SANDIA REPORT
SAND2000–8255
Unlimited Release
Printed August 2000

SIENA Customer Problem Statement And Requirements

Ly Sauer, Robert Clay, Charles Adams, Howard Walther, Ben Allan, Robert Mariano, Clark Poore, Bob Whiteside, Barry Boughton, Jay Dike, Edward Hoffman, Roy Hogan, Carole LeGall

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Sandia National Laboratories

SAND2000-8255
Unlimited Release
Printed August 2000

SIENA Customer Problem Statement and Requirements

Ly Sauer, Robert Clay, Charles Adams, Howard Walther, Ben Allan, Robert Mariano, Clark Poore, Bob Whiteside, Barry Boughton, Jay Dike, Edward Hoffman, Roy Hogan, Carole LeGall

Abstract

This document describes the problem domain and functional requirements of the SIENA framework. The software requirements and system architecture of SIENA are specified in separate documents (called *SIENA Software Requirement Specification* and *SIENA Software Architecture*, respectively).

While currently this version of the document describes the problems and captures the requirements within the *Analysis* domain (concentrating on finite element models), it is our intention to subsequently expand this document to describe problems and capture requirements from the *Design* and *Manufacturing* domains. In addition, SIENA is designed to be extendible to support and integrate elements from the other domains (see *SIENA Software Architecture* document).

This page intentionally left blank

Table of Contents

This page intentionally left blank.

1. Introduction

The move from a physical-testing-based approach to a more simulation-based approach for weapon engineering and certification along with the requisite solution of larger, and more complex computational models necessitates changes in the methodology that is employed by Sandia and the Nuclear Weapons Complex. The simulation-based methodology for a weapon can be delineated into three phases: *Pre-Processing*, *Processing*, and *Post-Processing*. The objective of the *Pre-Processing* phase is to determine which simulation studies are required, to assemble the inputs needed for each simulation study, and to build the simulation models for the analysis code required to produce the results. The preparation of simulation studies involves a complex sequence of steps to disassemble solid models (weapons and their environment), meshing geometric models, re-assembling human-managed and distributed meshed parts and assemblies, and specifying the input parameters of the analysis code. Specifically, the input to a simulation is the simulation model, which is a meshed representation of the analysis solid geometry with boundary conditions, initial conditions, loads, contacts, materials, and element properties. During the *Processing* phase, the simulation models are submitted to the analysis code for solution. The *Post-Processing* phase is where the simulation results are analyzed with the aid of visualization software packages.

SIENA (System for Integrating ENgineering and Analysis) is a software framework that provides a set of services to manage and maintain persistence of the design, pre-processing, processing, post-processing, and manufacturing artifacts and processes. This document describes the simulation-based environment and indicates areas where SIENA is used to improve and mature (time reduction, quality improvement, and ease of maintenance) the current processes. These areas of improvements are used as requirements for realizing SIENA.

2. Problem Domain Description

Figure 1 depicts the artifacts (models, data, information, documentation, and more), tools, and software packages *currently* being used in Sandia's simulation-based weapon engineering environment. Table 1 provides a summary of the artifacts and their associated attributes.

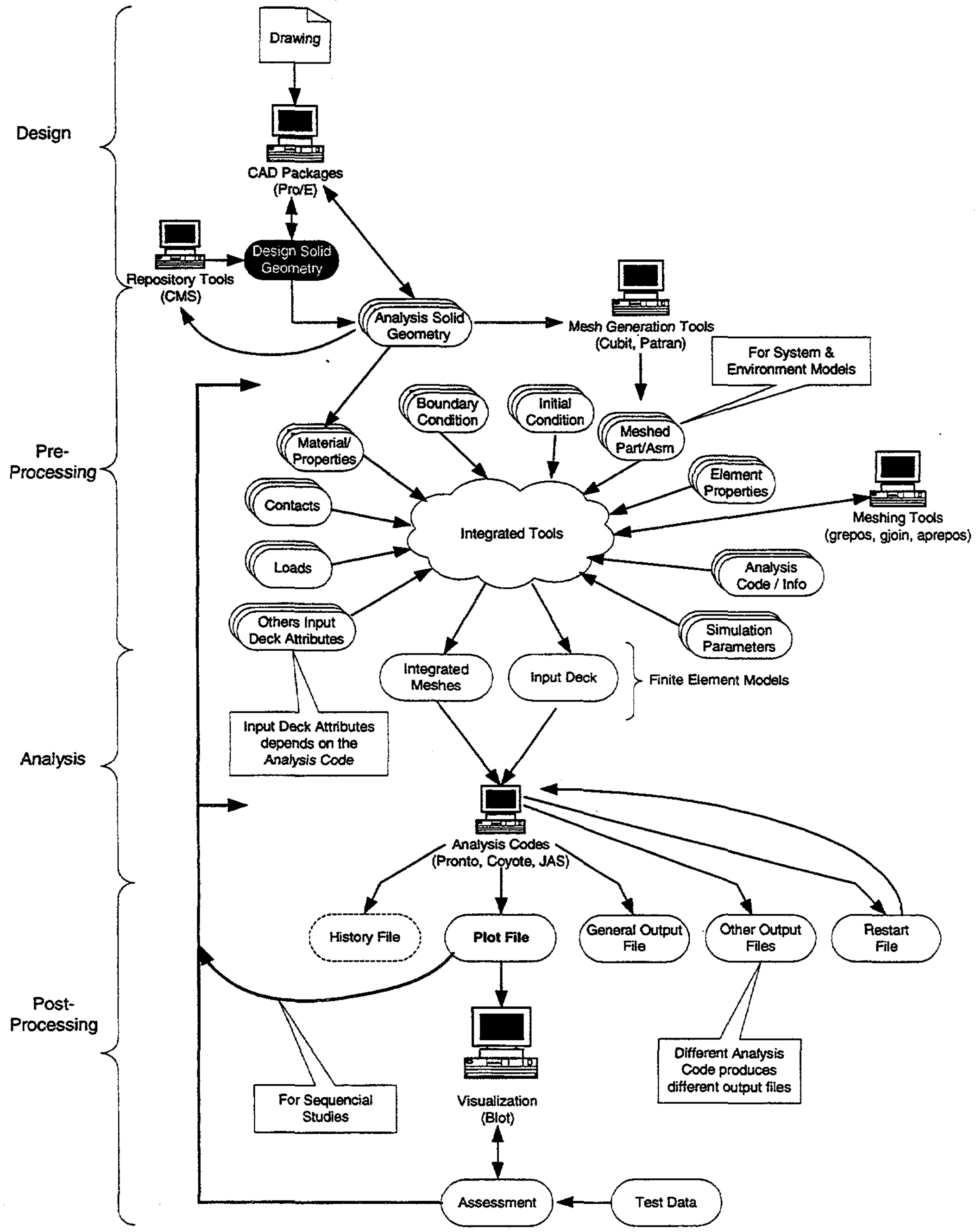

Figure 1. Artifacts of the Simulation Environment

Table 1. Pre-Processing, Analysis, and Post-Processing's Artifacts and Their Attributes

