2009-04-20

# CAD-Centric Dynamic Workflow Generation

Travis L. Kenworthy
*Brigham Young University - Provo*

# CAD-CENTRIC DYNAMIC WORKFLOW CREATION

by

Travis Kenworthy

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mechanical Engineering

Brigham Young University

August 2009

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Travis Kenworthy

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

| | |
|---|---|
| Date | C. Greg Jensen, Chair |
| Date | Jordan J. Cox |
| Date | Christopher A. Mattson |

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of Travis Kenworthy in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

C. Greg Jensen
Chair, Graduate Committee

Accepted for the Department

Larry L. Howell
Graduate Coordinator

Accepted for the College

Alan R. Parkinson
Dean, Ira A. Fulton College of Engineering and Technology

ABSTRACT

CAD-CENTRIC DYNAMIC WORKFLOW GENERATION

Travis L. Kenworthy

Department of Mechanical Engineering

Master of Science

CAD systems are important design tools that enable the designer to conceptualize, visualize, analyze, and manufacture a design (Shahin 2008). Although high-end CAD systems provide several built-in design applications, the users of CAD often select various custom or proprietary non-CAD analyses that constrain, optimize, or evaluate their designs. An efficient method is needed to perform trade studies from within the CAD environment. Methods have been developed to meet the challenges associated with this need. The methods have been implemented in a program, called the Process Integrator, which resides in a CAD system and allows the user to perform trade studies on an assembly model from within the CAD environment. The Process Integrator allows the user to create a generic process configuration to link analyses with CAD assemblies for optimization. The generic configuration can then be run at any time, on any assembly that meets the configuration requirements. Test cases are presented in which the

efficiency of the automated process is demonstrated. Results indicate that significant time savings can be achieved through the application of these methods.

# ACKNOWLEDGMENTS

I wish to thank my advisor, Dr. C. Greg Jensen, as well as my committee members Dr. Cox and Dr. Mattson for guiding me through the process of researching this topic and putting this Thesis together. I also wish to thank the great people at Pratt & Whitney who had enough faith in me to fund this project. Finally I wish to thank my wife, Rachael for supporting me through this time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# 1 Introduction

The use and automation of CAD (Computer-Aided Design) tools can enable a company to have a competitive advantage and improve market share by shortening the time to market for a product and/or creating improved products (Siddique 2006). A company can be enabled to maintain a competitive advantage through the continual adoption of new, more efficient design methods. Due to the design freedom available in early design phases, new design methods can have the greatest impact in the design process. As the product development process progresses, increased knowledge about the design problem is gained. Unfortunately, the freedom of the designer to make changes is reduced during this time, as demonstrated in Figure 1-1 (Ullman 2010). This creates a challenge for designers to obtain the most information about a design problem as possible during early design phases. The methods presented in this thesis allow for the increase in slope of the knowledge gaining curve to give designers the ability to make educated decisions earlier in the product development process. As seen in Figure 1-2, this increased knowledge improves the ability of the designer to make decisions while there is freedom to make large design changes. This increase in design knowledge in early design phases can be achieved through the reconfiguration of the placement of trade studies in the design process.

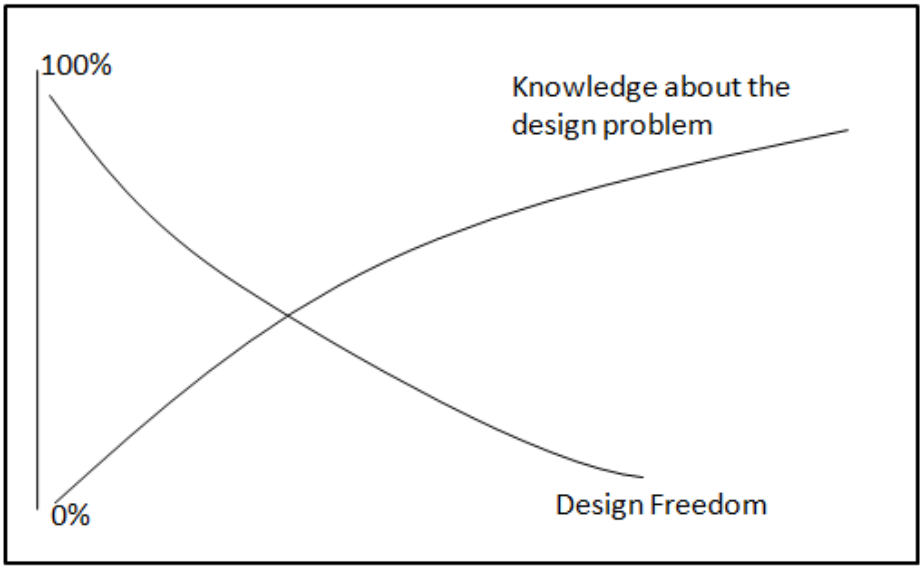


**Figure 1-1 Knowledge and design freedom VS time in the product development process**

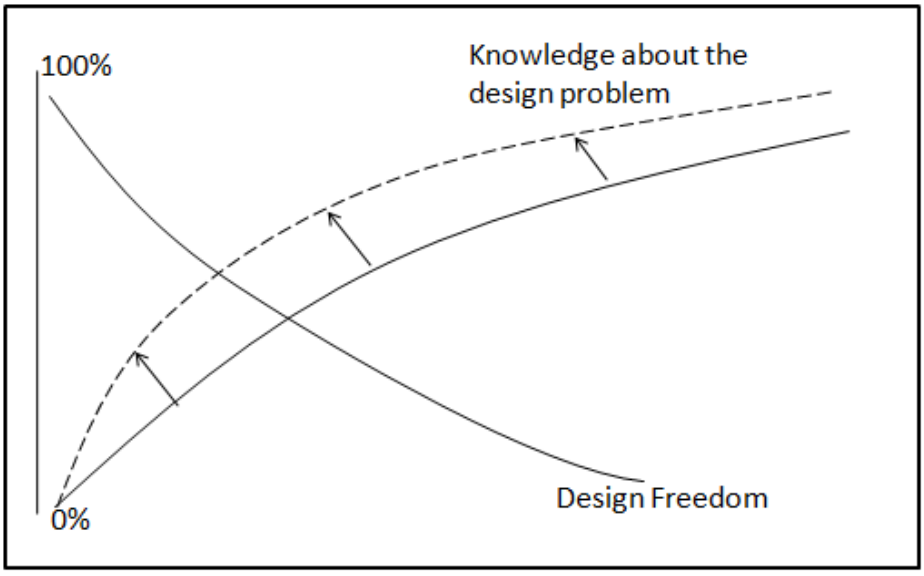


**Figure 1-2 Increased slope of knowledge gaining curve provided by new method**

## 1.1 Problem

CAD models are predominant tools in the product development process. Because CAD models are an embodiment of the product design intent they provide definition for

part documentation, drawings, analysis, and manufacture. Additionally, CAD models are used to make design decisions about the product. This predominant role held by CAD models in the product development process conflicts with the role of CAD during trade studies. Trade studies are often done by performing optimizations which drive slave applications, as seen in Figure 1-3, to perform design iterations (Wind 2008, Wang 2005, Tappeta 1999).

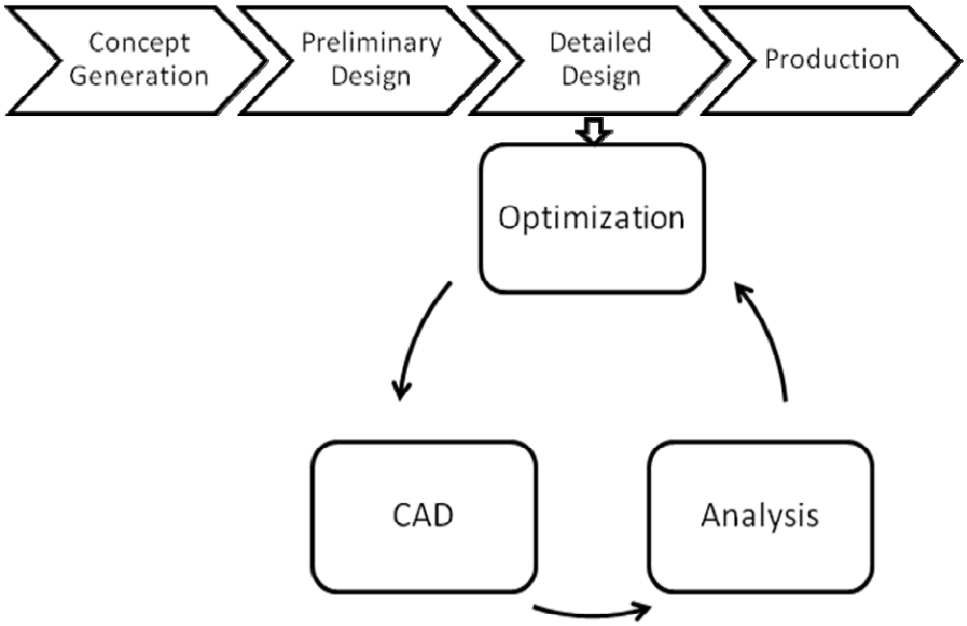


**Figure 1-3 Optimization studies drive slave applications**

Trade studies performed in this configuration remove the engineer from the design loop, leaving design decisions to the whims of the optimization software. This is problematic because optimization software cannot keep up with the variety of ever changing new design problems. This thesis proposes to allow CAD models to retain their predominant role during trade studies and keep the engineer in the design loop by creating a method to perform trade studies from within a CAD environment, as seen in

Figure 1-4. In this new configuration the optimization software becomes a slave of the CAD environment where the engineer maintains control of the design and can determine the best way to move forward with a project.

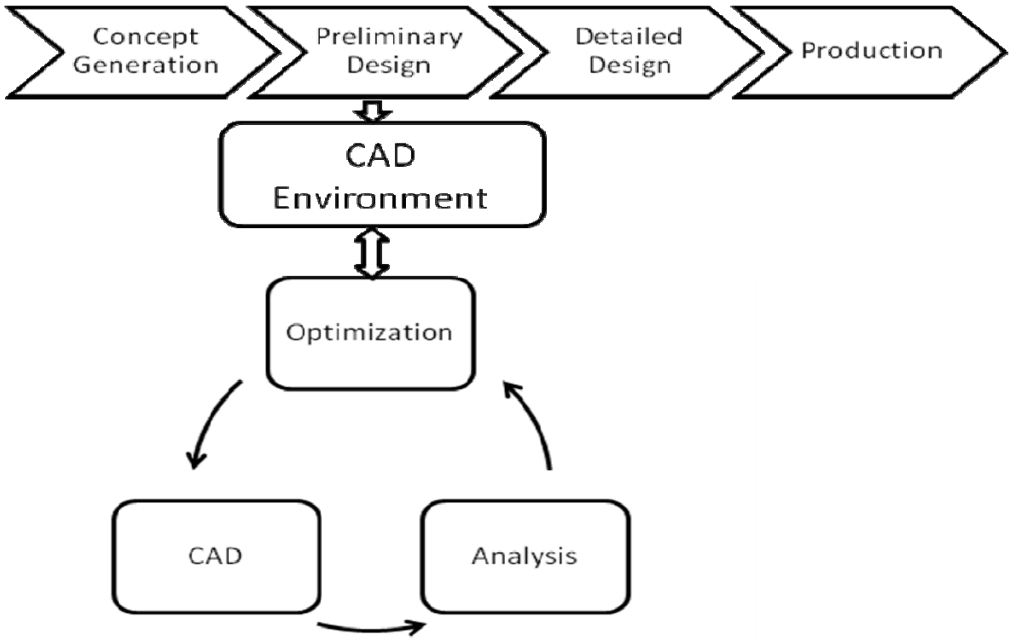


**Figure 1-4 Trade studies launched from CAD environment**

## 1.2 Research Objectives

The purpose of this research is to solve the challenges associated with performing trade studies from within a CAD environment. Making this idea a reality requires answers to the following questions:

1. How can trade studies be defined from within an assembly model?
2. How can recursive trade studies be launched from within an assembly model?
3. How can optimization workflows be automatically created and executed to perform trade studies?

4. How can internal parametrics be linked from the assembly model to the trade studies?

Methods were devised to answer each of these questions. As presented in Chapter three, a CAD-centric approach has been devised to perform trades studies from within an assembly model. The method allows the engineer to remain in the design loop and make design decisions based on information provided by automatically performed trade studies. Chapter four discusses an example of an implementation of the proposed method. The implementation demonstrates the feasibility and benefits of performing trade studies from within the CAD environment. Test cases, discussed in Chapter five, indicate that the new methodology is an efficient method compared with conventional methodologies. The benefits of using the method are particularly apparent when performing trade studies on large part assemblies because of the automated nature of the method. Chapter six presents conclusions that provide answers to the questions pursued by this research and a clear path of future work is proposed.

## 1.3 Delimitation of Project

The methodology produced, although applicable to any CAD environment, is applied to a single CAD system, NX4, which demonstrates the feasibility of performing trade studies from within CAD. For simplicity, the scope of this research is delimited to trade studies that use input and output text files to relay information. Additionally, the methodology has been demonstrated by linking only a single analysis at a time to the assembly model. Linking additional analyses together for a trade study requires only repeating the same functionality as developed for a single analysis. Finally, a single

CAD internal process, a mass properties analysis, has been used to demonstrate how CAD internal analysis can be included in the trade studies. Additional CAD internal processes could also be implemented by repeating the same method.

