



LaRC

Aircraft Design Problem Implementation under the Common Object Request Broker Architecture (CORBA)

Raj Sistla, Gus Dovi, Phillip Su & Ram Shan

Computer Sciences Corporation
Hampton, Virginia.

40th AIAA/ASME/ASCE/AHS/ASC SDM Conference

April 12-15, 1999 / St. Louis, MO.



LaRC

Route Map

- Background
- HSCT Problems
- Frameworks
 - FIDO
 - CORBA/Java
- Example
- Concluding Remarks



LaRC

Framework Definition

A framework for multidisciplinary design optimization is defined as a hardware and software architecture that enables integration, execution, and communication among diverse disciplinary processes.



LaRC

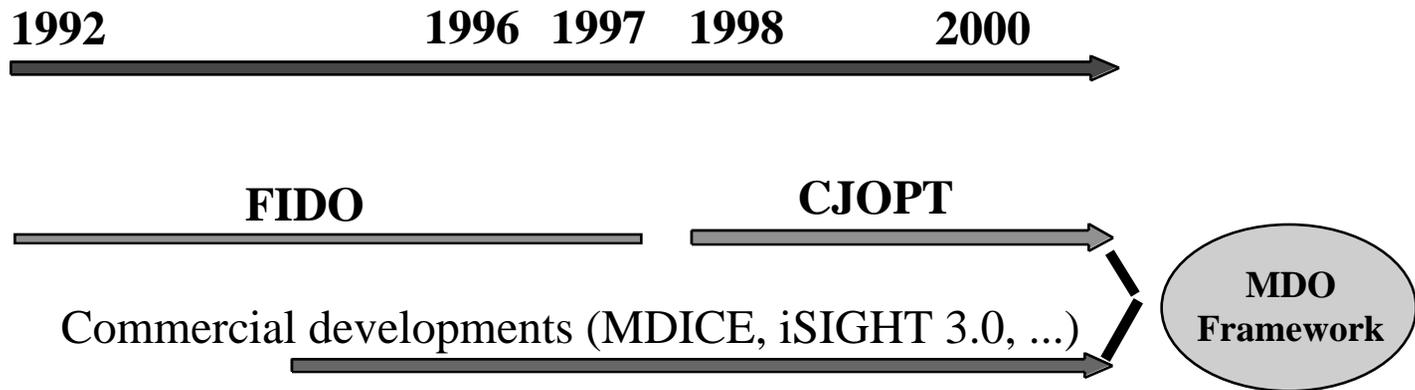
Motivation for a Framework

- Aircraft design is multidisciplinary in nature.
- Different disciplines execute independent of each other.
- Potential exists for concurrent execution of some subtasks.
- Hardware requirements vary with the discipline.
- Large quantities of data and files generated.
- Potential exists for automating the design process.



LaRC

Framework History





LaRC

HSCT Applications

Application (year)	HSCT2.1 (94 - 96)	HSCT3.5 (95 - 97)	HSCT4.0 (97 - 99)
Design Variables	5	7	271
Constraints	6	6	31868
Major Legacy Codes			
Aerodynamics	Wingdes	ISAAC	CFL3D, USSAERO
Structures	ELAPS	COMET	GENESIS
Performance	Range equation	Range equation	FLOPS
Propulsion	Engine deck	Engine deck	FLOPS
Analysis Processes (without looping)	10	20	70
Analysis Control			
Major Loops	Weight Conv., Trim	Weight Conv., Aeroelastic, Trim	Aeroelastic, Trim
Load conditions	2	2	7
Mission conditions	1	1	10
Process (with loops)	O(10)	O(100)	O(1000)
Total time	O(minutes)	O(hours)	O(1 day)
Optimization Cycle			
(ndv+1) #analysis processes	O(100)	O(1000)	O(100,000)
Total time/cycle	O(10 minutes)	O(3 hours)	O(3 days)



LaRC

FIDO - Framework for Interdisciplinary Design Optimization

- Home grown, PVM based.
- Communications Network connects the computers.
- Communications Library handles communications.
- Master Scheduler controls interactions between disciplines.
- Primary segments execute in a “host/slave” mode.
- Data Manager provides central database connectivity.