Artifacts	**Kind**	**Format**	**Description**
Design Solid Geometry	Model	Pro/E Geometry	See Section 2.1
Analysis Solid Geometry	Model	Pro/E Geometry	See Section 2.2
(System, Environment, & Simulation) Finite Element Model	Model	Depends on the analysis code (i.e., in PRONTO3D [5], it is the integrated meshes and its input deck)	See Section 2.11
Meshed Part/Assemblies	Mesh	Depends on the analysis code (i.e., PRONTO3D requires the meshes to be in ExodusII** format, BDF for NASTRAN/COTS)	See Section 2.3
Material /Properties	Data	User Provided Number & String	See Section 2.4
Bound Condition	Data	User Provided Number & String	See Section 2.5
Initial Condition	Data	User Provided Number & String	See Section 2.6
Loads	Data	User Provided Number & String	See Section 2.7
Element Properties	Data	User Provided String	See Section 2.8
Contacts	Data	User Provided String	See Section 2.9
Input Deck	Data	User Provided String	See Section 2.12
Simulation Parameters	Data	User Provided String	See Section 2.13
General Output File	Data	Analysis Code Generated ASCII file	See Section 2.16
History File	Data	Analysis Code Generated Result. For some analysis code, the result is the ExodusII** file	See Section 2.17
Plot File	Data	Analysis Code Generated ASCII or ExodusII** file	See Section 2.18
Restart File	Data	Analysis Code Generated Exodus** file	See Section 2.19
Test Data	Data	A text-based report consisting of graphs, tables, and references	See Section 2.20
Analysis Report	Data	A text-based document	See Section 2.21
Assessment Report	Data	A text-based document	See Section 2.22
Drawing	Data	Hard copy drawing	See Section 2.23
Other Input Deck Attributes	Depends on the attribute		See Section 2.10
Integration Tools	Plain text makefile, script file, executable program, or other software as needed.		See Section 2.14
Analysis Code	Executable Program		See Section 2.15

** ExodusII may be replaced with DMF (Data Models and Format) in the near future.

As depicted in Figure 1, one of the prerequisites for a simulation-based analysis is the *Design Solid Geometry* (DSG) produced by the designer. The designer uses a CAD package (e.g., Pro/E or SolidWorks) where the resulting files are optionally stored in (for example) the Workgroup Technologies (WTC) PDM system, commonly referred to as the *Configuration Management System* (CMS). From the design solid geometry for the weapon and its environment*, various other models, data, information, and documentation are generated in order to create the finite element models needed for simulations. The remainder of this section describes the simulation environment, the various artifacts, and their associated attributes (Sections 2.1 – 2.23). Section 2.24 outlines the typical steps in the pre-processing, processing, and post-processing phases.

2.1. Design Solid Geometry

The Design Solid Geometry is a software, geometric representation of a weapon and/or weapon environment. Different CAD software packages use different strategies to represent and capture the various parts that compose the system. For Pro/E, the DSG is represented in a hierarchical, tree-like organization as shown in Figure 2. The root of the tree is an assembly that describes how the weapon and/or its environment are assembled. It may consist of *n* levels of sub-assemblies in order to capture all the complexities of the weapon geometry. The leaves of the tree are parts, where each part describes a homogeneous solid (e.g., a nut, bolt, or machined piece of metal).

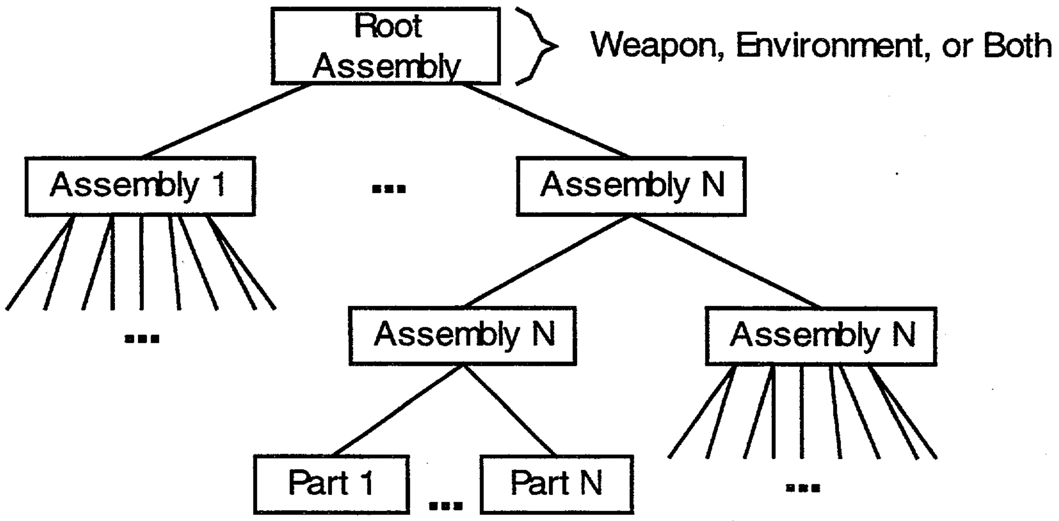

Figure 2. Design Solid Geometry Structure

In the case of Pro/E, *.asm and *.prt files are generated for assemblies and parts, respectively. A weapon may consist of multiple *.asm and *.prt files.

* The environment in which the weapon may be (e.g., targets, attachment arms, etc.). The Stockpile to Target Sequence (STS) document specifies the operational environments of the weapon. In some cases, the STS is a living document.

A Design Solid Geometry tree structure for a *Widget* generated by Pro/E might look like the diagram in Figure 3.

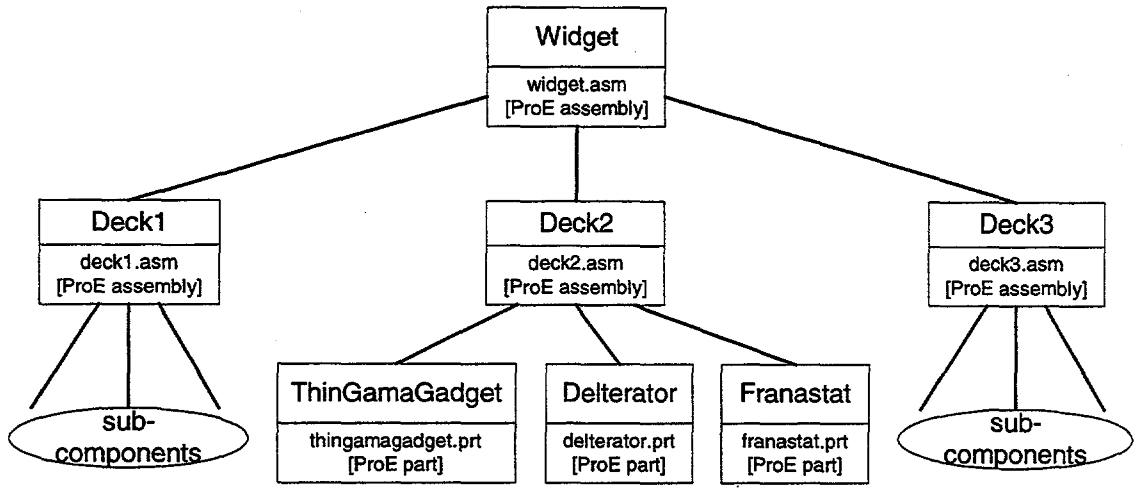

Figure 3. An Example of the Design Solid Structure

The file structure for the assemblies and part files from Figure 3 might look like this:

```
Widget.asm [Pro/E assembly]
     Deck1.asm [Pro/E assembly]
          <...sub-components...>
     Deck2.asm [Pro/E assembly]
          Thingamagadget.prt [Pro/E part]
          Delterator.prt [Pro/E part]
          Franastat.prt [Pro/E part]
     Deck3.asm [Pro/E assembly]
          <...sub-components...>
```

Thus, the *Widget* is composed of three components: Deck1, Deck2, and Deck3, each of which is assembled from sub-components. Deck2, for example, consists of three parts: a Thingamagadget, a Delterator, and a Franastat.