# 2 Background

The product design process can be enhanced by integrating CAE tools with CAD to aid the design progression from a concept to an engineered product. Leveraging engineering knowledge through the integration of CAE applications with CAD is an effective approach to capturing design intent in a form that can be used to document and manufacture the product.

## 2.1 CAD in Product Development

CAD is essential for efficient product development. CAD systems are used to conceptualize and visualize a design in early design phases and to analyze and refine designs in later phases. CAD models are also used to document and manufacture parts and assemblies after they have been designed (Ulrich 2004). Computer-Aided Technologies (CAx) is a term used to describe modern CAD tools because they incorporate several CAD (Design), CAE (Engineering), and CAM (Manufacturing) methods in a single package.

In the preliminary design stage CAD is used to define the design space. CAD models are very useful for defining the spatial relationships of parts in an assembly. CAD parts and assemblies in the preliminary design stage are often created based on

desired or available size and space. Models in this stage have yet to be analyzed to determine if they meet the engineering specifications of the product.

Zeid (2005) comments that CAD models are of little use unless they are used to analyze or evaluate a design. He states, "The amount of time and effort a designer spends to create a geometric model cannot be justified unless the resulting database is utilized by engineering applications." Zeid is referring to using geometric models to perform engineering design. Norton (2006) defines engineering design as "The process of applying the various techniques and scientific principles for the purpose of defining a device, process, or a system in sufficient detail to permit its realization." Converting a design from basic design space definition to engineered product definition is where engineering techniques and scientific principles are used to produce a viable design. This step, as illustrated in Figure 2-1, leverages engineering knowledge to convert preliminary designs into products that meet their engineering requirements.


**Figure 2-1 Important step in design process**

The final stage of the development of a CAD model is when all the required analyses have been performed and the arrangement, form, and dimensions have been defined, so the model can be called an engineered product (Pahl 2007). As CAD models are refined, several analytical tools are often used.

High-end CAD systems, such as those listed in Table 2-1, provide several built-in design tools that can help a mechanical designer to leverage engineering knowledge to evaluate designs.

**Table 2-1 Common high end CAD systems**

| CAD System | Developer |
|---|---|
| NX (formerly Unigraphics) | Siemens |
| CATIA | Dassult Systems |
| Pro-E | PTC |

Despite the existence of the many CAD internal applications, designers and engineers in industry often use various proprietary, custom non-CAD methods or standalone external analytical processes to perform trade studies. Unfortunately, there is often a disconnect between the part definition contained in CAD and analyses external to CAD. An efficient method is needed to perform trade studies on CAD assemblies (Perlak 2007). This need has been met by creating a generalized interoperability method and example program prototype described in this thesis.

CAD models can be created quickly by creating “rigid” or “dumb” geometry that meets the requirements of the current product, but are not easy to manipulate and are incapable of updating to external changes. Examples of the dimension values for a rigid model are displayed in Table 2-2.

**Table 2-2 Non-parametric beam dimensions**

| |
|---|
| P1 = 2 |
| P2 = 2 |
| P3 = 1 |

A more flexible approach to create CAD models is to use dimensions controlled by parameters. The parameters can then be used to create expressions to control the design (Burgland 2008). Examples of the expression values for a parametric model are displayed in Table 2-3. The relations used in the model enable the user to scale the beam by changing a single parameter, rather than changing all of the individual dimensions. The use of parameters as dimensions is referred to as parametrics and is vital to creating flexible CAD models, and opens the doors for fast design iterations and part optimization (Elliot 2007).

**Table 2-3 Parametric beam expressions**

| |
|---|
| Scale = 1 |
| Length = 2 * Scale |
| Width = Length |
| Height = Length / 2 |

## 2.2 CAE Background

Many engineering parameters, as displayed in Table 2-4, are used to evaluate products. These engineering parameters answer questions about the product and engineers must discover their values for design validation.

After a mechanical designer discovers the values of the desired engineering parameters associated with a design, changes to the design are usually required to assure the design will meet the product requirements. Many types of CAE tools are used to evaluate designs and determine the values of engineering parameters. Often math solvers like MATLAB, MathCAD, Maple, and Microsoft Excel are used to perform this function. Additionally, custom programs that perform calculations are often created in C, C++,

JAVA, FORTRAN, and many other languages. Finally, several meshing and finite element analysis packages are also used to evaluate designs.

**Table 2-4 Common Engineering Parameters**

| | |
|---|---|
| Fluid Flow | Stress |
| Heat Flow | Temperature |
| Reliability | Thermal Expansion |
| Shape | Tolerances |
| Shear | Volume |
| Strain | Weight |

Advancements are always being made to create ways that computers can aid in the design process to create better designs more quickly. The advances in the area of CAD create significant advantages for companies as they can create product families more quickly than competitors and reduce labor costs, etc. The large variety of analyses used by mechanical designers is the reason that a very generic method has been devised for this research to define many different types of analyses to link to the CAD assembly.

## 2.3 CAD Automation

CAD systems are designed to be used by engineers in many fields and contain general methods for performing operations to create designs. Because of the many different uses of CAD in industry, it is impossible for CAD developers to anticipate the needs of every engineering undertaking (Wang 2007). Significant advantages can be achieved by customizing a CAD system to automatically perform commonly performed

functions or create entirely new functionality. Some companies/researchers even create their own CAD environments to provide the desired level of customization to their processes (Blanding 2007). Automated approaches to repetitive CAD tasks are common place in industry and several researchers have recently published work regarding such approaches. Lamarche (2007) took full advantage of the benefits of automating CAD processes. Through automating modeling tasks using a CAD system's API (Application Programming Interface), Lamarche reduced a 320 minute modeling task to a 4.7 minute, fully automated process.

Dye (2007) reports of the development of a gas turbine design tool. The design tool, called the Cross-Section Designer, provides the user with an editable engine gas path and allows the user to manipulate geometry in various design schemes. The Cross-Section Designer is an example of how engineering knowledge can be merged with automation techniques to control a design. The concept of knowledge based automation is critical for creating a competitive advantage by not only automating modeler tasks, but by realizing engineering knowledge in the designs.

Lai (2009) reports another current research project is the creation of a constraint-based system for product design and manufacture. The system allows the designer to create a design "Anchor" that defines how a model should be constrained. Then the user can change aspects of the design, and the rest of the model updates accordingly to maintain the constraint settings. Lai's report is an excellent demonstration of how automation techniques can be used to create custom functionality to meet the specific needs of a company or project.

Danjou (2008) has created an automated CAD method in which UDFs (User Defined Features) are assembled to create a product using an input data file. According to Danjou, one of the unique aspects of this methodology is that the design knowledge is stored in the UDF components, rather than the custom application. This aspect of the method makes the system non-component specific. Danjou's methodology displays a desirable principle when creating automated systems; the more generic the custom application the more wide spread its uses will be, and the less often the program will need to be updated to be current with company business practices and projects. A related concern is that increasing the automation of a process can decrease the amount of future innovation related to that process (Salzman 1989). This problem can be minimized by making custom applications that are generic and adaptable to new innovations.

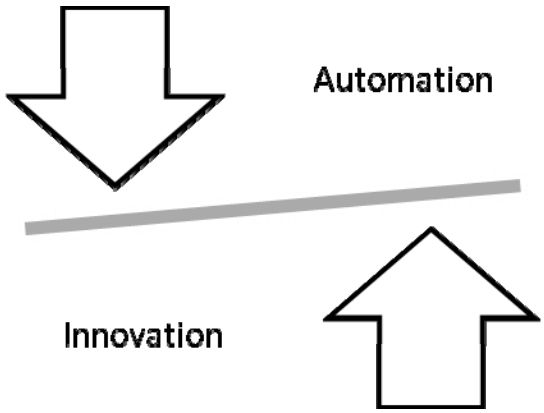


**Figure 2-2 Increased automation can reduce possible future innovations**

## 2.4 CAD-CAE Integration Background

Several researchers have been working on problems related to integrating analytical systems with CAD (Lee 2005). Kosavinta (2007) describes the integration of

CAD with a Decision Support System (DSS). The goal of the DSS is to enable the designer to make real-time decisions about a design. The DSS evaluates the project calculating project feasibility and cost. The DSS evaluation allows the designer to quickly make informed decisions about a project design. Similarly, Shehab (2006) details the integration of a knowledge based system with CAD. The knowledge based system has the capability to help the designer create an optimal assembly based on estimated assembly times and cost as well as providing design improvement suggestions. Additionally, King (2004) utilized the API interfaces of Unigraphics, Hypermesh, and Fluent to create a CAD-Centric approach to CFD analysis. These examples of merging CAE processes with CAD are similar to the goals of this research, in which optimization methods as well as a variety of CAE tools are to be merged with CAD. The integration method performed in these examples created programs specifically to merge CAD packages to the desired analyses. Although they were efficient for executing the specific analyses or applications, it would be repetitive to use this approach for executing many different types of applications. A more generic approach to process integration would enable the designer to interact with many CAD-external applications through a single custom tool.

## 2.5 CAD Optimization

A method to link analytical tools with CAD has been created by Engineous using their optimization software package iSIGHT-FD (Koch, 2002). According to Velden (2007), Engineous has created an automated process to drive geometric changes in CAD models using components in iSIGHT-FD. This methodology presents a flexible approach

to CAD/CAE integration because it allows CAD models to be integrated with any combination of analyses. However, this method has the limitation of requiring the manual configuration of each component to be used. Similar to the automation of repetitive modeling tasks in CAD systems, an automated method to create and configure optimization workflows can provide a great time saving advantage. Additionally, an advantage of automating the task of creating optimization workflows for CAD models is the possibility of expanding the user base for using engineering analysis and optimization tools. The automated nature of the new approach results in the need for very little training to enable modelers to utilize a whole host of CAE tools (Velden 2002).

An enhancement to the currently available CAD optimization methods would be to create a program that can be launched from within a CAD system, automatically create iSIGHT-FD optimization workflows, run optimization iterations, and use the results to instantiate the optimal CAD model. The benefit of this enhancement is the increased amount of automation in the process, saving time in both set-up and execution as well as minimizing the needed training of the user to use optimization and analysis software.

## 2.6 Java Application Development

Java is a versatile programming language that has been used for the implementation of all the methods developed by this research. It can be used to create graphical interfaces to interact with the user, interact with CAD systems, and interact with optimization software. Java programming is very similar to C++, and if a user has previously used C++ then Java can be picked up very quickly. Information about how to

program using Java can be found online (Eck 2007). Additionally, IDEs (Integrated Development Environments), like Netbeans, can be acquired to aid in Java development.

## 2.7 API Programming

Commercial CAx tools create interfaces for users to access the internal functionality through Application Programming Interfaces (APIs). An API allows the user/developer to perform custom operations and accomplish process specific tasks. Several types of APIs exist for different CAx tools. CAx tools commonly provide one or more APIs for scripting, macro, or object-oriented programming. Generally, an API enables the user to programmatically perform tasks that would normally be performed manually with the interactive software, but provides the many benefits of a programmatic environment:

1. Programmatic calculations
2. Automatic access and use of data files or other sources for information
3. Perform operations with extreme accuracy
4. Perform operations very fast

The objectives of this research are achieved by using the JAVA API for both NX by Siemens and iSIGHT-FD by Engineous. As an example of how the NX JAVA API works the following is an example of how part expressions can be accessed programmatically. First, the custom application must be compiled with a link to the NXOPEN library. Next, the NX session must be obtained. This can be done with the following call from the NXOPEN library:

*nxopen.Session theSession = (nxopen.Session)SessionFactory.get("Session");*

Once the NX session object has been obtained it can be used to get the part collection object that is a list of all the parts in the current session.

*nxopen.PartCollection prtCol = theSession.parts();*

The part collection can be used to obtain any part in the session. The following line obtains the work part, which in NX refers to the active part.

*nxopen.Part workPart = prtCol.work();*

Once the work part object has been obtained it can be used to obtain its expressions by obtaining the expression collection.

*nxopen.ExpressionCollection theExpressions = workPart.expressions();*

Now that the expression collection has been obtained it can be used to find any desired expression in the part and to access or change its value. The following code uses an iterator to step through each expression in the expression collection and check if the name of the expression is equal to "Height". When "Height" is found, the value of the expression is returned from the function.

```
String expName = "Height";
java.util.Iterator expIt = theExpressions.iterator();
    while(expIt.hasNext()){
       nxopen.Expression exp =(nxopen.Expression)expIt.next();
       if(exp.name().equals(expName)){
                    return exp.value();
       }
    }
```

In addition to accessing part expressions, the following are various tasks that are commonly done programmatically using a CAD system's API.

1. Create/open part
2. Geometry creation/edit
3. Expression creation/edit

4. Layer manipulation
5. Part mass properties analysis

## 2.8 iSIGHT-FD Custom Applications

iSIGHT-FD is an optimization package developed by Engineous and is used extensively for the implementation of the methods developed in this research. iSGIHT-FD has a Java API that can be used to make various types of applications including iSIGHT-FD components and iSIGHT-FD applications. The component is a tool that is placed in an iSIGHT-FD workflow and used to perform a task during the execution of the workflow. The application is a program that can build, configure, and execute a workflow. An application development guide located in the iSIGHT-FD install directory gives full instructions and examples of how to create both components and applications as of iSIGHT-FD version 3.0. Earlier versions provide instructions for the creation of components only.

# 3 Methods

Chapter 1 explained how performing trade studies from within a CAD system can improve the design process by allowing the designer to remain in the design loop to make design decisions. This chapter presents methods to overcome the challenges associated with defining and executing trade studies from within the CAD environment.