LaRC

FIDO Shortcomings

- Research tool.
- Platform (and OS) dependent.
- PVM dependent.
- User exposed to complex communications constructs.
- Intertwining of framework tools and application.
- Switching discipline codes is not a trivial task.
- Emphasis on framework.



LaRC

iSIGHT 3.0

- Computer-aided engineering management software to integrate custom/third party analyses, visualization, and monitoring tools.
- Uses MDOL language to describe problem to be solved.
- Comprehensive suite of graphical tools to:
 - assemble application from legacy codes
 - monitor program execution, analyze results, provide reports
- A suite of optimization programs to choose from.



LaRC

iSight 3.0 Shortcomings

- Not geared to handle our large problems.
- User required to learn the MDOL language.
- User required to learn Tk/Tcl scripting language.
- Sequential processing on single host computer.
- No database or file management capabilities.
- No debugging capabilities.
- Not ready for our application.



LaRC

Motivation for Present Approach

- Limitations of FIDO (home grown), iSIGHT (COTS).
- Reduction in resources for continued development and maintenance of framework.
- Desire for a common working environment for multiple MDO projects.
- CORBA is an industry standard for distributed applications.
- Java is becoming CORBA compliant.



LaRC

CJOPT Building Blocks

- Common Object Request Broker Architecture (CORBA).
- Java/CORBA implementation.
- Java Language and APIs:
 - Java DataBase Connectivity (JDBC)
 - Java Native Method Interface (JNI)
 - JavaBeans (Component technology)
 - SQL compliant database (mSQL)



LaRC

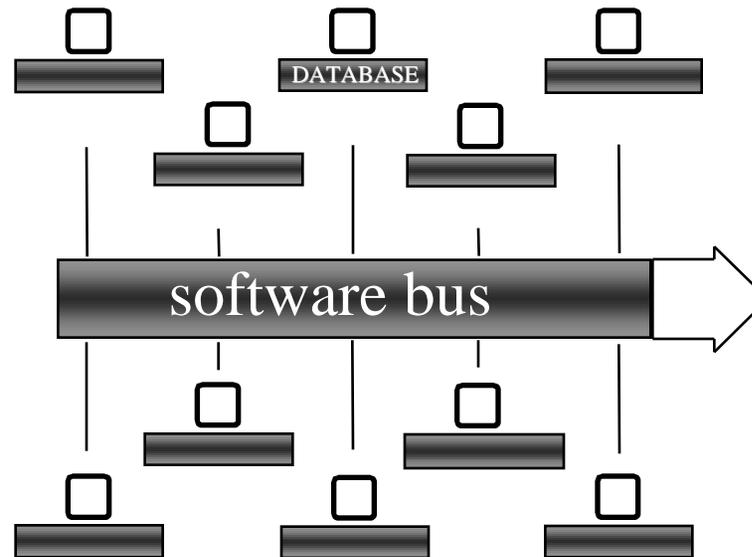
CORBA fundamentals

- Object-oriented in a Client-Server environment.
- Client needs to know only an “interface” for requesting services of a server object.
- Server does bulk of the work in providing this service.
- Client is not required to know:
 - How the service is provided,
 - Language the services are implemented in,
 - Where the service resides.



LaRC

CORBA fundamentals (contd.)

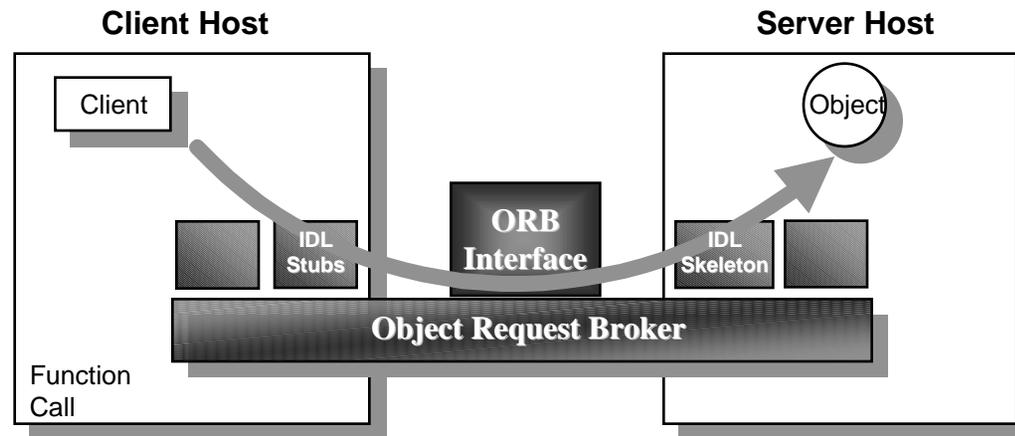


- specification for developing distributed applications.
- promotes re-use of existing software components.



