Abstract for 40th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit on Numerical Propulsion System Simulation (NPSS)

The Numerical Propulsion System Simulation (NPSS) is a framework for performing analysis of complex systems. Because the NPSS was developed using the object-oriented paradigm, the resulting architecture is an extensible and flexible framework that is currently being used by a diverse set of participants in government, academia, and the aerospace industry. NPSS is being used by over 15 different institutions to support rockets, hypersonics, power and propulsion, fuel cells, ground based power, and aerospace. Full system-level simulations as well as subsystems may be modeled using NPSS. The NPSS architecture enables the coupling of analyses at various levels of detail, which is called numerical zooming. The middleware used to enable zooming and distributed simulations is the Common Object Request Broker Architecture (CORBA). The NPSS Developer’s Kit offers tools for the developer to generate CORBA-based components and wrap codes. The Developer’s Kit enables distributed multi-fidelity and multi-discipline simulations, preserves proprietary and legacy codes, and facilitates addition of customized codes. The platforms supported are PC, Linux, HP, Sun, and SGI.
Background

Development History

1987 - 1995 Conceptual, Prototypes

1996 NDA, NICE-1, Formal Requirements Definition

June 1997 Space Act Agreement 3-83

July 1997 NCP Beta Release

August 1998 Initial NPSS Release (NCP Version 1)

March 2000 NPSS Version 1.1 - Full 0-D Functionality

March 2002 NPSS Version 1.5 - Initial Zooming, Code Coupling,
Visual Based Syntax (VBS, the GUI), Space Components


September 2003 NPSS Version 1.6 - Enhanced Functionality (based on user feedback)

VBS 1.6 - Enhanced Capabilities (based on user feedback)

CCDK Version 1.0 - CORBA Component Developer’s Kit

(Multi-Fidelity, Multi-Structural Distributed Objects)

NPSS V1.X is an extensible framework for performing analysis of complex systems:

First major propulsion thermal/fluid design and analysis system

Developed using a programming paradigm designed for simulating complex engineering systems (i.e. object-oriented programming)

Extensible Framework

Expand models easily

Build larger models, including more subsystems, in less time

Flexible to model wide variety of complex systems

Add individual or proprietary components, component libraries, and legacy codes

Four different mechanisms to add new components

Interpreted components

Internal components

External components

Dynamically Loadable Modules (DLM) components

Deploy locally/distributed/parallel using high-end computing and communications as required

Uses Common Object Request Broker Architecture (CORBA)

Multi-aware communication

Permits mixture of codes such as C, FORTRAN, and C++

To be collected within any simulation.

Technical Overview

What is NPSS V1.X?

NPSS V1.X is a framework for performing analysis of complex systems.

First major propulsion thermal/fluid design and analysis system developed using a programming paradigm designed for simulating complex engineering systems (i.e. object-oriented programming)

Extensible Framework

Expand models easily

Build larger models, including more subsystems, in less time

Flexible to model wide variety of complex systems

Add individual or proprietary components, component libraries, and legacy codes

Four different mechanisms to add new components

Interpreted components

Internal components

External components

Dynamically Loadable Modules (DLM) components

Deploy locally/distributed/parallel using high-end computing and communications as required

Uses Common Object Request Broker Architecture (CORBA)

Multi-aware communication

Permits mixture of codes such as C, FORTRAN, and C++

To be collected within any simulation.

Technical Overview (continued)

Architecture

PDM Compliant

Security

Interoperability

Simulation Integrations

Connector objects for MD, Coupling objects

Component objects

Visualization objects

Propulsion object API

CORBA wrappers to existing code

Affordable High-Performance Computing

Massively Parallel Supercomputing

Clusters

Network piping

Operating System Level Abstractions

Microsoft Windows, NT, Unix, Linux

GTL, Legion

AIAA/ASME/SAE/ASEE
40th Joint Propulsion Conference
July 13, 2004
Cynthia G. Naiman

NASA
Computing and Interdisciplinary Systems Office
Glenn Research Center

2004 JPC
Technical Overview (continued)

Building An Object-Oriented Model

- NPSS is an object-oriented framework for executing systems of components.
  - No matter what the user needs to simulate, the steps are the same.
- Divide system into discrete components (i.e. elements)
  - The user’s conceptual view of the physical components can be mapped directly onto the object class hierarchy.
  - An object may be one component or an assembly of components.
- Link components
- Setup solver and execution sequence

This same basic recipe is followed to build any NPSS model.

Technical Overview (continued)

Building Blocks of A Model

- Elements
  - Primary building blocks connected together via Ports
  - Perform high level calculations
- Subelements
  - Interchangeable secondary building blocks that plug into Elements or other Subelements
  - Perform detailed calculations
- Flow Stations
  - Responsible for thermodynamic and continuity calculations
  - Access the thermodynamic packages (Janaf, GasTbl, CEA, H2, O2, Combusted H2O, Tabular Data)
- Ports
  - Used to connect Elements together
  - Five types (Mechanical, Fluid, Fuel, Thermal, Data)
  - Directional in nature (i.e., outputs connect to inputs)
- Tables
  - Organized set of numbers that relate n-dimensional inputs to one or more outputs
  - Support linear and second or third order LaGrange interpolation
  - Support fixed value end-points or extrapolation (linear or third order LaGrange)
  - May be used at any location a function is called and vice-versa

All simulations are created from a collection of 5 basic types (classes) of building blocks, which represent engine components, and describe how components are linked together.

Technical Overview (continued)

Running A Model

- Batch
- Interactive
- Graphical

Technical Overview (continued)

Zooming

- NPSS Zooming is the coupling of analyses at various levels of detail.
- Run one or more components at a specified fidelity while the rest of the system-level simulation runs at another fidelity.

Technical Overview (continued)

Zooming

- NPSS Zooming is the coupling of analyses at various levels of detail.
- Run one or more components at a specified fidelity while the rest of the system-level simulation runs at another fidelity.

Summary

- NPSS object-oriented architecture has been proven on a wide variety of applications
- Involving partners throughout the development process has been invaluable and the main reason for success
- Flexible architecture supporting multi-fidelity, multi-discipline components using high-end computing and communications provides excellent candidate to support broader market
- Focus on Technology Transfer will continue
Backup Slides

Incremental Release Process

Original Requirements Specification → Initial Requirements Analysis → Initial Implementation of Architecture (Beta)

Requirements Definition for Version 1 → Initial Requirements Analysis → Assessment & Incorporation of Customer Feedback → Customer Feedback

Plan Subsets of Version 1 per Incremental Releases → Incremental Release

Assess & Incorporate Customer Feedback


Develop Subsets of Version N Requirements → Assessment & Acceptance Reviews

Delivery of Full Version N → Acceptance Review

NPSS V1.X Package Descriptions

NPSS Release Package

NPSS V1.6

Documentation

Visual Based Syntax

Components

Thems

Customer Deck

DevKit

DLLDevKit

Mico tar

Add-On Packages

CCDKBase

CCDKHiFi

CCDKRocketsITAR

Wrapper Tools

CCDK – CORBA Component developer’s kit

HiFi – high fidelity

NPSS Release Package

Visual Based Syntax

Components

Thems

Customer Deck

DevKit

DLLDevKit

Mico tar

Add-On Packages

CCDKBase

CCDKHiFi

CCDKRocketsITAR

Wrapper Tools

NPSS Release Package

Visual Based Syntax

Components

Thems

Customer Deck

DevKit

DLLDevKit

Mico tar

Add-On Packages

CCDKBase

CCDKHiFi

CCDKRocketsITAR

Wrapper Tools