## 3.1 The New Methods

Research was performed to develop methods to solve each of the challenges presented in Chapter 1. Each of the questions have been answered in the form of software applications that enable the definition of trade studies, the launch of the studies, the automatic generation of optimization workflows, and the linking of internal assembly model parametrics with the study. The combination of these methods, as displayed in Figure 3-1, represents a CAD-centric approach to performing trade studies on assembly models. The following is a brief description of each of these methods.

1. Process Initializer
    a. CAD internal application that presents the user with a graphical interface that allows the user to define and save generic trade study configurations.

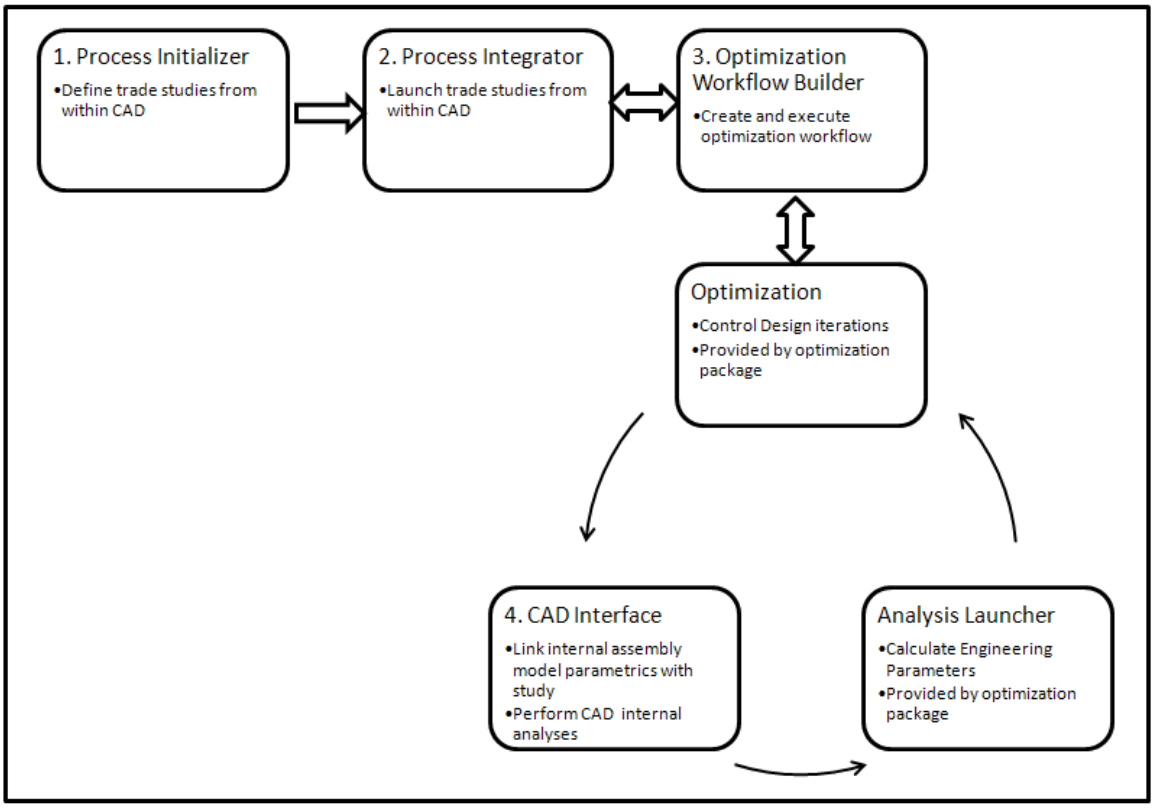


**Figure 3-1 Automated process for performing trade studies from within CAD**

2. Process Integrator
    a. CAD internal application that allows the user to launch pre-configured trade studies on any CAD assembly.
    b. Based on the configuration settings, this method obtains all needed part definition to initialize a part optimization for each applicable part.
    c. Makes final part instantiations based on trade study results.
3. Optimization Workflow Builder
    a. Creates optimization workflows using an optimization package
    b. Executes the optimizations and saves results.
4. CAD Interface

a. Links internal model assembly parametrics with trade study
b. Method creates a connection to a CAD package and uses it programmatically to create model instantiations.
c. Runs CAD internal processes/applications.

The custom methods listed above are described in further detail in the following sections.

### 3.2 Process Initializer

The Process Initializer is the method that has been devised to define trade studies from within the CAD environment. The method is to use a GUI (Graphical User Interface) that is linked to an assembly model in the CAD system. The link to the assembly model is made by using API calls to the CAD package. The GUI presents the user with options to define the information required for trade studies. The API calls are used to recreate the assembly part tree in the GUI. Additionally, API calls are used to obtain the assembly parametrics. This information is provided to the user so they can define what parts from the assembly to use in the study, what part/assembly parametrics to link to the study, what analyses to use in the study, and what parameters to use for optimization design variables, constraints, and objectives. Once defined all of this information is saved in a file and is ready for use when the study is launched.

### 3.3 Process Integrator

The Process Integrator is the method that makes the CAD-centric approach possible. The Process Integrator allows the user to launch external studies from within a

CAD system. This is accomplished by first obtaining all the needed information to create an optimization workflow for a particular study. This is done by reading the configuration file saved by the Process Integrator and using it to obtain relevant information from the current assembly. For example, if the trade study has been defined to optimize all disk parts in an assembly, then a list of the disks in the current assembly would be made along with their relevant expressions to be used in the study. Once all the needed information has been obtained and saved in a file, the Optimization Workflow Builder is launched by using a system call. After the study has been completed, the Process Integrator retrieves the results and presents them to the user as well as using them to make a final instantiation of the assembly.

### 3.4 Optimization Workflow Builder

The Optimization Workflow Builder method creates a workflow in an optimization package to perform a trade study. Although custom optimization algorithms could be developed for this application, commercially available optimization packages are recommended to take advantage of existing technology. Optimization packages also provide a method to launch analyses for the study. The aid of optimization software greatly decreases the complexity and amount of development that needs to be done to perform the desired design iterations. This method includes several steps needed to configure the desired optimizations and map the needed variables between the optimization, CAD, and analysis tools as follows. All of these steps are performed using API calls of the optimization package.

1. Create instances of the needed workflow components (see Figure 5-3)

    a. An instance of a loop component is needed to iterate through the desired optimizations.
    b. An instance of an optimization component is needed for each part to be optimized in the workflow.
    c. An instance of the CAD Interface tool and an Analysis Launcher is needed for each optimization loop.

2. Connect the component instantiations in the desired configuration
    a. The flow of the study is determined by indicating to the optimization package the order in which to run the instantiated components.

3. Configure the desired optimizations
    a. Set design, constraint, and objective variables with appropriate bounds, etc.

2. Configure CAD Interface
    a. Define which part/assembly to open.
    b. Define which expressions in which parts to update.
    c. Define which expressions to extract.
    d. Define which CAD internal process to run, if any.

3. Configure the Analysis Launcher
    a. Define which analysis inputs to update.
    b. Define which analyses to run.
    c. Define which analysis outputs to extract.

4. Map parameters

a. Map design variables from optimization to CAD Interface and Analysis Launcher tools.
b. Map CAD Interface outputs to Analysis Launcher and optimization constraints and/or objectives.
c. Map Analysis Launcher outputs to the optimization constraints and/or objectives.

5. Execute optimizations
6. Save results

### 3.5 CAD Interface

The CAD Interface tool is a method to link trade studies to the internal parametrics of an assembly model. The method is to use both the CAD package API and the optimization package API to access both the parameters of the trade study as well as the parameters of the CAD assembly. The CAD Interface tool allows the optimization workflows to have access to the CAD environment during optimization iterations. The connection to the desired CAD package is made through an API call from the CAD package. This connection is used to perform part updates and data extractions. Additionally, the CAD API can be used to perform internal processes or analyses, such as extracting mass properties from a model. The optimization package API is used to receive data, such as new values of design variables from the optimization, and to send data, such as extracted model parameters, to the trade study.

# 4 Implementation

The methodology presented in Chapter 3 was implemented to demonstrate seamlessly performing trade studies from within the CAD environment. This implementation has been performed in the CAD package NX4, and utilizes the optimization package iSIGHT-FD. The implementation has been called the Process Integrator, and it follows the CAD-centric approach outlined in Chapter 3. The implementation allows a mechanical designer to run analysis iterations on a CAD model or assembly from within a CAD system. The Process Integrator makes it possible for the CAD modeler to very easily update a model to an optimal design without directly interacting with an external analysis tool. The implementation and testing of the method presented by this thesis demonstrates that the devised method is a valid approach to part optimization for the cases studied in this research.

## 4.1 Development

Investigations were performed with NX4 and iSIGHT-FD before developing any custom applications to determine if they could be used in a satisfactory fashion to connect CAD models to analytical packages. It was found that NX4 models can easily be modified through expressions which control their dimensions and shape. Additionally, it was determined that iSIGHT-FD can be used to connect an analysis package with a CAD

model and perform the desired optimization algorithms. It was also determined that a component from the package could be used to launch analyses for trade studies. Finally, it was determined that a custom component to perform the CAD Interface method was needed. These software investigations were helpful for determining the types of functions and terminologies that would be required to use in the APIs of the two systems.

## 4.2 Program Layout and Communication

The implementation consists of the following applications that will be discussed:

- Process Initializer
- Process Integrator
- iSIGHT-FD Launcher
- Optimization Workflow Builder
- NX Component

Communication between these applications is performed using text files that convey the needed information. The Process Initializer creates an initialization file that defines a generic trade study, as demonstrated in Appendix A. The Process Integrator creates a runtime file that defines a specific instance of the generic study for that CAD assembly, as displayed in Appendix B. The Optimization Workflow Builder creates a results file, as seen in Appendix C, which conveys the optimization results to be displayed and instantiated in the CAD model.

## 4.3 Process Initializer Implementation

The Process Initializer is a graphical interface that makes it possible for the user to configure an optimization that will link any CAD model to various types of processes. The user can start the Process Initializer by selecting the "New" button in the Process Integrator as seen in Figure 4-8. The Process Initializer will appear as seen in Figure 4-1.

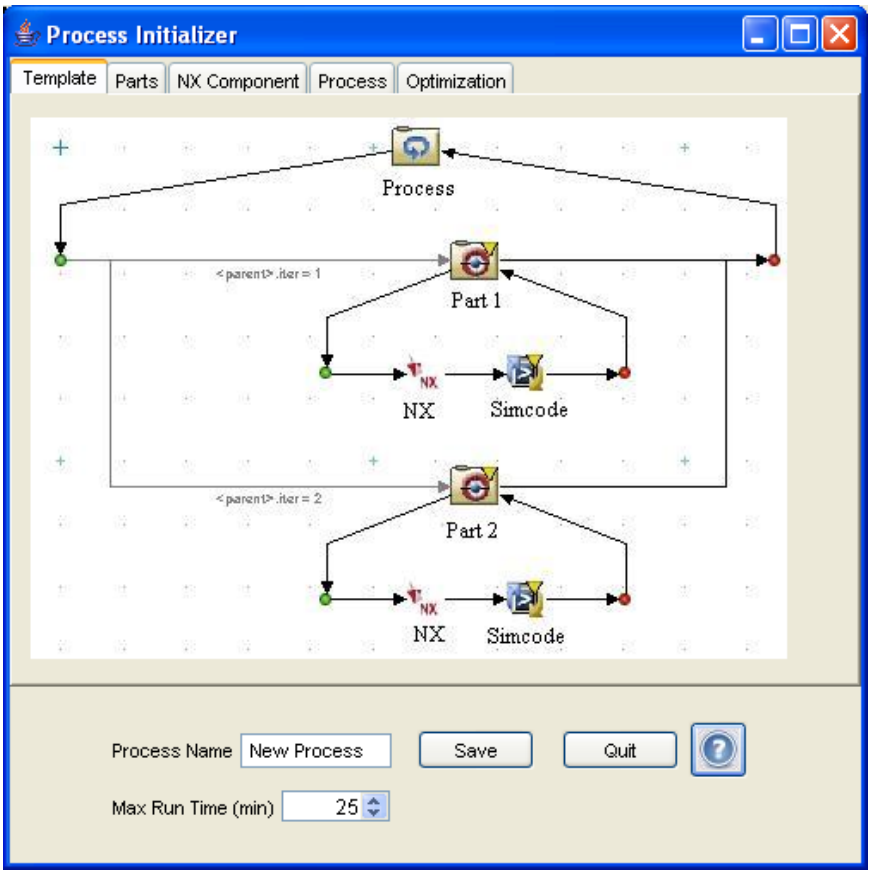


**Figure 4-1 Process Initializer dialog**

### 4.3.1 iSIGHT-FD Template

The first tab of the Process Initializer shows a template of what the iSIGHT-FD workflow will look like. At the top of the workflow is a Loop component that will cycle

through the workflow and optimize each part one at a time. Design iterations are performed on each part using an Optimization component. The design iterations are performed by changing design variables that are passed to the NX component and Simcode components. The NX component will use the new values of the design variables to instantiate a specified CAD model. Then values of expressions and mass properties can be accessed for use by an analysis run by the Simcode component. The Simcode component runs a specified analysis. The subsequent tabs in the Process Initializer allow the user to provide the information needed to configure all of the components in the workflow.