Additionally, each assembly embodies coordinate information and a corresponding transformation formula for its child assemblies and parts. The Franastat part, for instance, might have its origin at one of the Deck2 corners. Thus, the Deck2 assembly must describe how the Franastat part is positioned in the assembly. Analogously, the *Widget* assembly describes how Deck1, Deck2, and Deck3 assemblies are positioned within the *Widget*.

2.2. Analysis Solid Geometry

The Analysis Solid Geometry (ASG) is a simplification of the DSG, where an analyst creates various ASGs depending on the study he/she is interested in analyzing. Typically, the analyst first makes a copy of the DSG, and then uses a CAD package to manually simplify the ASGs.

Figure 4 shows that the ASGs (filled rectangle boxes) can be conceptually created at the part or assembly level of the DSG depending upon the objectives of the analysis.

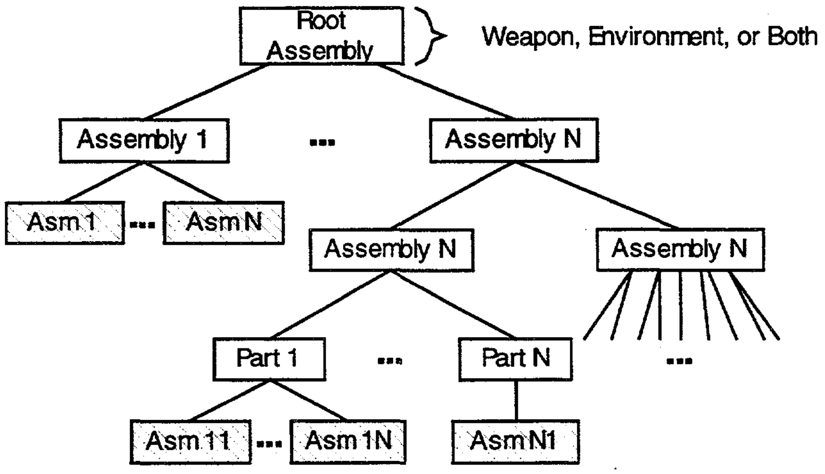

Figure 4. Analysis Solid Geometry Structure

The following subsections demonstrate how ASGs are created for the parts and assemblies using the *Widget* example discussed in Section 2.1.

2.2.1. ASG for Design Solid Geometry Parts

First, consider an analysis of the Franastat part at the leaf of the *Widget* tree structure. A Franastat holds a gas at high pressure, and a typical study of this part may include a pressure analysis or safety factor assessment. The creation of the ASG involves modifications (generally simplifications) to the copy of the corresponding DSG. These changes usually make the solid easier to mesh while preserving (in the analyst's opinion) the original relevant physics. For example, the analyst may remove the threads from a bolt (or remove the fillets from the anterior grommet brackets).

The work results in the creation of a new ASG associated conceptually with the original DSG for the Franastat. Thus, from the tree-like perspective, an ASG is the exact copy of the DSG with the particular part modified. Within the *Widget* tree structure, as shown below, the new ASG (Franstat-ASG.prt) is a simplification of the Franastat.prt.

```
Analysis Solid Geometry Structure           Design Solid Geometry Structure

Deck2.asm [Pro/E assembly]                  Deck2.asm [Pro/E assembly]
   Thingamagadget.prt [Pro/E part]            Thingamagadget.prt [Pro/E part]
   Delterator.prt [Pro/E part]                Delterator.prt [Pro/E part]
   Franastat-ASG.prt [Pro/E part]      ⇦      Franastat.prt [Pro/E part]
```

2.2.2. ASG for Design Solid Geometry Assemblies

Analysis Solid Geometry at the assembly level (i.e., higher up in the tree than parts) are often even coarser approximations of the underlying DSG. For example, a study of some aspect of the *Widget* as a whole might view it as simply an assembly of Deck1, Deck2, and Deck3, each of which is a "composite part". That is, the assembly for Deck2 in this scenario might be approximated as a single homogeneous part, discarding the interior structure of its Thingamagadget, Delterator, and Franastat subassemblies. The material properties (density, etc.) of this approximation of Deck2 are selected accordingly, and may not match any single real material. Using these ASG "composite parts" for the three Decks, the ASG for the *Widget* would be an assembly of the three parts.

This, generally, is the level of detail at which the analysts currently work. Studying the system as a whole, with approximations for the subassemblies fairly high in the tree, is demonstrated in the example below:

***Widget* Analysis Solid Geometry Structure**		***Widget* Design Solid Geometry Structure**
Widget-ASG.asm [Pro/E assembly]		Widget.asm [Pro/E assembly]
Deck1-ASG.prt [Pro/E composite part]	⇦	Deck1 [Pro/E assembly]
Deck2-ASG.prt [Pro/E composite part]	⇦	Deck2 [Pro/E assembly]
Deck3-ASG.prt [Pro/E composite part]	⇦	Deck3 [Pro/E assembly]

In the example above, the Pro/E assembly file *Widget-ASG.asm* is a Pro/E assembly file describing the analysis solid geometry. This ASG is composed of the three part files, one for each Deck.

2.3. Meshes

As depicted in Figure 1, after generating the ASG, the result is meshed, resulting in (for example) an ExodusII file (Data Models and Format in the near future). Conceptually, this mesh is derived from the corresponding solid model. For example, the three decks from the analysis solid geometry example above could be meshed as shown below:

***Widget* (Meshed) Model**		***Widget* Analysis Solid Geometry Structure**
Widget-ASG.asm [Pro/E assembly]		Widget-ASG.asm [Pro/E assembly]
Deck1-ASG.ex2 [ExodusII meshed part]	⇦	Deck1-ASG.prt [Pro/E composite part
Deck2-ASG.ex2 [ExodusII meshed part]	⇦	Deck2-ASG.prt [Pro/E composite part]
Deck3-ASG.ex2 [ExodusII meshed part]	⇦	Deck3-ASG.prt [Pro/E composite part]

The three newly added meshes above are ExodusII files and are denoted as *Meshed Part.*

Alternatively, a mesh is created directly with a mesh generation tool, skipping ASG creation in a solid modeling tool. This may happen when no corresponding DSG exists or when it is more convenient.

Currently, the analyst uses tools such as GREPOS [4] and others to position meshes in the correct position relative to the global coordinate system (e.g., the nose of the weapon). Thus far, the transformation has been relatively easy because the parts were created in their assembled location. Most of the difficult work was done when the ASG's were created where the designer/analyst must manipulate the parts into the global coordinate system. In general, when the parts are meshed they are not necessary transformed or located in the global coordinate system.

Meshed parts can be combined with other meshed parts, subassemblies, and/or auxiliary parts (e.g., joint part) tó create an assembly that can be used in the formation of a system or environment model.

Meshed parts are typically represented with large binary files, ranging from a few megabytes to several gigabytes. In fact, there is no theoretical upper limit for the size of meshed parts. Currently, the upper limit is constrained by the tools used in the simulation processes (e.g., GJOIN).

2.4. Materials

Material information is represented as: 1) Material Name, 2) Material Model, and 3) Material Property Data. The Material Name is an ASCII name that uniquely identifies the material from the other materials in the input deck.

The Material Model is an ASCII name that specifies the model used to describe the behavior of the material. Some structural analysis examples are *Rigid, Elastic, Elastic Plastic, EP Power Hardening, Johnson Cook, Sandia Damage, PLH Strength, Viscoplastic, Hydro, Low Density Foam, Power Law Viscoplastic, Soil N Foams, EP Temp Depend, EP Hydrodynamics, Hyperelastic, Thorne Damage, Thermoelastic, Wire Mesh, BCJ, Orthotropic Crush, New Foam,* and *Power Law Viscoplastic.*

Material Property Data is an ASCII name that specifies the attributes of the material. Some examples of material properties are *Density, Specific Heat,* (Thermal) *Conductivity, Tensor Rotation, Enthalpy,* (Radiative) *Emissivity, Mass, Inertia* (*momentum* and velocity *product*), *Center of Gravity, Elastic Modulus, Poisson's Ratio, Shear Modulus, Stress Strain Data, Hardening Modulus, Yield Stress, Ultimate Stress, Latent Heat of Fusion, Solidus Temperature, Liquidus Temperature,* and *Convective Heat Transfer Coefficents.*

The choice of which Material Model and Material Property Data to use depends on the study of interest and on the analysis code.