LaRC

CORBA fundamentals (contd.)



- ORB is a software component that mediates transfer of messages.
- hides the underlying complexity of network programming.



LaRC

Programming Steps

- “Wrap legacy code” as an object.
- Define an interface to this object.
- Write the implementation of this interface.
- Write a server code to register server with ORB.
- Write a client application to use the CORBA objects
or,
Write JavaBeans, import into Builder, build application.
- Run client application.



LaRC

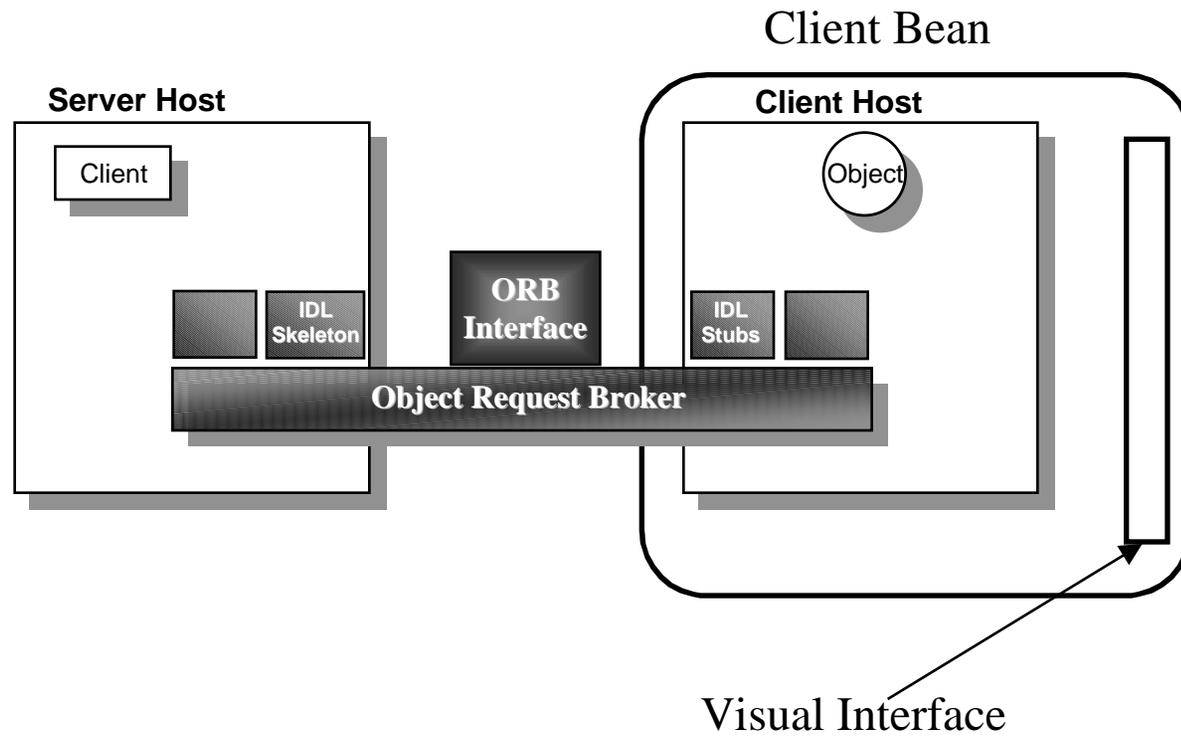
Data Management

- Central relational database.
- Commercial SQL-compliance.
- Objects use Java Data Base Connectivity (JDBC).
- User-specific tables for transient data.
- File management information stored in database.