### 4.3.2 Parts Configuration

The Parts configuration tab, as seen in Figure 4-2, allows the user to specify what parts in the assembly to optimize. There are three part configuration types to choose from. The Part List configuration allows the user to choose parts from the Available Parts List and move them to the Selected Parts List. An optimization will then be run on each of the parts in the Selected Parts List. The All Parts configuration obviously allows for every part in the assembly to be optimized. The Subset of Parts configuration allows the user to specify in any or all of the subset text boxes that indicate which parts to optimize. No matter the configuration, at least one part needs to be added to the Selected Parts List for use in configuring the NX Component.

Parts in the assembly can be analyzed individually or as a system of parts depending on how you configure your configuration. Specifying which parts to optimize, defines how many optimization loops will be generated and how part expression names might change with the part to be optimized. If it were desired to optimize a single system

of parts rather than several individual parts, a single part would be selected to be optimized and then the desired expressions from the system of parts would be chosen for extraction and update on the NX Component tab.

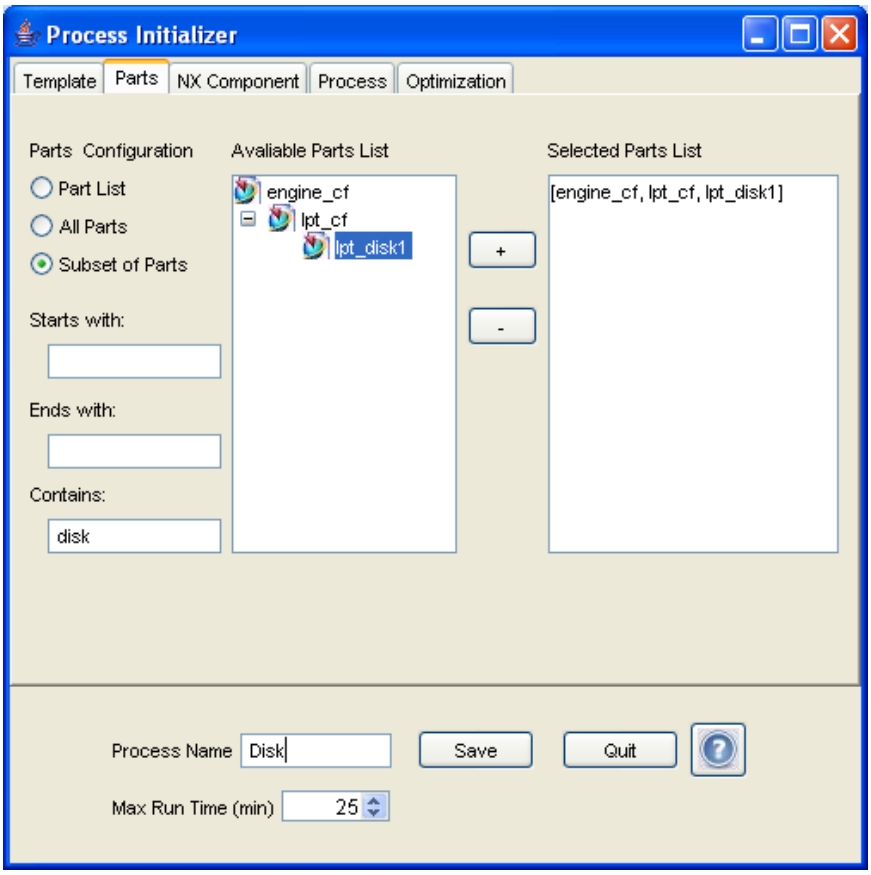


**Figure 4-2 Parts configuration tab**

### 4.3.3 NX Component Configuration

The NX Component tab, as seen in Figure 4-3, allows the user to choose which parts to use expressions and mass properties from in the CAD assembly. The Part Path Tree shows the parts in the path of one of the Selected Parts from the Parts tab. The expressions of each of the parts in the part path can be accessed by clicking on the

desired part in the tree. Then the expressions pertaining to that part will appear in the Expressions list. Once the Expressions appear in the list they can be chosen to be used as design variables by selecting the desired variable and then pressing the plus button next to the Update list. The expressions can also be chosen to be extracted from the part after the part has been updated by adding them to the Extract list. Mass Properties of the parts can be added to the Extract list, but not the Update list. The Starts with check box allows the user to specify whether the expression name will change based on a substring of the part name. There are currently three configurations that define how the expression name could change in relation to the part name.

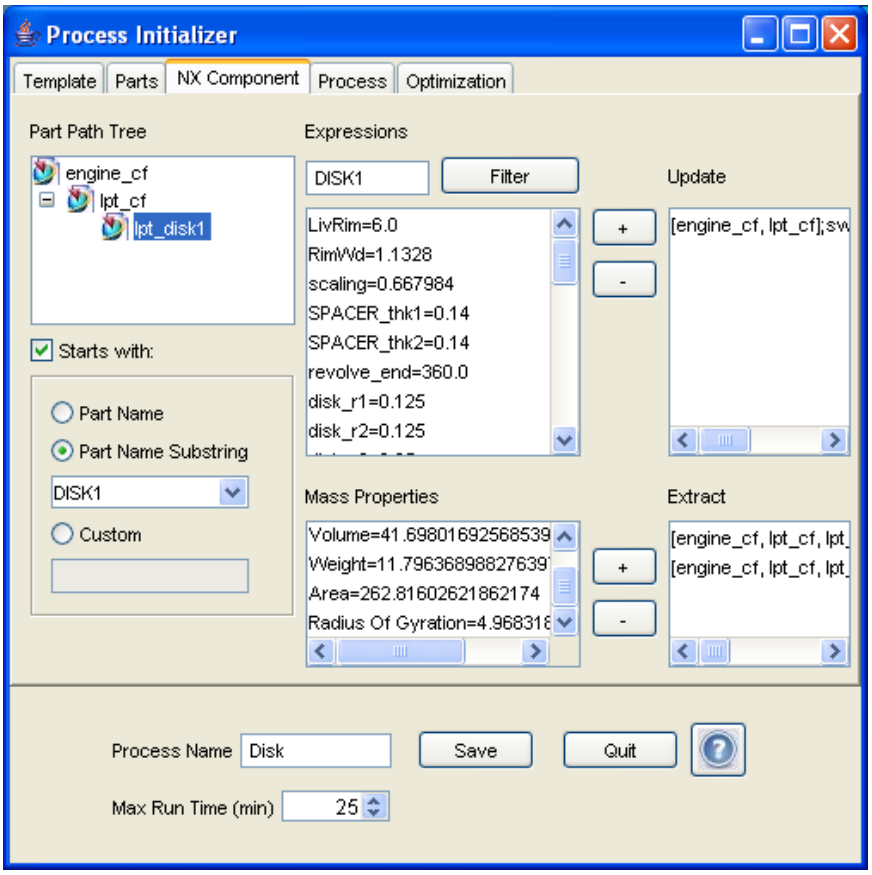


**Figure 4-3 The NX Component tab is used to define what expressions to update and extract**

### 4.3.4 Process Configuration

The Process configuration tab, as seen in Figure 4-4, is used to define what analysis to link to the CAD model. The most important input required to configure the analysis is the command line entry that will be used to launch the analysis. The browse button next to the Command Line text box can be used to browse to the location of an executable file and then other command line arguments can be added to the text box. The analyses that can be set up by this process are limited to those that read and write text files for inputs and outputs. The locations of the text files to be used by the analysis can be specified by typing in the text box for each file, or clicking on the browse button for each file and using the file browser to identify the file. An output template file is also needed so that iSIGHT-FD can know what the analysis output file will look like. The Name=Value drop-down menu for each file is used to specify how to split each line of the file. The files should be created to show a parameter name followed by a symbol like the equals sign and then the value of the parameter. The drop-down menu allows the user to pick '=', ' ', '/', and ':'.

Once all the process input and output files and the Name=Value parsing characters have been specified, the analysis parameters are ready to be mapped. Parameters from the NX component as well as design variables from the Optimization will be mapped to the Simcode component. The process outputs will be mapped to the Optimization for use as design constraints and objectives. The parameter mapping is configured by pressing the Parse button as seen in Figure 4-4.

The File Parser dialog, as seen in Figure 4-5, has three tabs. The Input tab is used to specify which parameters will be updated in the input file. The Output tab is used to

specify which parameters will be extracted from the output file. The third tab, called Map, is used to specify which NX parameters to map to input parameters.

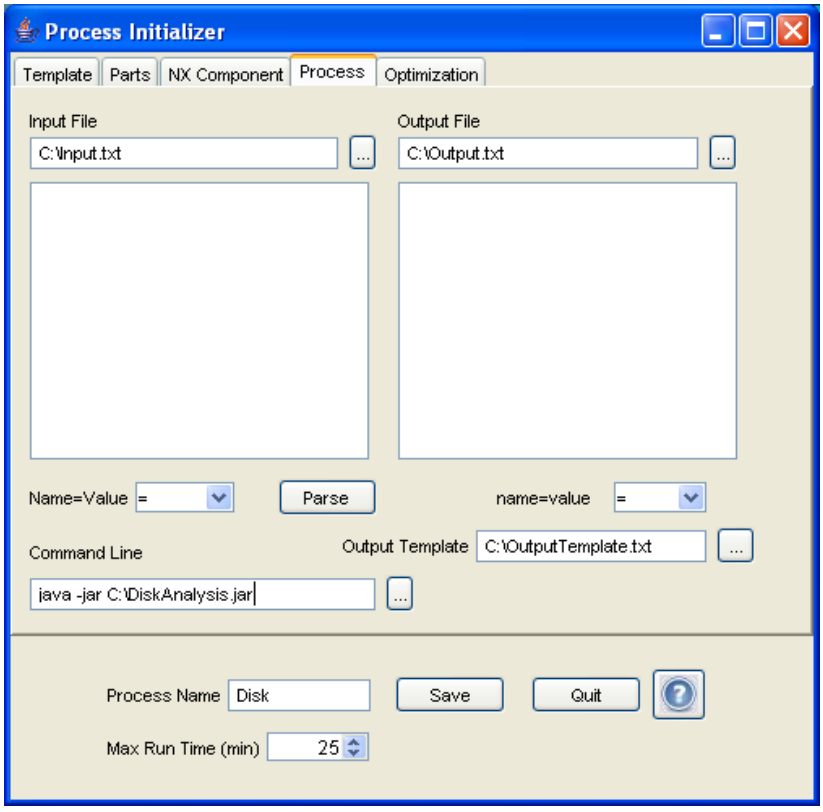


**Figure 4-4 The Process tab is used to define the analysis to be linked with CAD models**

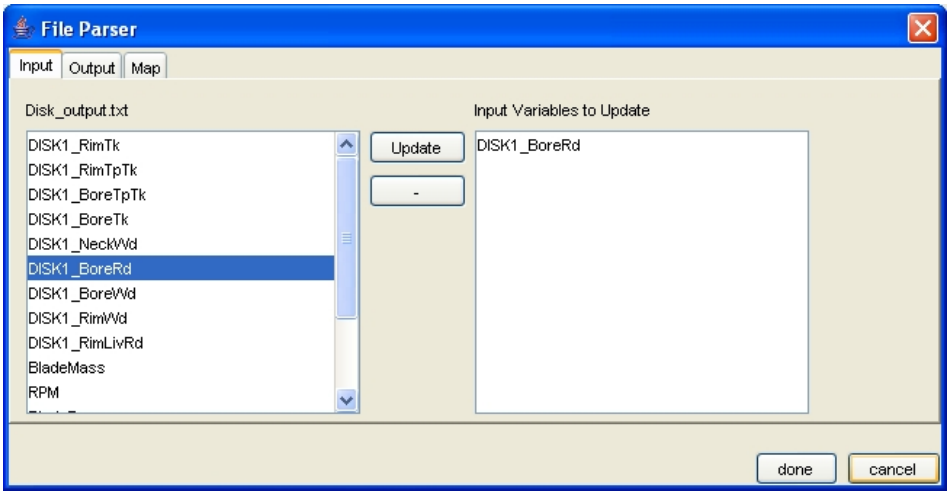


**Figure 4-5 File Parser dialog**

### 4.3.5 Optimization Configuration

After the parameters to be updated and extracted from the NX model and the analysis have been specified, they can be used to configure an optimization. The Optimization tab, as seen in Figure 4-6, allows the user to specify design variables, constraints, and objectives.

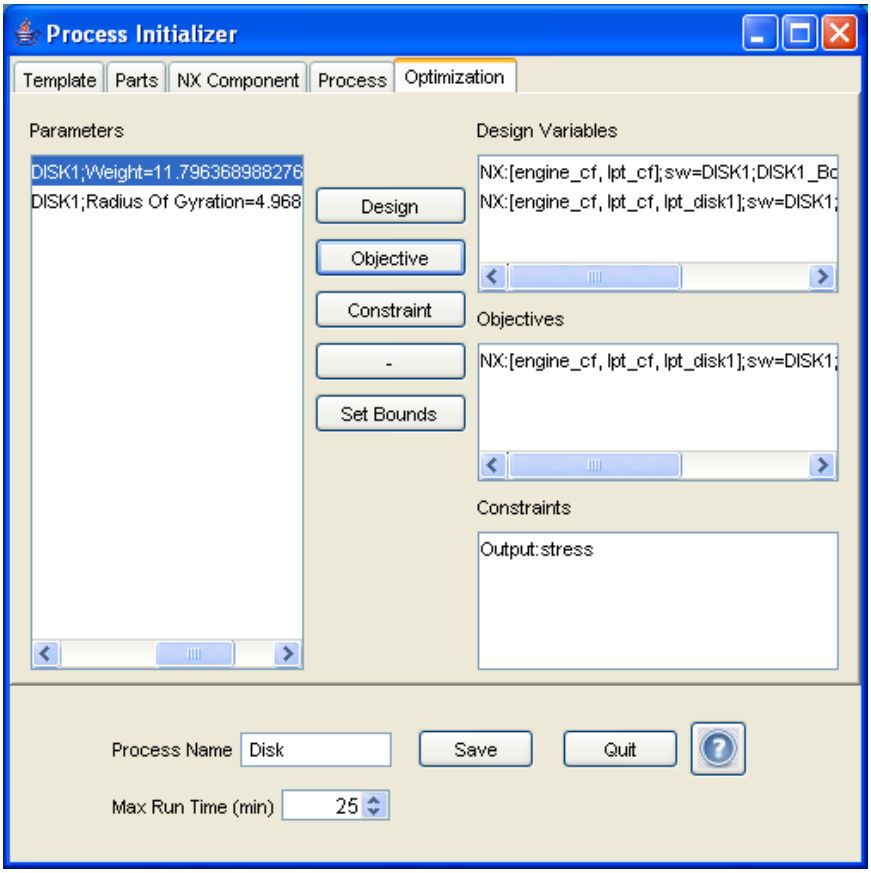


**Figure 4-6 Optimization tab**

After specifying the necessary optimization parameters the bounds and optimization direction for the variables must be set. The Set Bounds button provides this

functionality by creating a dialog where the user can specify that information as seen in Figure 4-7.