The assignment of the materials to the parts can happen at various points in the pre-processing phase. With the creation of the Analysis Solid Geometry, if the material information is known, the designer can assign the material information to the parts. It is also possible for the mesher (i.e., person doing the meshing) to assign a generic material identity to a meshed part (represented as an ExodusII or DMF file). In an ExodusII file, the materials are defined and associated with blocks of elements and are specified by block numbers in the *input deck* (see Section 2.12) when it is created.

With the PRONTO3D analysis code, material and block IDs are the same, but this is not the case with all analysis codes. For example, with SALINAS, material ID is separated from block ID, allowing one material to be assigned to multiple blocks.

2.5. Boundary Conditions

With *boundary conditions*, the analyst can specify the nodes to be either rigidly fixed in space and time or to be defined to move in a specified time-dependent manner. In general, the analyst specifies the node sets with unique identifications within the finite element model. Complex boundary conditions may require subroutines that describe the functional behavior of the boundary conditions.

In PRONTO3D, a boundary condition begins with an assignment of a unique identification number to a set of nodes or element faces in the meshed part or assembly (ExodusII or DMF file). When the input deck (see Section 2.12) is created, the boundary condition specification is defined using the identification number in the system or environment model. Other examples of thermal boundary condition types include *temperature*, *heat source*, *heat flux*, *convective heat flux*, *radiative heat flux*, and *contact or gap heat transfer*.

2.6. Initial Conditions

Initial conditions define the environment at the start of a simulation. For example, with lay-down bomb analysis, the initial conditions for the system model's nodes may be velocity, and the initial conditions for the target model could be zero initial velocity. Other examples include temperature, stress, velocity, and acceleration.

An initial condition definition is a two-phase process. First, a unique identification number is assigned to a set of nodes (Node Set ID) in the system or environment model (ExodusII or DMF file). Second, when the input deck (see Section 2.12) is created, the initial condition specification is defined using the identification number (Node Set ID) in the system or environmental model. In PRONTO3D, the initial conditions can also be read in from a Restart File (see Section 2.19) possibly created by the same or another analysis code (e.g., JAS).

2.7. Loads

A *load* is an external environment applied to the system model and is defined in a manner similar to initial conditions. First, within the system or environment model (ExodusII or DMF file), a unique identification number is assigned to a set of nodes, then the load specification is defined in the input deck (see Section 2.12) using the defined identification number.

The analysis code typical supports the ability to apply a variety of time-dependent and/or time-constant loads to a finite element model. Examples of mechanical loads can be point loads, surface pressures, or body forces (arising from acceleration or electromagnetic fields). Pressure loads are applied to subsets of surfaces using the Side Sets, and analogously points and force loads are applied at nodes defined by Node Sets.

2.8. Element Properties

Elements are entities used to discretize the solid model into meshed parts/assemblies. Each element consists of an associated list of nodes and an ordering of those nodes. The nodes have a location property (i.e., coordinates). The element type is a geometric shape (e.g., hex, quad, beam) used to mesh the parts. The *element properties* are attributes that further characterize each element of a particular type. For example, a spherical element has a radius element property, while a shell element has a thickness element property. Some element types may have multiple element properties. For instance, a three-dimensional beam element may have eight element properties, which are cross-sectional area, torsional moment of inertia, thickness, bending moment of inertia affects displacements in the current plane, bending moment of inertia affects bending out of the current plane, and vectors (x, y, z) that taken together with axis of the element defines a plane for the beam.

There are two ways to specify element properties. One option is to define the element properties within the system or environment model (ExodusII or DFM file). The ExodusII or DMF representation supports definition of element properties for each element. Another option is to define the element properties in the input deck (see Section 2.12). Most analysis codes supply a command for defining element properties, where this definition is specified in the input deck. For PRONTO3D, the command takes Block ID. However, most other analysis codes support finer-grain definitions of the element properties and allow the analysts to specify Block ID, Material ID, and Property ID. The latter option has a higher priority then the former option. If the latter option is specified, it will override the former option. If the element properties are not defined in the input deck, then it will use the definition in the system or environment model (if it exists).

2.9. Contact Surfaces

Some analysis codes provide a mechanism (*contact surfaces*) to model the behavior when surfaces (external element faces) are in contact. The contact surfaces can be paired side set contact, global contact, and "fixed" contacts. For paired side set contact cases, the contact definition is enforced between the two surfaces. These surfaces may have master, slave, or symmetric relationship. The global contact condition is enforced between a surface contacting itself (i.e., crushing or buckling deformation) and another in the surface list, which is automatically populated with external surfaces that contact. The "fixed" contact surface allows for parts of the structure to be very finely modeled to obtain the required resolution. More specifically, it allows the analysts to fix two surfaces in position and instructs the analysis code to ignore some small gap between the surfaces.

The *contact surfaces* are defined as ASCII strings that are specified in the input deck (see Section 2.12).

2.10. Other Input Deck Attributes

Typically, boundary conditions, initial conditions, loads, and materials are the minimum input data for most analysis code. However, this is not always true for every analysis code. Moreover, different analysis codes require different sets of attribute specifications in the input deck. Thus, depending on the analysis code, other attributes may be required.

2.11. (System, Environment, & Simulation) Finite Element Models

A *finite element model* consists of an assembly of meshed parts/subassemblies with its input data, and is created to perform a particular type of study. A finite element model is in a condition such that it is ready to be submitted to an analysis code for simulation. There is always a relationship between the finite element model and the corresponding analysis solid geometry or other design definitions.

There are three types of a finite element model: System Finite Element Model (or System Model), Environment Finite Element Model (or Environment Model), and Simulation Finite Element Model (or Simulation Model). The *System Model* is the finite element model that represents/approximates a physical system, such as the nose, tail, or body of the weapon. Similarly, the *Environment Model* is the finite element model that represents the environment of the system (e.g., the target for the weapon nose, and the aircraft arm, which holds the weapon). The possible operational environments in which the system might be utilized in are specified in the Stockpile to Target Sequence (STS) document. The *Simulation Model* is the finite element model that represents the combination of both the System and Environment Models. In some cases, the analyst may not be interested in studying the system combined with its environment. In this situation, the Simulation Model is the System Model.

2.12. Input Decks

Each meshed assembly that is ready for simulation submission has an associated *input deck*, which contains all of the required input data to execute a particular analysis code. Specifically, the input deck may include material, properties, boundary conditions, initial conditions, loads, contact surfaces, and element properties. For most analysis codes, the minimum information stored in the input deck is the materials, properties and boundary conditions. Depending on the analysis code, other information may be required or included in the input deck. Moreover, different analysis code uses different methods to represent the input deck. For example, PRONTO3D represents the input deck as an ASCII file, while PATRAN stores the input data and the integrated meshes in one (binary) file.

2.13. Simulation Parameters

Before a simulation model is submitted for processing, the *simulation parameters* can be specified to parameterize the simulating study. The parametric data includes position of the system model with respect to the loads (e.g., force of the wind against the weapon or the weapon's angle from the ground) and the position of the system with respect to the target.

