Advanced Capabilities For Gas Turbine Engine Performance Simulations

A. Alexiou, E.H. Baalbergen, O. Kogenhop, K. Mathioudakis, P. Arendsen

Laboratory of Thermal Turbomachines
National Technical University of Athens

National Aerospace Laboratory, NLR
The Netherlands

http://www.vivaceproject.com
Component Zooming: execution of higher order analysis code and integration of its results back in the 0-D engine cycle

Distributed Simulations: technologies that enable a simulation program to execute on a computing system containing multiple processors interconnected by a communication network
Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen

Contents

- PROOSIS OVERVIEW
- Compressor Stage-Stacking
- The Engine Model
- COMPONENT ZOOMING
  - The ‘de-coupled’ Approach
  - The ‘semi-coupled’ Approach
  - The ‘fully-coupled’ Approach
- DISTRIBUTED SIMULATIONS
  - Implementing Distributed Simulations
  - Prototype Development
  - Future Developments
- SUMMARY & CONCLUSIONS
Component EL object-oriented code containing mathematical description of real component
PROOISIS OVERVIEW: Schematic View

Drag-and-drop icons from palette to construct engine model.

Connect components through appropriate communication ports.
PROOISIS OVERVIEW: Simulation View

Graphical representation of results

Define Engine Mathematical Model and Simulation Cases

Describe in EL calculation mode or type
## Contents

- **PROOSIS OVERVIEW**
- **Compressor Stage-Stacking**
- The Engine Model
- **COMPONENT ZOOMING**
  - The ‘de-coupled’ Approach
  - The ‘semi-coupled’ Approach
  - The ‘fully-coupled’ Approach
- **DISTRIBUTED SIMULATIONS**
  - Implementing Distributed Simulations
  - Prototype Development
  - Future Developments
- **SUMMARY & CONCLUSIONS**
Compressor Stage-Stacking: Methodology

Calculation of individual stage exit properties from dimensionless stage characteristics and geometry data

‘stack’ stages together to evaluate overall compressor performance
FORTRAN: `SUBROUTINE stageStack (arguments)` compiled as static library (.lib)

PROOISIS: “FORTRAN” FUNCTION `stageStack (arguments)` IN `stageStack.lib`

OR

C++ wrapper for FORTRAN subroutine:
```
extern "C" void __stdcall STAGESTACK (arguments);
void stageStackClass::stageStack(arguments)
{STAGESTACK (arguments); }
```

PROOISIS: \textbf{EXTERN CLASS} `stageStackClass`
\textbf{METHODS}
\textbf{EXTERN METHOD} `stageStack (arguments)`
END CLASS
\textbf{INCLUDE} “`stageStack.h” IN “`stageStack.lib””
Engine Model

Single-Shaft Industrial Gas Turbine Engine
with 15-stage axial compressor

Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen
Engine Model

Single-Shaft Industrial Gas Turbine Engine with 15-stage axial compressor

Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen
Component Zooming: Compressor 1-D

Compressor 1-D Inheritance Tree

- **GasInGasOut**: Abstract component containing general interface & equations for standard library components
- **GasTurbo**: Abstract component containing general interface & equations for standard library turbo-components
- **AbsCompressor**: Abstract component containing core compressor calculations
  - User-specified
  - No of bleed ports
- **Compressor#SasP**: Abstract component calling stage-stacking function
  - BETA map
  - MFT map
- **AbsCompressor1D**: Abstract component calling stage-stacking function
  - User-specified
  - No of bleed ports
- **Compressor1D#SasP**:
Component Zooming: De-coupled Approach (DCA)

 Instantiate Compressor 1-D

 Run multi-point steady-state simulation

 Produce PROOSIS compatible compressor 1-D map file
Component Zooming: De-coupled Approach (DCA)

Run engine simulation using compressor 1-D map

Select this map in compressor Editor

Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen
Component Zooming: Results (DCA)

Comparison of 0-D & 1-D Maps

Effect of Compressor Zooming on Heat Rate vs Load Characteristic

Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen
Component Zooming: Semi-coupled Approach (SCA)

1. Corrected flow zooming scalar
2. Isentropic efficiency zooming scalar
3. Fuel flow rate

Run Engine Model
All 0-D components

Get Compressor 0-D rotational speed, pressure ratio
Inlet temperature & pressure

Call External Compressor Stage-Stacking Function

Get Compressor 1-D mass flow and efficiency

1. Mass flow error
2. Isentropic efficiency error
3. Rotational speed

Intrinsic Newton-Raphson Function

PROOSIS EXPERIMENT SCHEME

Independent Variables
Specified load & Rotational speed
Stage geometry & characteristics

Objective Variables
The meaning of Modification factors

\[ f_q = \frac{q}{q_0} \]

Transformation of component performance maps
Structure of Adaptive models

Modification factors $f_k$ for components

\[ f_k = \frac{x_{p,k}}{x_{p,ref,k}} \]

- $x_{p,k}$: Actual value for parameter
- $x_{p,ref,k}$: Reference value for parameter

Transformation of component performance maps
Component Zooming: Results (SCA)

Variation of Zooming Scalars with Load

Scalar

Corrected Mass Flow

Isentropic Efficiency

Load
Component Zooming: Fully-coupled Approach (FCA)

Replace Compressor 0-D with 1-D one
### Component Zooming: Results (FCA)