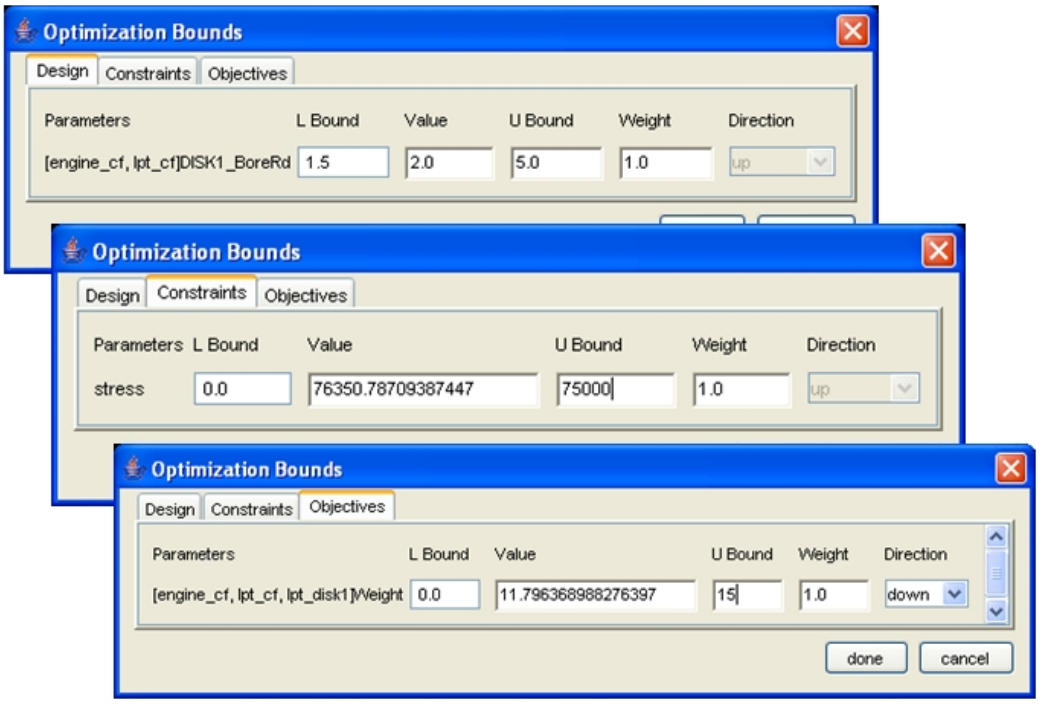


**Figure 4-7 Optimization Bounds dialog**

Once a process has been configured in the Process Initializer the user can specify a descriptive name to identify the process. As seen in Figure 4-6, the Process Name text field provides a place for the user to specify the process name. Then the configuration can be saved. When Save is pressed, the configuration is saved in a text file that contains all of the specified settings and the name of the process as displayed in Appendix A. The process name is also used to populate the drop-down menu in the Process Integrator.

## 4.4 Process Integrator Implementation

The Process Integrator resides as a toolbar in Siemens NX4. The program is designed for the user to run with a part/assembly open that they would like to use in a trade study. When the button on the toolbar is pressed to run the program, a dialog appears as seen in Figure 4-8.

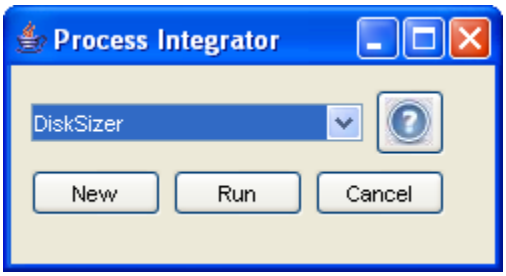


**Figure 4-8 Process Integrator**

The Process Integrator presents the user with a list of configured processes that they can choose to run. When the user runs one, an initialization file for that process is examined. Next, the current CAD assembly is examined to see if the study applies to any of the current parts. Then, all of the parts that the study applies to are saved, along with current values of their expressions.

Once all the required data has been acquired and saved for later use, the Process Integrator uses a system call to run the iSIGHT-FD Launcher. The Process Integrator then waits for the iSIGHT-FD optimization to be completed. After the optimizations have concluded, the Process Integrator reads a results file created by the optimization. The results are then presented to the user in the CAD package in a display window.

Finally, the new optimal design variables are used to instantiate an optimized CAD model, as displayed in Figure 4-9.

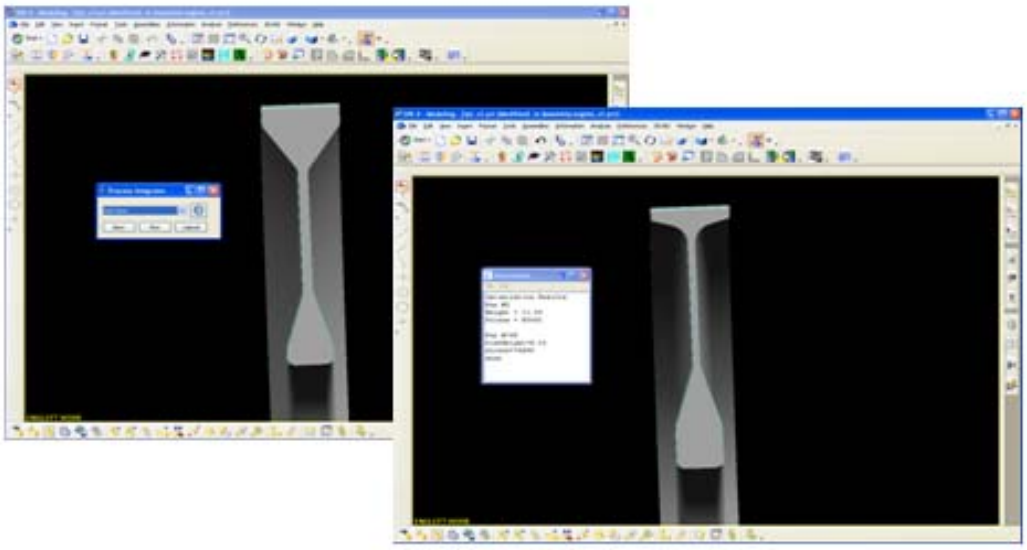

**Figure 4-9 Design is updated to optimum**

## 4.5 iSIGHT-FD Launcher

The iSIGHT-FD Launcher is a windows batch file that sets up and runs the iSIGHT-FD environment. The batch file also indicates to iSIGHT-FD the custom application to run. The batch file, as seen in Appendix D, calls two other batch files. The first, fiperenv.bat, sets several environment variables needed to run iSIGHT-FD and the second, launch.bat, actually launches iSIGHT-FD. The environment variable LaunchClasspath is used to specify the location of the program to be launched inside of iSIGHT-FD. The LaunchPgm variable indicates the name of the program at the LaunchClasspath location to be launched.

## 4.6 Optimization Workflow Builder

The Optimization Workflow Builder is the application that runs in iSIGHT-FD and builds optimization workflows to run analyses on the CAD model. The program performs four basic steps as follows:

1. Create iSIGHT-FD connection
2. Build the iSIGH-FD optimization workflow
3. Run the optimization workflow
4. Retrieve the job results from iSIGHT-FD

The connection is made to iSIGHT-FD by creating a ConnectionProfile object. The ConnectionProfile object is configured to read a connection profile specification file that is located in the iSIGHT-FD install directory. Next, the ConnectionProfile object is used to make the iSIGHT-FD connection and set up the needed iSIGHT-FD component libraries that will be used to build and execute the workflow.

The Optimization Workflow Builder builds a workflow that follows the configuration information specified by the Process Integrator. This information was stored in text files that the Workflow Builder reads and parses. Then the data is used to determine how to build the workflow. The workflow always has the same general layout, as seen in Figure 4-10. The information from the Process Integrator determines the number of optimization loops to create; there is one for each part to be optimized. Additionally, the text files indicate which parameters are to be used as design variables, etc for each optimization, as well as what analysis to configure the Simcode components to run. Each NX component is configured to open the same assembly, but the expressions to be updated and extracted are changed based on the part to be optimized.

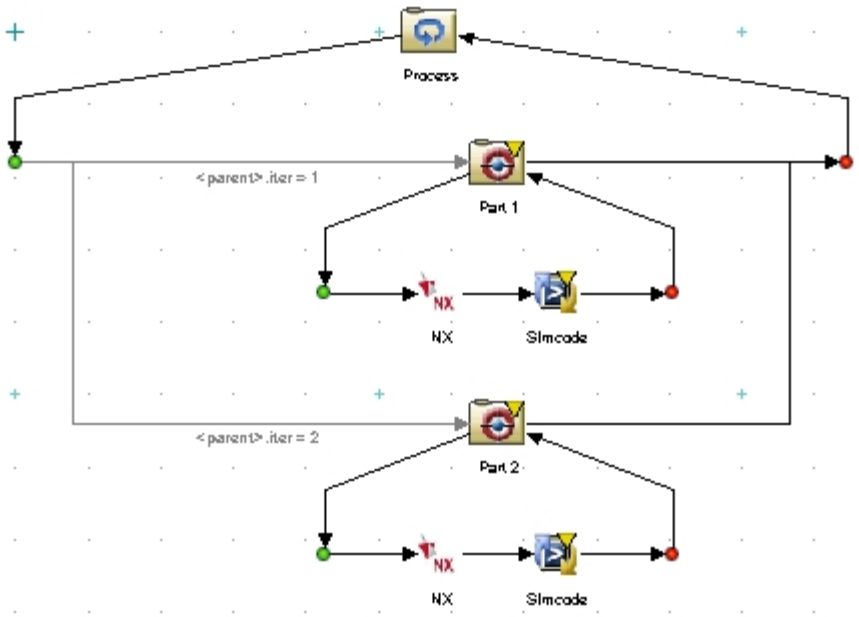


**Figure 4-10 iSIGHT-FD Workflow Template**

Once the workflow has been created it is saved in the Process Integrator directory under Results->Models. Then an iSIGHT-FD job is created to run the workflow. While the job is running, the Workflow builder queries the iSIGHT-FD run engine for a job status every 5 seconds to see if it has been completed. Every 20 seconds a job status is displayed for the user. When the job is completed, the results from each of the optimization loops are queried and saved in a text file in the Results folder. If the job does not complete within a user specified amount of time, the job is terminated and the program exists. Once the Optimization Workflow Builder program has terminated successfully, or unsuccessfully, the Process Integrator program, recognizes that it has terminated and continues its operations.

### 4.7 NX Component

As part of each optimization iteration, the NX assembly must have expressions updated to new values as design variables, and have data extracted as design constraints and/or objectives. A custom component was developed for iSIGHT-FD to perform this functionality. The component was written using the iSIGHT-FD and NXOpen APIs and published to the local iSIGHT-FD library for use in creating the needed workflows. The iSIGHT-FD component has two main functions: an initialization function and an execution function. The initialization function obtains an NX session and holds onto it for the duration of the iSIGHT-FD session. The initialization function saves a significant amount of time by getting the NX session only once. The process of obtaining the NX session takes roughly 30 seconds. If running many optimization iterations this amount of time would be prohibitive if it were to be repeated for each iteration. The execution function is called once during each optimization iteration. The execution function queries the iSIGHT-FD parameters that correspond to NX expressions and uses them to update the NX assembly. Then it queries which iSIGHT-FD parameters are needed and extracts them from the NX assembly.

# 5 Discussion of Results

Tests were performed to validate the implementation of the devised methods. Three test cases were performed. The first test, a beam analysis, tests the functionality for optimizing a single part with a single design variable. The second test, a disk analysis, tests the functionality of the process for optimizing multiple parts in an assembly with five design variables each. The third test, a static 25-bar truss, tests the functionality of the process for performing large optimization problems and has 25 design variables. Each of the three test cases connect a CAD model to a different type of analysis. The beam analysis was written in C++ and compiled as an executable. The disk analysis was written in Java and compiled as a Java Archive file. The static 25-bar truss is analyzed in Matlab using an M-file.

## 5.1 Beam Analysis Test Case

A beam analysis test case, as seen in Figure 5-1, was performed to determine how the methods perform to optimize a single part. The beam optimization was configured to change a single design variable, width. The objective of the optimization was to minimize weight. A maximum stress constraint was applied to the optimization. The Process Integrator was used to configure and run the optimization.