Other examples of simulation parameters include the equation solver selection, the time-step size, the output variables and output frequency, and restart file information.

2.14. Integration Tool (Makefile)

The *integration tool* provides the capabilities to create the finite element models (integrated meshes and the associated input deck). The most common realization of this is a makefile. The B61 analysis group uses a makefile commonly referred to, in this report, as Howard's Makefile. The meshed parts used in the creation of the finite element models reside in the file structure (tree-like) and their location is specified by the analysis solid geometry assembly structure. There is a file directory structure for the system model, and a different file directory structure for the environment finite element model. An implementation of the integration tool would invoke the necessary software tools (e.g., GROPE, GINPUT, GREPOS, GJOIN, APREPRO, NUMBERS, etc.) to accomplish the objective. The inputs required for creating the input decks includes material properties, boundary conditions, initial conditions, loads, element properties, contacts, and other attributes that the selected analysis code may require. The integration tool is also responsible for combining the system and environment finite element models to create the simulation finite element model. This also supports manual specification of parameter information for the simulation model. The parametric information (simulation parameters) includes the position of the system with respect to the loads (e.g., force of the wind and angle of the weapon), and the position of the system with respect to the target.

2.15. Analysis Codes

Analysis codes are application (modeling) software packages that support various analysis studies. For example, PRONTO3D is a three-dimensional, transient, solid dynamics code for analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates. Some analysis codes commonly used at Sandia include COYOTE, PRONTO3D, JAS, ALEGRA, and Sierra (i.e., codes such as Presto instantiated within the Sierra framework).

2.16. General Output File

The *general output file* is a product of an executing analysis code. The general output file is a text summary of the simulating study that includes (for example) an echo of the input commands (specified in the input deck), useful derived constants, any error messages that might be generated from the code, and general informational messages.

PRONTO3D analysis code, the general output file generated for a *beam* study is *beam.o*.

2.17. History File

During the execution of some analysis code, the analyst may want to monitor or analyze results for a select few nodes and elements for each execution time step. These requested results are stored in the *history file*. A list of possible results that the analyst may be interested in include velocity, force, acceleration, displacements, coordinates, nodal mass, reactions, current position, and temperature. Typically, the history file includes a subset of the output results.

In the case of PRONTO3D, the history file stores results in ExodusII format and has a *.h* file extension. For example, the history file generated by the PRONTO3D for a *beam* study is *beam.h*.

NOTE: COYOTE does not support the concept of a history file.

2.18. Plot File

The analyst takes the results from the simulation run and interprets them to help make engineering decisions. Commonly, visualization tools are used to aid in understanding the simulation results. Input to the visualization tool is the *plot file*, which is produced by the analysis code. The plot file is in (for example) an ExodusII (or DMF) format and consists of the assembled meshes submitted for simulation and the simulation results. The plot file generated by PRONTO3D for a *beam* study is *beam.e*.

To keep plot files to a reasonable size, simulation results are typically not written for every solution time step but rather at some user specified interval (e.g., plot step might be 10^{-3} or 10^{-4} where the solution time step might be 10^{-7}, 10^{-8}, or smaller).

Currently, visualization is done using a variety of tools such as BLOT [3], MUSTAFA, PATRAN, and ENSIGHT.

2.19. Restart File

Restart file is another output from a simulation. It contains all of the information (nodal and elemental variables) needed to restart the next simulation.

The *restart file* of the PRONTO3D finite element program stores results in the same format as the plot file (see Section 2.18), ExodusII format, and has a *.rsout* file extension. For example, the restart file generated by the PRONTO3D for a *beam* study is *beam.rsout.*

2.20. Test Data

Test data is a text-based report containing (for example) information that describes the results to be expected from (or associated with) a test of a particular study. The information may consist of graphs, tables, and references describing the test results.

2.21. Analysis Report

An *analysis report* is a text-based document that describes the rationale for the current formulation of the finite element model. It also provides more readable information about the model such as the IDs used, the material selected and why, lessons-learned and more. The objective of the analysis report is to provide an analyst that is not familiar with the model, sufficient information to continue the work or to understand the reasons for the work that has been performed to date.

2.22. Assessment Report

The *assessment report* is analogous to the analysis report except it contains information about the assessment performed on a particular simulation result.

2.23. Engineering Drawing

The engineering *drawing* is a graphical specification of the DOE requirement for development of a product. Among other things, it contains the necessary information to create the Design Solid Geometry and some bill-of-material data.

2.24. Summary of Simulation and Analysis Process

Creating a simulation study package for a particular analysis study is a manual, difficult and time-consuming process. One current strategy uses a combination of a compiler *make*, a set of in-house applications that perform specific tasks (e.g., GREPOS, GJOIN, APREPOS, NUMBERS, CUBIT, FASTQ, etc.), some GOTS[1] and COTS[2] tools, and a lot of manual processing and calculations.

[1] Government Off-The-Shelf Software. This includes software developed by government or non-profit organizations.
[2] Commercial Off-The-Shelf Software. This includes all software that can be purchased.

For a particular study, the analyst/designer[3] (use Figure 1 as a guide):

1. Receives a product drawing specification (e.g., B61, W80). The drawing specification is converted into design solid geometry. In some cases, the design solid geometry is stored in the repository (e.g., WTC CMS). From this point forward, changes to the design solid geometry are not reflected throughout the process or more commonly the participants are not aware of the changes. Moreover, changes can occur at any time and may not be automatically passed on to the analysts.

2. Obtains the system and environment design solid geometry from the designer if the analyst is aware of the design definitions. In this case, the design solid model may be sent to the analyst via one of the traditional approaches (e.g., email, postal mail, or hand-carry). The analyst may also retrieve the design solid geometry from the repository (e.g., WTC CMS). With the latter approach, the analyst may not have been aware of design modifications and may have to do some manual querying to determine/identify those changes.

3. Studies and visualizes the system and environment design solid geometry and creates the analysis solid geometry accordingly. If the design solid geometry was created using the CAD package's layering capabilities, then the analyst/designer would use the design solid geometry as the base for the analysis solid geometry. The analyst/designer would remove the unnecessary layers and create only what is needed. However, this is not the normal operation. The analyst/designer typically has to create the analysis solid geometry from scratch. The design solid geometry is only used to determine what analysis solid geometry is needed. At this point, if the materials/properties are known, the designer and analyst may assign them to the analysis solid geometry. In general, creating the analysis solid geometry is a joint effort between the analyst and the designer. The analyst studies the design solid geometry and advises the designer in creating the analysis solid geometry.

4. Determines which analysis code to use because this dictates much of what is done in the way of mesh simplifications, contact definitions, and so on.

5. Studies and visualizes the analysis solid geometry and determines the portion to mesh. With the current mesh generation tools, the analyst may have to recreate or decompose the design solid geometry definition. Some analysis studies (e.g., thermal) require homogeneity across different materials and parts, and therefore, a solid model is typically meshed into one discretized part (mesh).

6. Meshes the geometric parts/assemblies in the analysis solid geometry into meshed parts/assemblies. At this point, a generic unique material identification is assigned to the meshed parts/assemblies. During the creation of the input deck, this generic material identification is assigned to the 'real' material.

[3] An analyst may not perform all of the steps outlined. For example, one analyst may work with the designer to mesh analysis solid geometry, while a different analyst may use these meshes to create the simulation model.