LaRC

JavaBeans Technology





LaRC

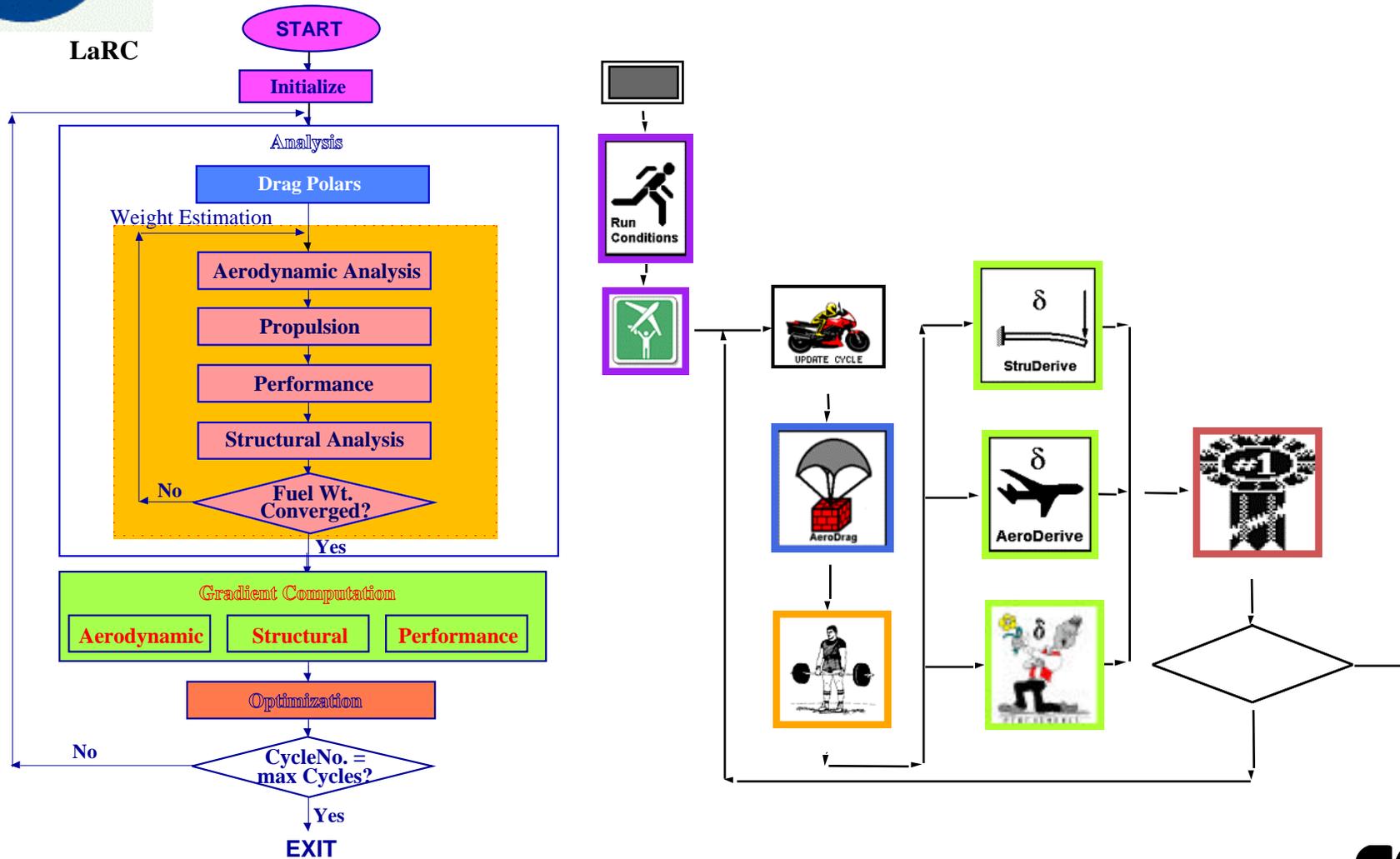
JavaBeans Component Technology

- Reusable software component that can be manipulated visually.
- Based on event sources and generators.
- Custom property editors.
- Bean customizers.
- Bean can be saved to persistent storage.
- Visual, interactive bean connectivity.