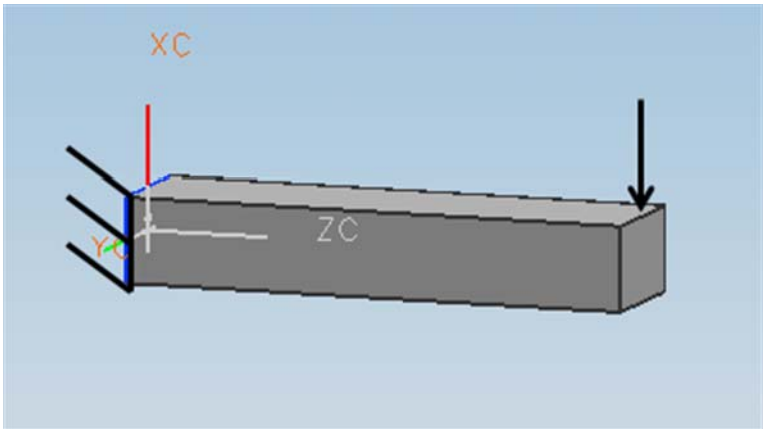


**Figure 5-1 Beam test case**

The beam analysis was able to be configured and run with smooth efficiency. Test results showed that it took under 5 minutes to configure the simple optimization at a leisurely pace. The test showed that the Process Integrator is a useful tool, capable of performing part optimizations. The Process Integrator provided a convenient interface for the set-up and execution of the desired optimization. Additionally, the next time that a beam analysis is needed the process configuration will be available for selection without requiring any set-up time.

## 5.2 Disk Analysis Test Case

A disk analysis test case was performed in which the disks in a jet turbine engine assembly, as displayed in Figure 5-2, were optimized. Several dimensions of the disks were used as design variables and the assembly was linked to an analysis to run a stress calculation based on part expressions and mass properties. Maximum stress was used as a design constraint. Disk weight was minimized as the design objective.

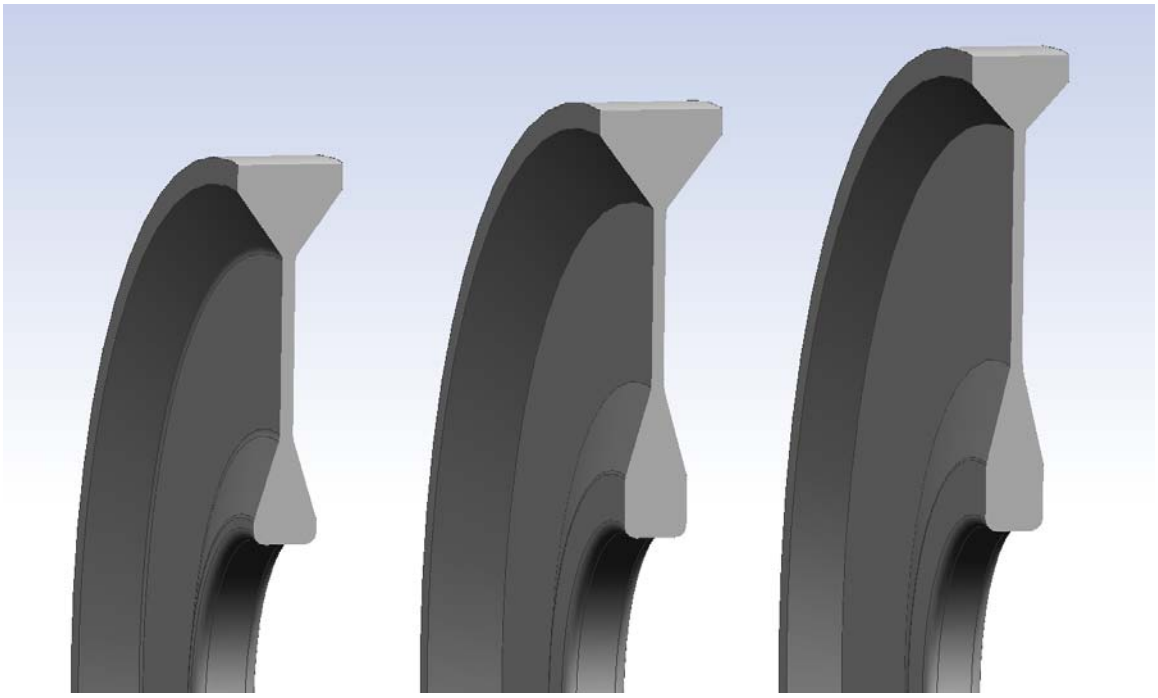

**Figure 5-2 Turbine disk cross-Sections**

Two methods were used to perform the test case. The first method was a base test in which iSIGHT-FD was used manually to create the workflow using a custom NX Component as the only custom technology. The second test was performed using the Process Integrator implementation to create a process configuration that automatically creates the workflow. The two methods were used to create the exact same iSIGHT-FD workflow, as seen in Figure 5-3. Because the two workflows are the same, they would require the same amount of time to execute, so the time to configure the workflow and not execution time will be examined.

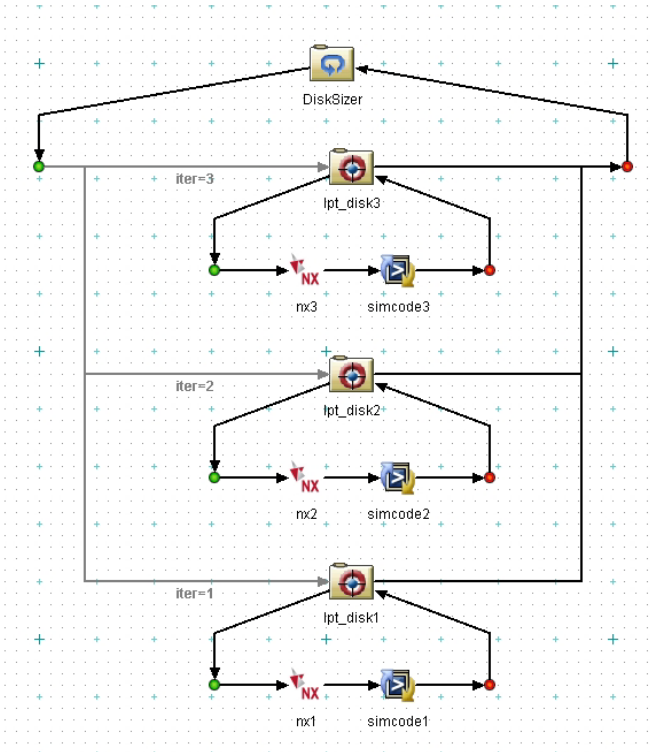


**Figure 5-3 Optimization loop used to optimize 3 turbine disks**

Both methods were performed by an experienced user of both the Process Integrator and iSIGHT-FD. The resulting times to configure an optimization loop for each disk in the assembly are displayed in Table 5-1. The results of the tests indicate that the workflow created using the Process Integrator took approximately 2/3 of the time to create as to manually create a single optimization loop. Once a generic process configuration has been defined, the Process Integrator takes approximately 3 seconds to create each part optimization. Manually, after the first optimization loop was created, the subsequent loops were created by creating a copy of the first loop and modifying the names of the expressions and parts to be used in the optimization.

**Table 5-1: Time to configure an optimization with one optimization loop per part**

| Part # | Manual (min:sec) | Process Integrator (min:sec) |
|---|---|---|
| 1 | 18:27 | 12:48 |
| 2 | 3:15 | :03 |
| 3 | 3:10 | :03 |
| Total | 24:52 | 12:54 |

An extrapolation of the test data, as seen in Figure 5-4, demonstrates how the time to create a workflow with many parts to be optimized stays relatively constant using the automated process. The time to create them using manual methods grows to undesirable quantities.

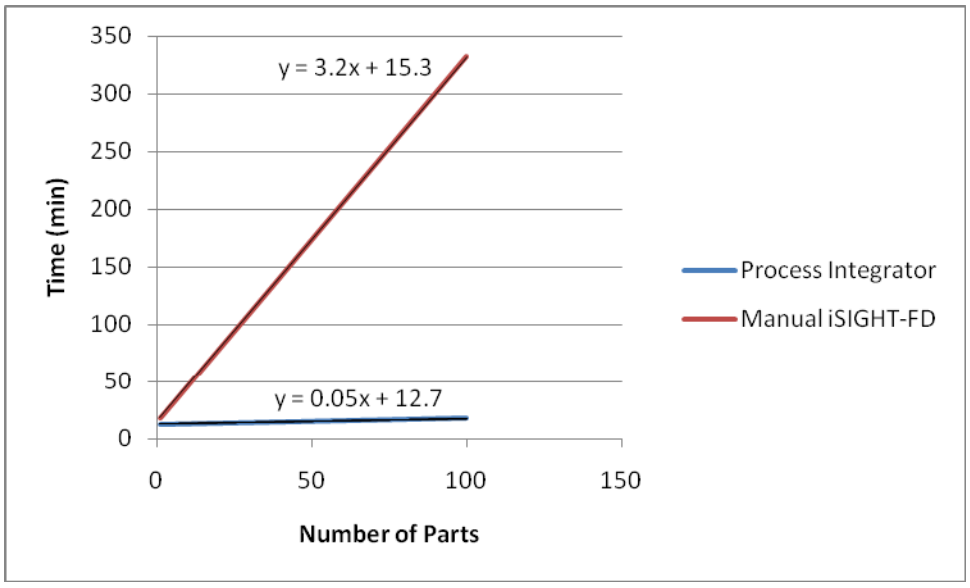


**Figure 5-4 A time extrapolation indicates the benefits of the automated process**

The benefits of using the Process Integrator become evident when it is desired to optimize many individual parts in an assembly. Although this test was performed on only three turbine disks, jet engines can have as many as 30 or more disks including both

the compressor and turbine. Configuring 30 optimizations manually would take almost 2 hours. The same optimizations could be created by the Process Integrator in under 2 minutes using a previously created configuration. This test case demonstrates the successful implementation of the methods presented in this thesis as well as its benefits.

## 5.3 Static 25-Bar Truss Test Case

The static 25-bar truss optimization, as displayed in Figure 5-5, was performed to demonstrate the capabilities of the Process Integrator to configure and execute a large optimization problem. The areas of each bar in the truss were used as design variables for this optimization. The analysis calculates the stress in each bar and a maximum stress constraint was imposed on the optimization (Rahami 2007). Finally, weight was minimized as the design objective.

The truss optimization was easily configured using the Process Integrator. The optimization drove each bar area to the minimum allowed value that would not violate the maximum stress constraint, as shown in Appendix E. This test demonstrated that the Process Integrator can be used to configure and execute large optimization problems.

## 5.4 Significance of Results

The successful execution of the three test cases using the Process Integrator demonstrates that the methodology developed by this research is both feasible and beneficial. The results of these test cases indicate that the implementation of this method provides significant time savings over traditional part optimization methods.

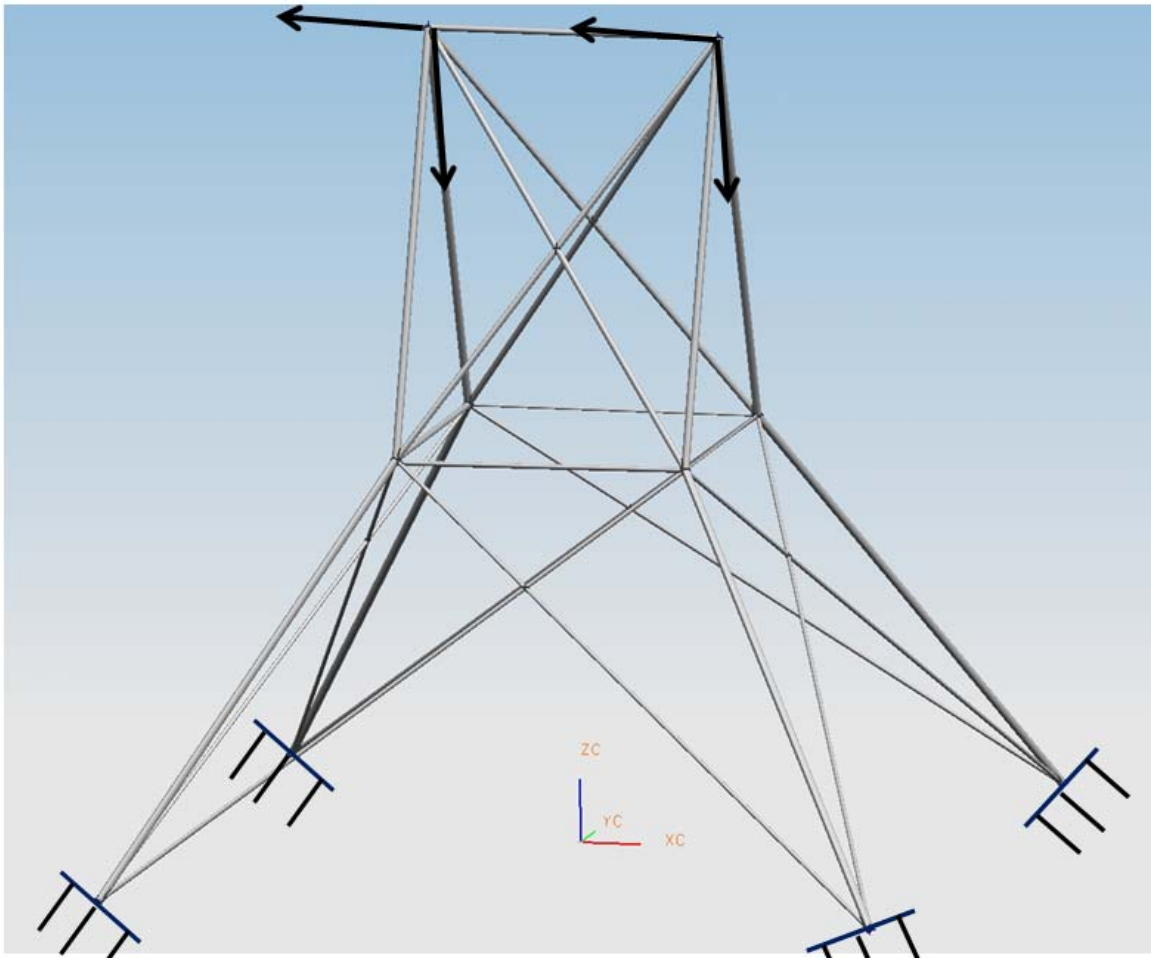


**Figure 5-5 Static 25-bar truss**

# 6 Conclusion

The goal of this research has been to resolve the conflict that exists between the role of CAD during the design process and its role during trade studies. As explained in Chapter 1, the predominant role of CAD occurs because of the way it allows the designer to interact with the design. The role of CAD is less significant during trade studies because the optimization controls it rather than the designer. The lack of the designer in the design loop during trade studies is the root of the inconsistency addressed by this research.