7. Manually creates the file structures to match the tree-like structure captured in the system and environment design solid geometry. Initially this may be done on a disk local on one machine. If a disk failure occurs or analysis is interrupted, much of the work created thus far may be lost. Depending on the analyst ability to backup and recover, some of the data can be recovered from backup copies
8. Copies meshed parts and assemblies to the appropriate file locations of the working machine, which may be a server or a machine in some analyst's office.
9. Manually calculates the coordinate offset of each meshed part to the actual weapon system.
10. Manually determines which meshes may have identical IDs (node #, block #, element #, side set / node set #) that need to be renumbered, and invokes a tool (e.g., GREPOS) to renumber the IDs so all meshes can be joined without ID name collision.
11. Creates the input deck by deterring the boundary conditions, materials/properties, initial conditions, loads, contacts, element properties, data to parameterize this particular simulation study, and any other input data that the selected analysis code may require. The complexity and time-consuming nature of this step depends on how much information is given in the analysis solid geometry's bill-of-materials (BOM).
12. Creates an integrated build tool (typically a makefile), which allows the analyst to specify which software to invoke, with which dependencies, and in what order. In the case of a makefile, it also requires the analyst to specify the input data and indicates if this particular simulation study requires combining the system and environment models to create the simulation model. The result generated from the build tool is a simulation model ready for submission to the analysis code. NOTE: this step is not applicable if the part meshes are contiguous.
13. If necessary, modify the analysis code to work for the particular simulation study.

Now, once an integrated build tool (makefile) is executed to create the analysis code inputs, and the inputs are submitted for processing. The results of the simulation can then be analyzed, for example using a visualization tool. The analysis results are used to improve the next simulation study. This process is repeated until the simulation study of a particular part of a system or the entire system is completed. In some cases, the simulation results (submitted simulation finite element model plus visualizing data) are used in the next simulation study (commonly referred to as sequential simulation study).

After the simulation study is complete, the analyst determines the set of artifacts to archive for recreation of the study, reuse in future programs, or for educational/training purposes. Currently, the determination as to which artifacts should be archived varies between analysts.

3. Problem Statement

Below is a summary of the problem statement derived from an interview with the B61, W80, W76, and W88 representatives.

3.1. Management of Simulation Studies

As indicated in Section 2.24, creation of a *Simulation Study* is a manual, difficult, and time-consuming process. In addition, changes to the input or intermediate artifacts (e.g., meshed parts, assemblies, makefiles, or script files) result in recreating the meshed model relationship tree structure and reprocessing steps in the procedure. Thus, the analyst needs/requires the procedure to create and manage the simulation study automated. This includes:

- Automation of all procedures or sub-procedures, from retrieving the design solid geometry to generation of the simulation study. This includes creation and management of the finite element models (meshed parts/assemblies and their associated input decks).
- Management and persistence storage of all intermediate artifacts (e.g., meshes, material/properties, boundary conditions, initial conditions, loads, engineering notebooks, analyst workflow, project workflow, model relationship, mesh relationship (mesh recipe), input deck relationship (input deck recipe), simulation study relationship (simulation study recipe), simulation outputs (plot files, history files, general output files)).
- Tracking all software tool versions and their dependencies.
- Integration of mesh generation and meshing tools, and a mechanism for interoperating (at the function and data level) with other tools. This implies the integration of tools (e.g., CAD, meshing, and data management) necessary to manage analysis products (e.g., meshes, mesh recipes, and simulation studies).

3.2. Automated Design-to-Analysis Workflows

The process outlined in Section 2.24 is what is typically done from design-to-analysis of the weapons. This is a somewhat manual process and results in input creation work being duplicated and the potential for lost information at each incremental step. For instance, the analysis solid geometries are typically drawn from scratch and the design solid geometries are only used manually to determine what to draw in the analysis solid geometries. In addition, the coordinate information from the design solid geometries is not kept and thus the analyst must manually track the information. Thus, this is a very labor-intensive and error-prone process. The analyst needs a more integrated environment that allows products from the previous increment to be directly usable in the current increment, and so on. As products progress through the life cycle, the analyst would like the mechanism to annotate the products, as well as capturing any design/analysis intent or knowledge gained. More importantly, the analyst needs to be able to capture the current working environment and preserve it to be completed or reused in the future.

3.3. Coordinate System Management

The Pro/E CAD package (and maybe other drawing packages) has the capability to calculate the coordinate system of various parts and how they are positioned relative to each other to create the geometric model. By the time the model is meshed, this geometric coordinate information is no longer available to associate with the mesh. Since meshes use a local coordinate system, composition of meshes with the specification of the necessary input data required to create a finite element model is very difficult.

The current, manual solution is to do all the meshing in the global coordinate system. Thus, no repositioning of meshes is necessary during assembly. Although this approach works, this method offers no flexibility with the assembled mesh. Indeed, there are some future studies planned in which this approach will not work. Improved coordinate system management will be essential for future work.

The analysts would also like the capability to capture this geometric coordinate information from the design solid geometry so that meshed parts can be assembled in any coordinate system.

3.4. Archiving of Models and Processes

The analyst would like a set of services that allows them to provide archiving of the artifacts listed in Table 1. The objective of archiving is to store the artifacts for future reuse, for recreation of the study, and for educational/training purposes. Thus, archiving is defined as the mechanism for storing, retrieval, and querying the artifacts associated with simulation studies from long term, persistence repository.

As indicated in Section 2.24, the analysts typically do not determine the set of artifacts to archive until they are finished or satisfied with the analysis. In addition, there is usually some assumed hierarchy of the artifacts, thus, the analysts commonly query the archived repository based on this hierarchy. For example, it is understood that a program consists of a group of simulation studies and that each study consists of a set of design analysis.

3.5. Support of Certification and Qualification

As described in Section 2.24, the artifacts may reside on various designer's and analyst's machines. Thus, they are difficult to trace, which leads to a difficult and time-consuming process to certify the system.

The W80 team has selected the system verification process as outlined by Dean and Barrett [2] to facilitate the certification and qualification process. The first step of the approach is to screen the response models, damage modes, and associated STS environments. The results from the screening are documented in a damage-response mode matrix. The intersection of the damage and response mode is a potential vulnerability issue; each is ordered and screened according to some calculated margin. In addition, the interaction is also used to uniquely identify a set of verification activities to further refine the response, damage, and failure level. Next, the approach calls for identifying and correlating verification activities with the requirements and is documented in the Verification-Compliance Matrix. Each intersection is indicated with a status such as task not started, task underway, data obtained, and test complete. A more complete description of the verification activities is described in the Verification Activities Matrix. The matrix identifies the tasks for the verification activities.

The designers and analysts, particularly representative from the W80 program, would like the capabilities to support the system certification process similar to the process described above.

3.6. Version Control of Design Solid Geometries

A given design solid geometry is used as the starting input product to the simulation-based process. Much effort and capitol is spent in developing the necessary artifacts for simulating and analyzing the design solid geometry. Thus, depending on the state of the simulation model, changes to the design solid geometry or attributes that effect the design solid geometry can greatly affect the analyst's modeling efforts. Thus, the customers (designers) would like the capability to control design changes. Changes may be allowed based on authorship, the progress of the simulation and analysis, the effect they have on the rest of the artifacts in the process, and within an "changeable" range.

4. Assumptions, Dependencies, and Constraints

1. From the Customers Problem Statement given in Section 3.1, SIENA must provide an open interface for interacting with the in-house (e.g., GREPOS, GJOIN, BLOT, CUBIT, FASTQ, etc.), GOTS, and COTS (e.g., IDEAS, COSMOS, Pro/E) tools. The interaction includes invoking the external tool functions, as well as being able to interpret the tool's data (to some specific level of understanding).
2. SIENA must be accredited for classified computing.
3. The SIENA project will use an incremental and iterative software development methodology (e.g., Rationale Unified Process). This implies that we will deliver SIENA framework services in increments. Perhaps, one service capability at a time that will allow the customers to quickly evaluate the capabilities and provide the necessary feedback. This iterative relationship will continue until the product meets the customer's requirements. Each increment needs to be deployable for classified computing.
4. All user interfaces must have both graphical and command-line support. Graphical support includes web-based support.
5. Design Solid Geometries may be produced by any CAD drawing tools. However, since there is a tri-lab directive to use Pro/E as the CAD drawing package, there must be deep integration with Pro/E, and shallow integration with the other CAD packages.
6. Design Solid Geometries, Analysis Solid Geometries, and artifact of the simulation studies and analysis may be stored and retrieved from any of a variety of corporate configuration management tools.
7. SIENA should support capability extensibility.
8. SIENA must provide a deployment and maintenance strategy.
9. SIENA must provide user-friendly online documentation as well as more traditional documentation and help.