#### Design Point Case
1.5% inter-stage bleed from 10\textsuperscript{th} stage

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>% DIFFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuel Flow Rate</td>
<td>0.289</td>
</tr>
<tr>
<td>Compressor Inlet Flow</td>
<td>0.111</td>
</tr>
<tr>
<td>Compressor Delivery Temperature</td>
<td>0.438</td>
</tr>
<tr>
<td>Compressor Pressure Ratio</td>
<td>0.211</td>
</tr>
<tr>
<td>Compressor Polytropic Efficiency</td>
<td>-0.238</td>
</tr>
<tr>
<td>Compressor Power</td>
<td>0.583</td>
</tr>
</tbody>
</table>
23 Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen

Contents

- PROOSIS OVERVIEW
- Compressor Stage-Stacking
- The Engine Model
- COMPONENT ZOOMING
  - The ‘de-coupled’ Approach
  - The ‘semi-coupled’ Approach
  - The ‘fully-coupled’ Approach
- DISTRIBUTED SIMULATIONS
  - Implementing Distributed Simulations
  - Prototype Development
  - Future Developments
- SUMMARY & CONCLUSIONS
Distributed Simulations: Rationale & Features

- Collaborative modelling among possibly geographically dispersed engineers
- Easy and efficient deployment of subsystem models
- Protection of ownership and IPR
- Reduction of simulation time through load distribution
- Size and complexity of the simulation model may grow irrespective of capability of computing infrastructure
- Reuse of submodels in different simulations
Technologies considered:

- **CORBA**: complex; did not catch up with growing Web developments and demands; high run-time costs; difficulties with security; versions & difficulties in backward compatibility; not supported by Microsoft...

- **DCOM**: serious security problems; did not catch up with growing Web developments and demands; deprecated in favour of .NET

- **Java RMI**: Java specific; being obscured by Web Service technology

- **XML and SOAP**: slower than e.g. CORBA and RMI but providing good basis for secure distributed web-based solutions in wide-area contexts

- **Web Services**: uses open standards and protocols (incl. SOAP and XML); commonly used nowadays to implement secure distributed solutions in SOA style; standards and tools are emerging

Web Service: *state-of-the-art technology enabling software components (clients, servers) to communicate over a network using standard messages and formats*
Prototype in VIVACE context: PROOSIS with compressor stage stacking function, available as “User Library”, running on a remote computer

Compressor stage stacking function:

- developed by, and proprietary code of NTUA
- written in Fortran, available as a shared library (DLL) on Windows
- available to PROOSIS users at NTUA as a PROOSIS User Library (PROOSIS’ mechanism to include customer code in engine simulations)
- code may be used but cannot be installed outside NTUA
Compressor Zooming via Remote Web Service Invocation between NLR & NTUA

PROOSIS simulation in The Netherlands

Stage Stacking Calculation in Greece
Distributed Simulations: Prototype Development (III)

Client (NLR)

PROOSIS

PROOSIS Web Component

Enables remote use and sharing of User Libraries over the Internet without distributing the code of the libraries

Server (NTUA)

Application Server

PROOSIS Web Service Operation

Stage-Stacking Function

Internet

Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen
Layered Structure of Prototype

Client Side (NLR)

1. PROOISIS Custom Library (C++)
   - JNI

2. Web Service Client (Java)

   SOAP over internet

Server Side (NTUA)

3. PROOISIS Web Service Component (Java)
   - JNI

4. Intermediate Code (C++)

5. Stage-Stacking Function (FORTRAN)

SOAP over internet

Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen
Advanced Capabilities For Gas Turbine Engine Performance Simulations
Alexiou, Baalbergen, Kogenhop, Mathioudakis, Arendsen
Distributed Simulations: Future Developments

- **Design of more generic (re-usable) interface**
  - ✓ multiple function implementations
  - ✓ not all layers need to be modified
  - ✗ additional overhead and delays in communication

- **Reduction of overhead caused by conversions & data transfers**
  - ✓ use pure C++ development environment
  - ✗ C++ support for Web Services limited/unstable
  - ✗ Java platform allows integration with other collaborative tools

- **Reduction of overhead in DLL loading and unloading**
  - ✓ load DLLs once and dispose after final calculation

- **Multi-user and security**
  - ✓ Use Web Service Security specification
  - ✓ allow multi-user access
Contents

- PROOSIS OVERVIEW
- Compressor Stage-Stacking
- The Engine Model
- COMPONENT ZOOMING
  - The ‘de-coupled’ Approach
  - The ‘semi-coupled’ Approach
  - The ‘fully-coupled’ Approach
- DISTRIBUTED SIMULATIONS
  - Implementing Distributed Simulations
  - Prototype Development
  - Future Developments
- SUMMARY & CONCLUSIONS
PROOSIS is a standalone, multi-platform, object-oriented simulation environment for gas turbine engine performance simulations. It can be used to create, run, manage and share engine models using either the standard or custom libraries of engine components. The feasibility of performing multi-fidelity and distributed simulations with PROOSIS was demonstrated in this paper.

Using the model of an industrial gas turbine engine and a 1-D compressor stage stacking code as an example, different implementations for integrating high fidelity component analysis in overall engine simulations were presented. The tool’s flexible and extensible architecture gives the user the freedom to select the most suitable approach for a particular simulation case.
The stage stacking code is also used to demonstrate distributed simulations. A prototype of a Web Component has been created and successfully tested that remotely invokes the code from an engine simulation, via the internet, using Web Services technology.

These demonstrations prove that the tool’s architecture is adaptable enough to integrate different modelling methods and its potential to fulfil its role as a shared simulation environment.