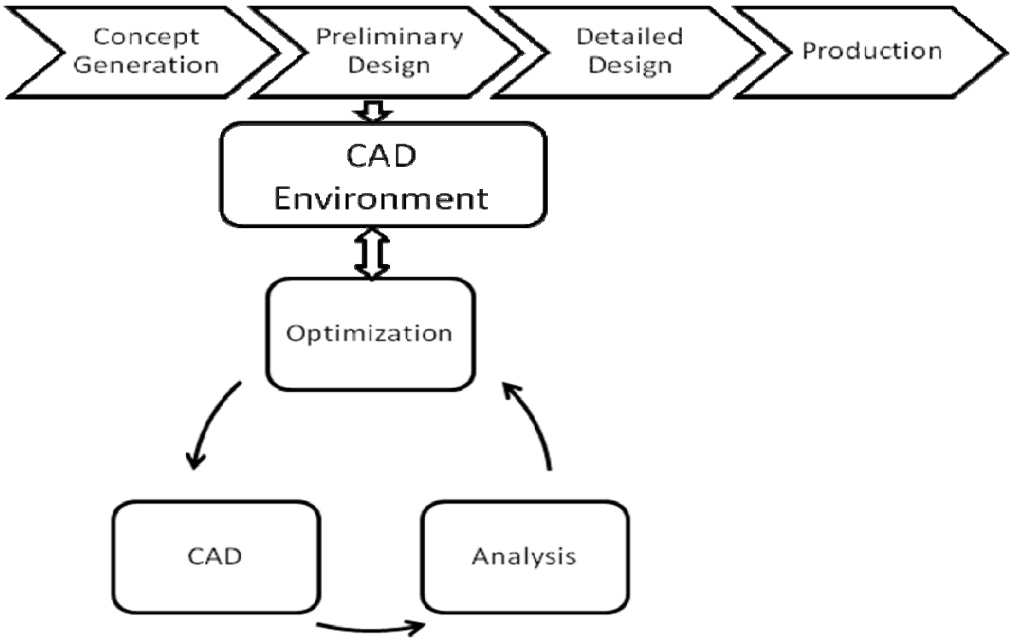


**Figure 6-1 Trade studies performed from CAD environment**

Performing trade studies from within CAD, where the designer has the ability to interact with the design, is the solution to this problem, as portrayed by Figure 6-1. Methods have been developed, as presented in Chapter 3, to solve the problems associated with defining and performing trade studies in this fashion. The implementation and case studies performed using the developed methods demonstrate the feasibility and benefits of this methodology.

## 6.1 Contribution

As discussed in Chapter 5, the use of these methods can provide significant time savings over conventional methods in applicable cases. However, the real benefit of this method is not merely automating a single process and saving five hours, but creating a framework for widespread use of CAx tools. This approach allows for the development of better designs earlier in the design process, by providing a tool to perform studies in the preliminary design phase rather than only in the detailed design phase, as expressed by Figure 6-1. Performing these studies in an earlier design phase than traditionally feasible allows for more knowledge early in the design process, see Figure 6-2. Modelers who don't have time or training to perform analyses on preliminary designs can be supplied with a tool that can automatically perform design iterations. Engineering knowledge can be leveraged for their designs after they leave work for the night through the automated process. Engineered designs, rather than design space definition, can be passed from preliminary design teams to detailed design teams requiring a negligible added work load for the preliminary design team. In this way, engineering knowledge

can be gathered while there is design freedom to make significant changes to the design. Additionally, significant time savings can be provided for the detailed design team.

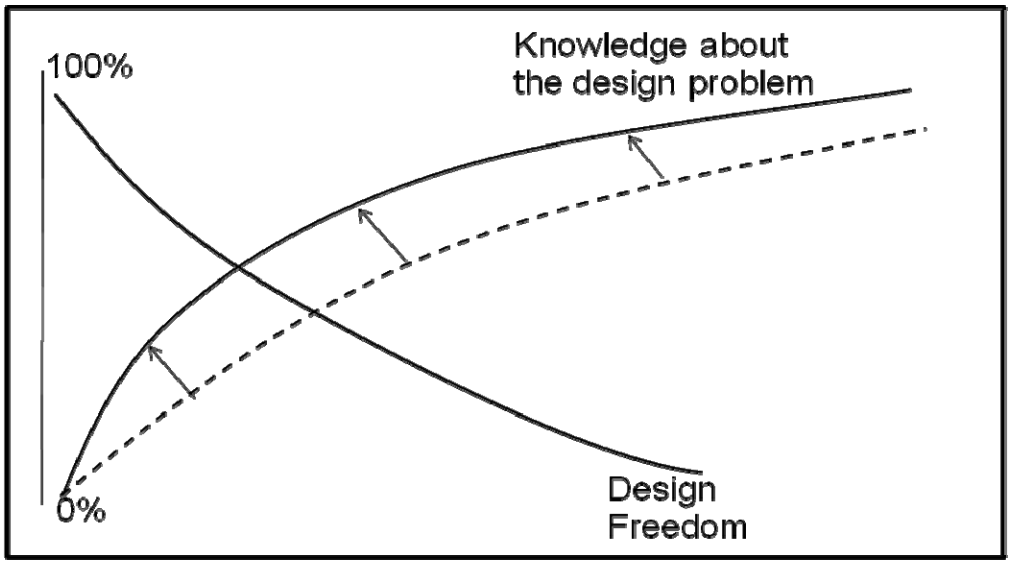


**Figure 6-2 Increase of slope of knowledge gaining curve indicates improved product knowledge**

The methodology presented by this research creates a huge advantage for performing trade studies because of the fact that it can automatically create a virtually unlimited number of optimization loops. Additionally, this method reduces the amount of training required to enable mechanical designers to run perform trade studies on CAD models. CAD-Centric Dynamic Workflow Creation is a leap forward towards leveraging engineering knowledge in CAD models. The implementation of this method leads to more accurate models at an earlier design stage and to big savings in time and money throughout the design cycle. Finally, because this method has been implemented in a very generic application it successfully balances automation with the ability to innovate within a process. By not restricting the optimization process to a specific analysis or to

run on certain types of parts, this method can be easily adapted to new products and company practices.

## 6.2 Future Work

CAD-Centric Dynamic Workflow generation is a great foundation for automating the many engineering tasks that might go into creating a product. A useful enhancement to the demonstrated implementation of this method would be the incorporation of multiple optimization configuration templates. Additionally, a method to create new templates to be configured would also be a good addition. Also, the incorporation of an efficient method to make all of the components in the iSIGHT-FD library available for use would be a good enhancement to the implementation.

The application of this methodology to a PLM system rather than a standalone CAD system would also be an interesting and useful direction to take this research.

# 7 References


Blanding, Robert; Turkiyyah, George: ECAD - A Prototype Screen-based VR Solid Modeling Environment Incorporating Tangible Deformable Models, Computer-Aided Design & Applications, 4(5), 2007, 595-605.

Berglund, Courtney L.; Jensen, C. Greg: Robust Parameterization Schema for CAx Master Models, Computer-Aided Design & Applications, 5(5), 2008, 715-729.

Danjou, Stephane; Lupa, Norman; Koehler, Peter: Approach for Automated Product Modeling Using Knowledge-Based Design Features, Computer-Aided Design & Applications, 5(5), 2008,622-629.

Dye, Christopher; Staubach, Joseph B.; Emmerson, Diane; Jensen, C. Greg.: CAD-Based Parametric Cross-Section Designer for Gas Turbine Engine MDO Applications, Computer-Aided Design & Applications, 4(1-4), 2007, 509-518.

Eck, David J.: Java Notes, http://math.hws.edu/javanotes/, .Accessed January 29, 2008.

Elliot, Jason H.; Berglund, Courtney L.; Jensen, C. Greg: An Automated Approach to Feature-Based Design for Reusable Parameter-Rich Surface Models, Computer-Aided Design & Applications, 4(1-4), 2007, 498-507.

King, Matthew Lee: A CAD-Centric Approach to CFD Analysis with Discrete Features, Brigham Young University, 2004.

Koch, Patrick N.; Evans, J.P.; Powell, David: Interdigitation for Effective Design Space Exploration Using iSIGHT, Journal of Structural and Multidisciplinary Optimization, 23(2), 2002, 111-126.

Kosavinta, Satakhun; Kanongchaiyos, Pizzanu; Jinuntuya, Pinyo; Integration of CAD Software with DSS for Engineering and Architectural Project Design, Computer-Aided Design & Applications, 4(1-4), 2007, 467-476.

Lai, Yuan-Lung: A constraint-based system for product design and manufacturing, Robotics and Computer-Integrated Manufacturing, 25, 2009, 246-258.

Lamarche, Bruno; Rivest, Louis : Dynamic Product Modeling with Inter-Features Associations: Comparing Customization and Automation, Computer-Aided Design & Applications, 4(6), 2007, 877-886.

Lee, Sang Hun: A CAD-CAE integration approach using feature-based multi-resolution and multi-abstraction modeling techniques, Computer-Aided Design, 37, 2005, 941-955.

Lippman, Stanley B.; Lajoie, Josee; Moo, Barbara E.: C++ Primer 4ed., Addison-Wesley, Upper Saddle River, 2005.

Norton, Robert L.: Machine Design: An Integrated Approach 3ed., Pearson Education, Inc, Upper Saddle River, NJ, 2005.

Pahl, G.; Beitz, W.; Feldhusen, J.; Grote, K.H.: Engineering Design: A Systematic Approach 3rd Ed., Springer-Verlag, London Limited, 2007

Perlak, Jeff: Personal interview, November 2007.

Rahami, Hossein: Truss Analysis, http://www.mathworks.com/matlabcentral/fileexchange/14313, 2007, Accessed February, 24, 2009.

Salzman, Harold: Computer-Aided Design: Limitations in Automating Design and Drafting, IEEE Transactions on Engineering Management, 36(4), 1989, 252-261.

Shahin, Tamer M. M.: Features-Based Design – An Overview, Computer-Aided Design & Applications, 5(5), 2008, 639-653.

Shehab, E. M.; Abdalla, H. S.: A cost-effective knowledge-based reasoning system for design for automation, Proc. IMechE, 220(Part B: J. Engineering Manufacture), 2006, 729-743.

Siddique, Zahed; Ninan, Jiju A.: Modeling of modularity and scaling for integration of customer in design of engineer-to-order products, Integrated Computer-Aided Engineering, 2006, 13(2), 133-148.

Tappeta, R.V.; Nagendra, S.; Renaud, J.E.: A multidisciplinary design optimization approach for high temperature aircraft engine components, Structural Optimization, 18, 1999, 134-145.

Ullman, David G.: The Mechanical Design Process 4th ed., McGraw Hill Higher Education, Boston, 2010, 19.

Ulrich, Karl T.; Eppinger, Steven D.: Product Design and Development 3ed, McGraw-Hill, New York, 2004

Velden, Alex Van der: CAD to CAE process automation through iSIGHT-FD, Proceedings of the ASME Turbo, Expo, 1, 2007, 87-93

Velden, Alex Van der; Kokan, David: The Synaps Pointer Optimization Engine, Proceedings of DETC/CIE: Computers and Information in Engineering Conference, 2002.

Wang, Charlie, C. L.; Hui, Kin-Chuen; Tong, Kai-Man: Volume Parameterization for Design Automation of Customized Free-Form Products, IEEE Transactions on Automation Science and Engineering, 4(1), 2007, 11-21.

Wang, S. Y.; Tai, K.; Bar-system representation for topology optimization using genetic algorithms, Engineering Computations [serial online], 2005, 22, 206-231. Available from: ABI/INFORM Global. Accessed January 29, 2009, Document ID: 828950401.

Wind, J.W.; Perdahcioglu, D. Akcay; Boer, A. de: Distributed multilevel optimization for complex structures, Stuct Multidisc Optim, 36, 2008, 71-81.