5. Requirements

This section describes high-level requirements that were derived from the Customers Problem Statement (Section 3).

The priorities in the table below were assigned by our customers. The ones marked with P<number> are given the highest priority, which implies that they are of paramount importance to quickly and greatly improve the customer productivity and reduce the customer's time spent setting up the problems for simulation. The requirements marked with S<number> are secondary and means the customers would like to see them realized with the primary requirements, but they would be satisfied if these are delivered later in the following increments. The rest of the requirements are of less immediate important to our customers and are marked with N<number>.

P1.	Generic Representation of the Artifacts Each mesh, mesh recipe, simulation study recipe, input deck, input deck recipe, and other artifacts of SIENA (see Figure 1) may be represented by a number of data representation (e.g., a DSG may be represented in Pro/E or SolidWorks format). In addition, these artifacts are associated with large quantities of information. Most of the information is currently known to our customers, but they are expecting that additional information may be needed in the future. Thus, the customers requested that the mechanism used to represent these artifacts be an abstract representation (e.g., component). SIENA must provide plain text representations of these components so that they can be edited outside SIENA using any plain-text editor. In addition, SIENA must also provide graphical and other means (via an editor) of accessing the components.
P2.	Management of Meshes SIENA must provide the analyst with the capability to integrate (contiguous) meshes (add, remove, transform, import, and export meshes). This includes management of the properties associated with the meshes and mesh versions. Mesh properties include management of the various IDs (node set, side set, block, element, the mesh assembly structure, etc.) needed in creating the simulation model. In short, SIENA needs to automate the portion of Howard's makefile that manages the "tree"-like relationship of meshed parts. SIENA must not enforce any strict procedure for accomplishing a set of tasks during the simulation-based process. For example, in order for the analyst to integrate a set of meshes, SIENA should not first require the analyst to create the design solid geometry or analysis solid geometry. SIENA may display a list of meshes in which it is has knowledge and allow the user to import meshes into the SIENA environment. After the integration of meshes, the analyst should not be required to associate any input data with the mesh subassembly. However, SIENA should give the analyst the option of exiting and indicate that the analyst has now completed the task or proceed to further define the finite element model.

	NOTE: The customer suggested some kind of composition ID structure. For example, the root of the meshed assembly is assigned B61HeadSS1; one of its children nodes is assigned *B61HeadSS1.Asm1*, and so on.
P3.	Management of Materials Currently, the materials are not stored in any single repository and the association of the materials to a simulation study is done by a makefile or some in-house integrated tools. There is an effort underway to put all materials into a repository. SIENA must support management of materials and provide the capability for the analyst to associate properties to a mesh or an analysis study. Note: the **Management of Mesh** and **Management of Simulation Study** Sections cover tracking of material assignments to various meshes and analysis studies, respectively.
P4.	Management of Mesh Recipes SIENA must provide a mechanism for the analyst to create, delete, modify, query, configure, store, and retrieve mesh recipes. A mesh recipe is a set of instructions that indicate how submeshes are assembled into a single subassembly of meshes, system models, environment models, or simulation models. In addition, a mesh recipe is a set of instructions for translating and mirroring of meshes. A mesh recipe may have multiple versions. A possible creation of a new version of a mesh recipe is when a new version of submesh or any of the properties have been introduced or substituted. SIENA must also provide the analyst with the option to explicitly create new versions or to commit the changes to the current version. In addition, SIENA should allow the analyst to delete or modify versions. In short, a mechanism is needed that automates the portion of the integrated build tool (e.g., Howard's makefile) that creates mesh recipes.
P5.	Management of Input Decks SIENA must support creation, deletion, modification, querying, configuring, storing, and retrieving of input decks. This includes the management of boundary conditions, initial conditions, loads, contacts, element properties, simulation model parameterizing data, and any other input deck attributes required to create the input deck for simulation studies. See Sections 2.5, 2.6, 2.7, 2.8, 2.9, 2.10, and 2.13 for descriptions of these attributes.
P6.	Management of Input Deck Recipes SIENA must support creation, deletion, modification, querying, configuring, storing, and retrieving of input deck recipes. There may be other artifacts that go into the input decks in addition to boundary conditions, initial conditions, loads, element attributes, and contacts. This is analysis code dependent. For example, the analysis code PRONTO3D has contact information (e.g., internal boundary conditions) whereas Coyote has convection properties.

P7.	Archiving of Artifacts for Future Use From the Customers Problem Statement given in Section 3.4, SIENA must provide the capability to archive the final study and associated analysis intent for future recreation and reuse. SIENA must provide the mechanism to store and retrieve the minimum set of artifacts needed to recreate the simulation study as well as adding to or deleting from the minimum set. They are the Design Solid Geometry, Analysis Solid Geometry, Meshed Parts, Simulation Finite Element Model, Simulation Outputs, Materials, Input Deck, and Test Data. Also, SIENA must provide the capability for the user to query for any artifacts stored in the long term, persistence repository. The querying statement can be composed of any metadata/data captured by the artifacts. For example, the user can ask SIENA for a list of simulation studies submitted to PRONTO3D. The simulation studies are the artifacts and the analysis code name "PRONTO3D" is an attribute of the artifacts. For large quantities of hard copy information, an approach that is acceptable to the analysts is to only convert important portions of the information into soft copy and place it in the SIENA or Archive repository that is accessible to the SIENA framework. For example, for each test data, convert graphs, tables, and references to soft copy and let SIENA manage these metadata that particular test data. SIENA should also consider supporting dynamic schema that is to say, the user can dynamically specify attributes that do not exist in the archive repository's schema. SIENA would dynamically and continuously learn about the interactions with it and dynamically add the attributes to its schema.
S1.	Management of Simulation Studies and Simulation Study Recipes SIENA must support creation, deletion, modification, querying, configuring, storing, and retrieving of simulation studies and simulation study recipes. A simulation study recipe contains the necessary instructions to recreate a particular simulation study.
S2.	Management of Analysis Codes SIENA must provide the mechanism to submit the simulation models to analysis codes for execution and retrieving of the simulation results. In addition, SIENA should support the management of the attributes (e.g., remote links, and names) which describe the analysis codes. NOTE: SIENA may use the capabilities of Distributed Resource Management (DRM), as appropriate, to accomplish this requirement.
S3.	Management of Software Tool Versions SIENA must track the associated software tools, their versions, their builds, and associated dependencies (e.g., libraries used).
S4.	Coordinate System Management and Manipulation This was derived from the problem statement in Section 3.3. Partly, this is being

