern of basic assumptions, (2) invented or developed by a group as it (3) learns to cope with external adaptation and internal integration, (4) has worked well enough to be valid, and is (5) taught to new members as the (6) correct way to perceive, think and feel in relation to those problems. (Example: OR group is independent)*
- What are the sub-cultures within the groups? (Example: New OR members dress poorly, and veterans are neat dressers)
- What are some of the social norms within the groups? What means is used to enforce these social norms (praise, acceptance, rewards, blame, etc.)? Are they constructive or destructive? (Example: Working 60 hours a week)
- What are some of the factors/cultural issues preventing the extended use math analysis to replace hardware testing in a product development program?
- Hardware testing is also not perfect because of tolerances on prototype parts, non-production parts and processes, etc. What do we currently do to build confidence in hardware testing? Can this be done for analytical testing?
- What are the repercussions of a flawed part design? Where do you feel the blame lies? Is it okay to make design mistakes?
- What could be done to change the culture to increase Math based testing to reduce testing hardware?

Appendix C: Quantitative Survey

Below are the survey questions. Although answers to some questions may be different for different situations, we just want your general opinion. Choose the answer that best fits your opinion and put that number in the answer column, add any clarifying comments you might have in the comment column, and email the survey to quang.nguyen@gm.com when you're finished. Thanks for your time and participation.

Filters
Factors (polarity will determin

	Questions	1	2	3	4	5	Filters
1	In general, do you feel testing should move from mainly hardware-based to more math-based, in order to reduce hardware?	Yes	No				Issue position
2	What is your job function?	Validation Engineer	DRE	Analysis Eng	Development Engineer	Other	Job Position
3	How long have you been in your current job function?	Less than one year	Greater than one year and less than five	Five years and less than ten	Ten or more years		Experience
1	In the very unlikely case of a major design flaw, do you feel there is a personal risk to your career from using Math Analysis to validate a requirement?	No Risk	Low Risk	Some Risk	Medium Risk	High Risk	Risk
2	How important is it to you that GM reduces overall development cost?	Not important	Somewhat important	Important	Pretty Important	Very important	Cost accountability
3	How responsible do you feel about the quality of the system or part design that you are working on?	Not responsible	Somewhat responsible	Responsible	Pretty responsible	Very responsible	Part Design accountability
4	Do you feel responsible for the quality of the overall vehicle design?	Not responsible	Somewhat responsible	Responsible	Pretty responsible	Very responsible	Overall Design accountability
5	Do you feel it requires more effort to try to use more Math Analysis (keeping updated on available Math tool, personel, interest, etc)	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Maintenance
6	Do you feel that using Math Analysis adds to your workload?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Effort

7	Do you think people/politics influence your decisions on whether to use Math-based testing in lieu of hardware testing?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Politics
8	Do you feel the increased use of Math Analysis could decrease your role in Product development?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Job Security
9	How has the quality of the product or design been impacted by Math Analysis ?	Much Worse	Worse	No effect	Better	Much better	Quality
10	Do you feel you've been properly trained about Math Analysis (when to use it, procedures, etc)?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Training
11	Do you feel Management has done a good job leading the development of Math Analysis?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Leadership
12	How would you rate the general Math Analysis Capability at GM compared to industry?	Crude	Under average	Industry average	Better than average	Industry Leaders	Technology
13	What provides you with the most confidence in Math Analysis?	Analysis Engineer's strong recommendation	Analysis Engineer's experience/track record	Past program correlation data	Previous Hardware stage correlation data	Concurrent Hardware Test	Source of Confidence
14	Do you generally feel confident with the results from Math Analysis?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Confidence
15	Are you confident in the people doing the Math Analysis?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	People
16	Do you feel the people performing Math Analysis have good design knowledge and experience?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Analysis Design Experience
17	Do you think the upper level management has a good understanding of the actual Math Analysis Capability/Technology?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Gap between Upper Management and front line

18	In general, do you understand how the Math Analysis tool used to analyze your part/system works?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Knowledge	
19	Do you know the benefits of using Math Analysis? What are they (note in comment box)?	No	Know a few of the benefits	Know some of the benefits	Know most of the benefits	Know all the benefits	Understanding	
20	Do you feel you personally receive any benefit from using Math-based testing in lieu of hardware testing?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Incentive for using MA	
21	Do you know upper management's direction on using Math Analysis to validate requirements?	No Idea	I have a guess	Some idea	Pretty sure I know	Definitely know	Awareness	
22	Do you think all managers act in alignment with GM's direction the role of Math Analysis?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Alignment consensus	
23	Do you feel that using Math Analysis personally saves you development time?	Strongly disagree	Disagree	Neutral	Agree	Strongly agree	Time	
24	How has your feeling towards math analysis been effected by your experience?	NA	Worse	No effect	Better	Much better	Experience	
28	What are some other factors that you feel might affect the usage of Math Analysis?							
29	What do you feel might increase the use of Math-based testing over Hardware testing?							

Appendix D: Math Analysis Usage System Dynamics Model