Zeid, Ibrahim: Mastering CAD/CAM, McGraw-Hill, New York, 2005, 151

# Appendix A. Sample Initialization File

```
MaxTime=45
PartDef=subsetcontain=disk
[engine_cf, lpt_cf, lpt_disk3]
DesignVars
[engine_cf, lpt_cf];sw=DISK3;DISK3_RimTk;0.1;0.296994422076;1.0;1.0
[engine_cf, lpt_cf];sw=DISK3;DISK3_RimTpTk;0.1;0.5;1.5;1.0
[engine_cf,
lpt_cf];sw=DISK3;DISK3_NeckWd;0.1;0.1242707453465401;1.0;1.0
[engine_cf,
lpt_cf];sw=DISK3;DISK3_BoreTpTk;0.1;1.0340098541414398;2.0;1.0
[engine_cf,
lpt_cf];sw=DISK3;DISK3_BoreTk;0.1;0.7010131817845199;2.0;1.0
ObjVars
[engine_cf, lpt_cf,
lpt_disk3];Weight;0.0;76.4966959185043;100.0;1.0;minimize
ConsVars
stress;0.0;76350.78709387447;75000.0;1.0
NXextract
[engine_cf, lpt_cf, lpt_disk3];Mass=76.4966959185043
[engine_cf, lpt_cf, lpt_disk3];Volume=270.4018943743524
[engine_cf, lpt_cf, lpt_disk3];Weight=76.4966959185043
[engine_cf, lpt_cf, lpt_disk3];Radius Of Gyration=17.175148494493868
[engine_cf, lpt_cf];RPM=21000.0
[engine_cf, lpt_cf];BladeMass=3.6
[engine_cf, lpt_cf];BladeRgy=7.4
[engine_cf, lpt_cf];sw=DISK3;DISK3_RimTpTk=0.5
[engine_cf, lpt_cf];sw=DISK3;DISK3_RimTk=0.296994422076
[engine_cf, lpt_cf];sw=DISK3;DISK3_RimWd=1.040580000000001
[engine_cf, lpt_cf];sw=DISK3;DISK3_RimLivRd=19.2753
[engine_cf, lpt_cf];sw=DISK3;DISK3_NeckWd=0.1242707453465401
[engine_cf, lpt_cf];sw=DISK3;DISK3_BoreTk=0.7010131817845199
[engine_cf, lpt_cf];sw=DISK3;DISK3_BoreTpTk=1.0340098541414398
[engine_cf, lpt_cf];sw=DISK3;DISK3_BoreRd=13.992555908518442
[engine_cf, lpt_cf];sw=DISK3;DISK3_BoreWd=0.6181076451938406
ProcessData
InputFile=C:\ug_customJava2\DiskAnalysis\dist\diskAnalysisInput.txt
OutputFile=C:\ug_customJava2\DiskAnalysis\dist\diskAnalysisOutput.txt
OutputTemplate=C:\ug_customJava2\DiskAnalysis\disAnalysisOutputTemplate
.txt
InputSplitter==
OutputSplitter==
Command=java -jar C:\ug_customJava2\DiskAnalysis\dist\DiskAnalysis.jar
InputFileParms
Mapped:DiskMass:[engine_cf, lpt_cf, lpt_disk3];Mass=76.4966959185043
```

```
Mapped:DiskVolume:[engine_cf, lpt_cf,
lpt_disk3];Volume=270.4018943743524
Mapped:DiskRgy:[engine_cf, lpt_cf, lpt_disk3];Radius Of
Gyration=17.175148494493868
Mapped:DiskWeight=[engine_cf, lpt_cf,
lpt_disk3];Weight=76.4966959185043
Mapped:BladeRgy:[engine_cf, lpt_cf];BladeRgy=7.4
Mapped:RPM:[engine_cf, lpt_cf];RPM=21000.0
Mapped:BladeMass:[engine_cf, lpt_cf];BladeMass=3.6
Mapped:DISK_RimTpTk:[engine_cf, lpt_cf];sw=DISK3;DISK3_RimTpTk=0.5
Mapped:DISK_RimTk:[engine_cf,
lpt_cf];sw=DISK3;DISK3_RimTk=0.296994422076
Mapped:DISK_BoreTpTk:[engine_cf,
lpt_cf];sw=DISK3;DISK3_BoreTpTk=1.0340098541414398
Mapped:DISK_BoreTk:[engine_cf,
lpt_cf];sw=DISK3;DISK3_BoreTk=0.7010131817845199
Mapped:DISK_NeckWd:[engine_cf,
lpt_cf];sw=DISK3;DISK3_NeckWd=0.1242707453465401
Mapped:DISK_BoreRd:[engine_cf,
lpt_cf];sw=DISK3;DISK3_BoreRd=13.992555908518442
Mapped:DISK_BoreWd:[engine_cf,
lpt_cf];sw=DISK3;DISK3_BoreWd=0.6181076451938406
Mapped:DISK_RimWd:[engine_cf,
lpt_cf];sw=DISK3;DISK3_RimWd=1.040580000000001
Mapped:DISK_RimLivRd:[engine_cf,
lpt_cf];sw=DISK3;DISK3_RimLivRd=19.2753
OutputFileParms
Output:stress
Output:DiskWeight
Output:BoreTkUP
Output:BoreTpTkUp
Output:RimtkUp
Output:NeckWdUp
Output:RimpTpTkUp
```

# Appendix B. Sample Runtime File

ProcessName=DiskSizer
DisplayPart=C:\Documents and Settings\Travis\Desktop\diskAssmTest\engine_cf.prt
[engine_cf, lpt_cf, lpt_disk1]
DesignVars
[lpt_cf];disk1_RimTk;0.1;0.28;1.0;1.0
[lpt_cf];disk1_RimTpTk;0.1;0.7;1.5;1.0
[lpt_cf];disk1_NeckWd;0.05;0.14;1.0;1.0
[lpt_cf];disk1_BoreTpTk;0.1;0.86;2.0;1.0
[lpt_cf];disk1_BoreTk;0.1;0.25;2.0;1.0
ObjVars
[engine_cf, lpt_cf, lpt_disk1];MP_Weight;0.0;76.4966959185043;100.0;1.0;minimize
ConsVars
stress;0.0;76350.78709387447;75000.0;1.0
[engine_cf, lpt_cf, lpt_disk2]
DesignVars
[lpt_cf];disk2_RimTk;0.1;0.275398051984;1.0;1.0
[lpt_cf];disk2_RimTpTk;0.1;0.78388;1.5;1.0
[lpt_cf];disk2_NeckWd;0.05;0.13437405797633997;1.0;1.0
[lpt_cf];disk2_BoreTpTk;0.1;0.9461405010790398;2.0;1.0
[lpt_cf];disk2_BoreTk;0.1;0.6300107359993201;2.0;1.0
ObjVars
[engine_cf, lpt_cf, lpt_disk2];MP_Weight;0.0;76.4966959185043;100.0;1.0;minimize
ConsVars
stress;0.0;76350.78709387447;75000.0;1.0
[engine_cf, lpt_cf, lpt_disk3]
DesignVars
[lpt_cf];disk3_RimTk;0.1;0.296994422076;1.0;1.0
[lpt_cf];disk3_RimTpTk;0.1;0.5;1.5;1.0
[lpt_cf];disk3_NeckWd;0.05;0.1242707453465401;1.0;1.0
[lpt_cf];disk3_BoreTpTk;0.1;1.0340098541414398;2.0;1.0
[lpt_cf];disk3_BoreTk;0.1;0.7010131817845199;2.0;1.0
ObjVars
[engine_cf, lpt_cf, lpt_disk3];MP_Weight;0.0;76.4966959185043;100.0;1.0;minimize
ConsVars
stress;0.0;76350.78709387447;75000.0;1.0

# Appendix C. Sample Results File

Part:lpt_disk1
DISK1_BoreTk=0.25
DISK1_BoreTpTk=0.86
DISK1_NeckWd=0.14
DISK1_RimTk=0.28
DISK1_RimTpTk=0.7
Run #=0
DISK1_BoreTk=0.323652696723423
DISK1_BoreTpTk=1.0849995244853117
DISK1_NeckWd=0.1
DISK1_RimTk=0.1
DISK1_RimTpTk=0.1
Run #=189
stress=74974.88561564762
Weight=6.739615789479852

# Appendix D. Fiper Launcher Windows Batch File

```
setlocal
call %fiperhome%\bin\win32\fiperenv.bat
LaunchClasspath=C:\ug_customJava2\FiperWorkFlow\dist\FiperWorkFlow.jar;%FiperJa
rs%

set LaunchPgm= FiperWorkFlow
set LaunchArgs=%*
call %fiperhome%\bin\win32\launch.bat
```

# Appendix E. Beam Analysis Test Results

| Design | Constraint | Objective |
|---|---|---|
| | Stress < 30ksi | min Weight |
| **Starting Design** | | |
| H = 3 | 80000 | 203.68 |
| **Final Design (16)** | | |
| H = 4.8989 | 30000 | 332.1 |

# Appendix F. Disk Analysis Test Results

| Design | Constraint | Objective |
|---|---|---|
| | Stress < 75ksi | min Weight |
| Disk 1 | | |
| **Starting Design** | | |
| DISK1_BoreTk=0.25 | 84934.05 | 11.79 |
| DISK1_BoreTpTk=0.86 | | |
| DISK1_NeckWd=0.14 | | |
| DISK1_RimTk=0.28 | | |
| DISK1_RimTpTk=0.7 | | |
| **Final Design (189)** | | |
| DISK1_BoreTk=0.323 | 74974.885 | 6.739 |
| DISK1_BoreTpTk=1.084 | | |
| DISK1_NeckWd=0.1 | | |
| DISK1_RimTk=0.1 | | |
| DISK1_RimTpTk=0.1 | | |
| Design | Constraint | Objective |
| | | |
| Disk 2 | | |
| **Starting Design** | | |
| DISK2_BoreTk=0.63 | 85329.723 | 15.65 |
| DISK2_BoreTpTk=0.94 | | |
| DISK2_NeckWd=0.13 | | |
| DISK2_RimTk=0.27 | | |
| DISK2_RimTpTk=0.78 | | |
| **Final Design (215)** | | |
| DISK2_BoreTk=1.315 | 74999.999 | 13.141 |
| DISK2_BoreTpTk=1.117 | | |
| DISK2_NeckWd=0.1 | | |
| DISK2_RimTk=0.251 | | |
| DISK2_RimTpTk=0.1 | | |

| Design | Constraint | Objective |
|---|---|---|
| | | |
| Disk 3 | | |
| **Starting Design** | | |
| DISK3_BoreTk=0.7 | 91164.429 | 13.72 |
| DISK3_BoreTpTk=1.03 | | |
| DISK3_NeckWd=0.124 | | |
| DISK3_RimTk=0.296 | | |
| DISK3_RimTpTk=0.5 | | |
| **Final Design (148)** | | |
| DISK3_BoreTk=1.266 | 74999.999 | 9.74 |
| DISK3_BoreTpTk=0.580 | | |
| DISK3_NeckWd=0.1 | | |
| DISK3_RimTk=0.1 | | |
| DISK3_RimTpTk=0.1 | | |

# Appendix G. Static 25-Bar Truss Optimization Results

| Design | Constraint | Objective | Design | Constraint | Objective |
|---|---|---|---|---|---|
| All bar areas | Stress < 30ksi | min Weight | | | |
| **Starting Design** | | | **Final Design (1857)** | | |
| a1=.4 | s1=116760.993 | Weight=1376.9 | a1=1.126 | s1=29997.574 | Weight=2703.3 |
| a2=.1 | s2=427244.400 | | a2=5.155 | s2=377.068 | |
| a3=.1 | s3=442781.141 | | a3=2.851 | s3=29998.624 | |
| a4=.1 | s4=500126.310 | | a4=4.022 | s4=28162.109 | |
| a5=.1 | s5=369899.231 | | a5=1.184 | s5=22586.206 | |
| a6=3.4 | s6=15518.907 | | a6=0.735 | s6=25825.724 | |
| a7=3.4 | s7=66491.595 | | a7=5.796 | s7=29999.882 | |
| a8=3.4 | s8=36460.093 | | a8=3.472 | s8=26084.589 | |
| a9=3.4 | s9=45550.409 | | a9=4.823 | s9=21375.798 | |
| a10=.4 | s10=2893.133 | | a10=0.572 | s10=1830.450 | |
| a11=.4 | s11=15267.836 | | a11=0.295 | s11=20838.78 | |
| a12=.4 | s12=58400.931 | | a12=0.486 | s12=14422.481 | |
| a13=1.3 | s13=50341.376 | | a13=0.477 | s13=19478.583 | |
| a14=.9 | s14=36573.363 | | a14=5.056 | s14=6921.841 | |
| a15=.9 | s15=30516.34370 | | a15=4.946 | s15=5165.39 | |
| a16=.9 | s16=10338.994 | | a16=0.495 | s16=18700.352 | |
| a17=.9 | s17=56750.712 | | a17=1.698 | s17=29998.536 | |
| a18=1 | s18=10515.168 | | a18=0.525 | s18=7376.342 | |
| a19=1 | s19=40705.536 | | a19=1.141 | s19=29998.503 | |
| a20=1 | s20=56212.197 | | a20=2.464 | s20=29995.699 | |
| a21=1 | s21=26021.829 | | a21=4.950 | s21=8869.162 | |
| a22=3.4 | s22=34205.626 | | a22=4.164 | s22=25220.902 | |
| a23=3.4 | s23=42926.971 | | a23=4.989 | s23=29997.899 | |
| a24=3.4 | s24=2247.227 | | a24=0.451 | s24=26705.252 | |
| a25=3.4 | s25=74885.370 | | a25=8.046 | s25=29969.248 | |