	addressed in part by the GJOIN [5] rewrite, which is currently under development. Some aspects of geometry management are not currently being addressed. Specifically, in the normal case, much of the transformation description (or global geometry description) is available in the associated Pro/E assembly files, but no software tool is available to make the global part coordinates accessible to SIENA. Nevertheless, providing convenient access to these data is something that the analysts consider essential.
S5.	Automating Design-to-Analysis Workflow Derived from the Customers Problem Statement given in Section 3.2, SIENA shall provide an environment that incrementally mends the various disconnects that occur in the process (e.g., between design solid geometry and analysis solid geometry, loss of material properties, etc.). The environment needs to also support collaboration among various agents (e.g., analysts, designers, and engineers) in the workflow. For example, when there is a change to a design solid geometry, all parties involved would be notified automatically. In addition, the environment shall allow the users to track their daily progress. Thus, if the analyst cannot continue working on a particular study, he/she can archive the particular daily environment and complete the study in the future.
N1	Automated Engineering Notebook Provide software support for engineering notebooks. Each engineer is assigned a notebook and a project notebook consists of project related information (e.g., project schedule, project metrics, project defects, etc.) and all the notebooks belonging to all engineers working on the project. The engineers should be able to use any ASCII editors and the most common editors, word processors, a diary, and still be able to have the information imported into SIENA's engineering notebook. Each engineering notebook has an associated access control list. The engineer has the option to make portions of the notebook private. In this case, only the owner of the notebook has access to the private sections of the notebook.
N2	Verification and Validation of Finite Element Models SIENA must provide access to a capability that determines if the finite element model was correctly integrated for a set of meshed parts/assemblies by determining whether the integrated meshes geometrically matches its analysis solid geometry.
N3	Mesh Collision Detection SIENA must provide access to a capability that determines if there are any meshes within a finite element model that overlap or collide. NOTE: VERDE provides various capabilities similar to the request stated in this requirement. Thus, SIENA may be able to take advantage of VERDE to address this requirement.
N4	Open Interface to Extend Capabilities In our discussions for automating/helping/keeping-track-of various things, many

	details kept arising that hadn't been mentioned previously (block ID, node-sets, etc.). SIENA must provide an open interface for the analyst to add capabilities that may be overlooked at this time or may need to be added in the future.
N5	Support of Engineer and Project Workflows SIENA should support creation, deletion, modification, querying, configuring, storing, and retrieving of engineering and project workflows. SIENA should provide a set of default templates for engineering and project workflows that are tailorable to fit the individual's and project's needs. Each engineer and analyst has an associated workflow that indicates how they would like to develop, simulate, and analyze studies, which is referred to as *engineer workflow*. Similarly, each project has an associated workflow, which indicates the tasks, input products, output products, dependencies, and tools of the project. The *project workflow* consists of the engineer's workflows plus tasks that are specific to the project. NOTE: There are software COTS that support automated workflows, and SIENA may take advantage an appropriate one to support this requirement.
N6	Support for Certification Process SIENA shall support the system certification process similar to the Dean and Barrett [2] as summarized in Section 3.5. In general, SIENA shall provide the mechanism to trace the activities of the weapon system designs. Each activity, its artifacts, tools, objectives, reasons, and performing agents must be traceable and identifiable.
N7	Version Control of Design Solid Geometry SIENA must provide the mechanism to control changes made to the design solid geometry or attributes that effect the design solid geometry as describe in Section 3.5.

6. References

[1] S. W. Attaway et al., *PRONTO3D User's Instructions: A Transient Dynamic Code for Nonlinear Structural Analysis*, Sand Report, SAND98-1361, Sandia National Laboratories, June 1998.

[2] F. Dean and W. Barrett, *Transitioning from Test-based to Science-based Certification for Hostile Stockpile-to-Target Sequence Environments*, Sand Report, SAND99-0677, Sandia National Laboratories, July 1999.

[3] A. P. Gilkey and J. H. Glick, *BLOT – A Mesh and Curve Plot Program for the Output of a Finite Element Analysis*, Sand Report, SAND88-1432, Sandia National Laboratories, June 1989, Available at http://sass2248.jal.sandia.gov/SEACAS/Documentation/Blot.pdf.

[4] G. D. Sjaardema, *GREPOS: A Genesis Database Repositioning Program*, Sand Report, SAND 90-0566, Engineering and Manufacturing Mechanics, Sandia National Laboratories, Last Modified on 5/20/89, Available at http://sass2248.jal.sandia.gov/SEACAS/Documentation/grepos.pdf.

[5] Sandia National Laboratories, Cross-Cut Integration Project, *GJOIN: A Program for Merging Two or More Genesis Database*, Version 2.0, available at http://z.ca.sandia.gov/~santa/xcut/commands.txt.

UNLIMITED RELEASE

INITIAL DISTRIBUTION:

1	MS 9003	K. E. Washington, 8900
1	MS 9011	P. W. Dean, 8903
1	MS 9011	B. V. Hess, 8910
1	MS 9011	P. E. Nielan, 8920
1	MS 9012	S. C. Gray, 8930
1	MS 9037	J. C. Berry, 8935
1	MS 9019	B. A. Maxwell, 8945
1	MS 9019	R. Trechter, 8945
1	MS 9217	P. T. Boggs (a), 8950
1	MS 9012	J. A. Friesen (a), 8990
1	MS 9001	M. E. John, 8000
		Attn: R. C. Wayne, 2200
		J. Vitko, 8100
		W. J. McLean, 8300
		D. Henson, 8400
		P. N. Smith, 8500
		T. M. Dyer, 8700
3	MS 9018	Central Technical Files, 8940-2
1	MS 0899	Technical Library, 4916
1	MS 9021	Technical Communications Dept. 8815/Technical Library, 4916
1	MS 9021	Technical Communications Dept. 8815 for DOE/OSTI
10	MS 0847	C. R. Adams, 9125
5	MS 9217	B. A. Allan, 8920
5	MS 9217	R. C. Armstrong, 8920
1	MS 0557	T. J. Baca, 9125
1	MS 0986	R. S. Berg, 02662
1	MS 0841	T. C. Bickel, 9100
5	MS 0828	B. D. Boughton, 9132
1	MS 0847	S. N. Burchett, 9132
5	MS 9003	D. L. Crawford, 9900
5	MS 9042	J. J. Dike, 8727
1	MS 0986	G. J. Dodrill, 02662
1	MS 0628	P. Erickson, 2996
1	MS 0555	M. S. Garrett, 9122
1	MS 0627	L. Grube, 2991
5	MS 0321	A. L. Hale, 9220
1	MS 1137	A. L. Hodges, 6534
5	MS 9014	E. L. Hoffman, 2267
5	MS 0836	R. E. Hogan, 9117
1	MS 0847	J. Jung, 9132
1	MS 0835	S. N. Kempka, 9111

5	MS 9217	M. L. Koszykowski, 8950
5	MS 9014	C. A. LeGall, 2267
1	MS 0847	R. W. Leland, 9226
1	MS 1138	P. J. Lewis, 6531
5	MS 9217	R. Mariano, 8910
1	MS 0847	D. R. Martinez, 9124
1	MS 1113	R. A. May, 9126
1	MS 0847	H. S. Morgan, 9123
1	MS 0828	J. Moya, 9132
1	MS 0624	C. Neugabauer, 2994
1	MS 0638	D. E. Peercy, 12326
5	MS 1137	C. A. Poore, 6534
1	MS 0834	A. C. Ratzel, 9112
1	MS 0828	C. A. Romero, 9133
15	MS 0316	L. D. Sauer, 8920
1	MS 0847	G. Sjaardema, 9131
1	MS 0847	T. J. Taugtes, 9226
1	MS 0628	S. Trauth, 2993
1	MS 0469	A. Verardo, 2990
5	MS 0847	H. P. Walther, 9125
1	MS 0847	D. R. White, 9226
5	MS 9012	R. A. Whiteside, 8920
1	MS 0826	J. D. Zepper, 9131

External Copies:

Robert Clay (5)
TeraScale, LLC
1202 Mae Avenue
Tracy, CA 95376