HSCT2.1 PROBLEM IMPLEMENTATION

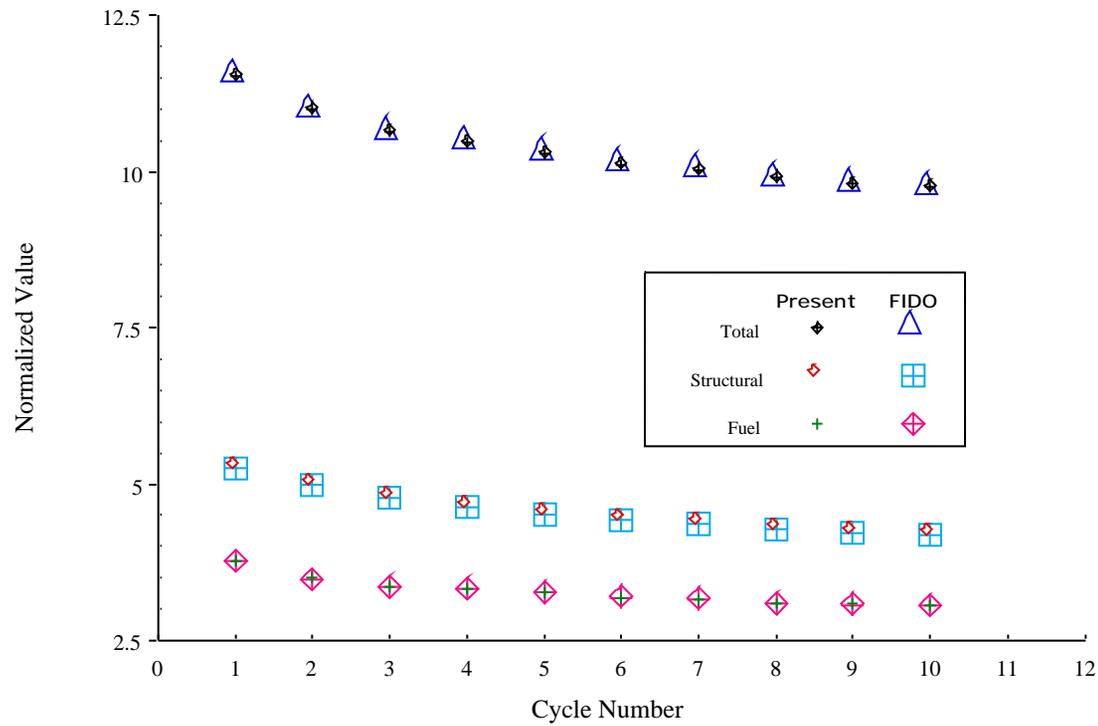
LaRC





LaRC

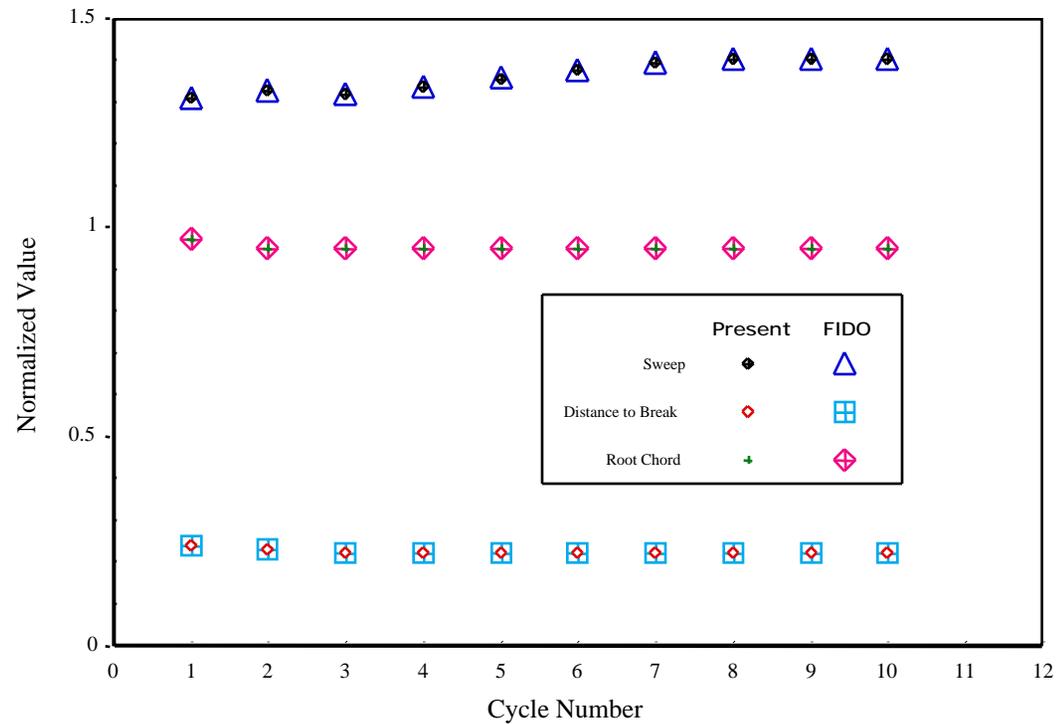
Variation of Airplane Weights





LaRC

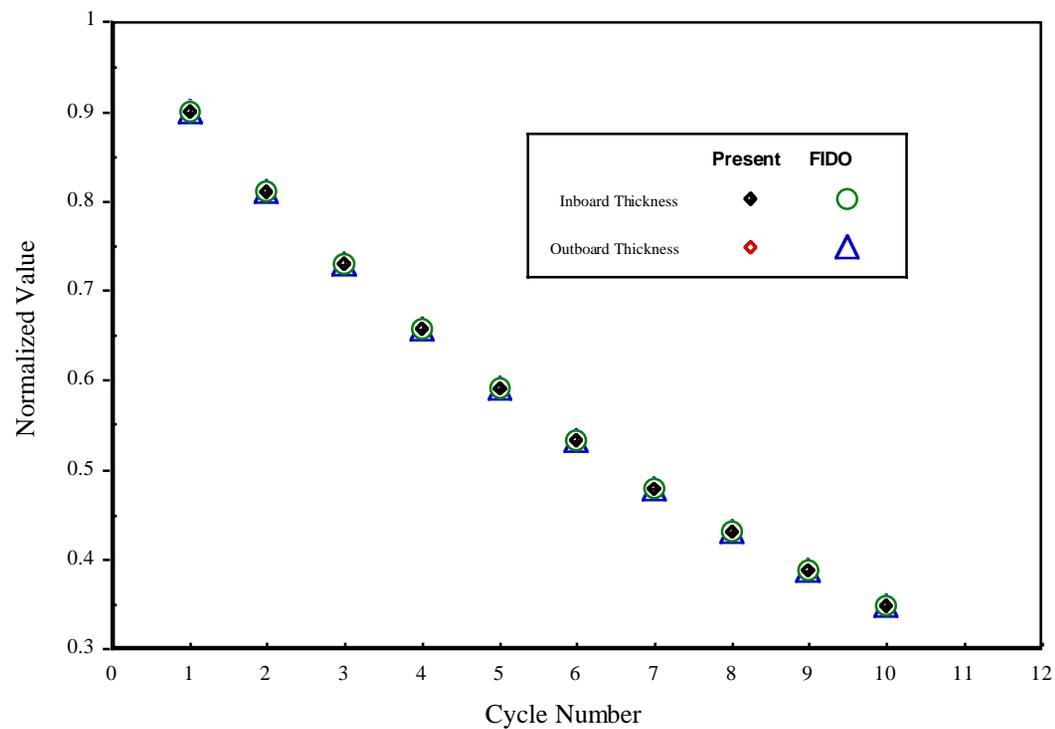
Variation of Shape Design Variables





LaRC

Variation of Structural Design Variables





LaRC

Current Focus

- The present approach is being used for the implementation of a more realistic HSCT design problem.

Application	HSCT2.1	HSCT 4.0
Design Variables		
Shape	3	27
Structures	2	244
Constraints		
Geometry	-	216
Aerodynamics	2	-
Performance	-	10
Weights	-	2
Structures	4	4520 (per load condition)
Analysis Codes	10	70



LaRC

Concluding Remarks

- Component based framework.
- User spared the complexity of network programming.
- Server code responsibility is distributed.
- Server functionality maintained and upgraded without client knowledge or service interruptions.
- Easy to configure and use via graphical interface.
- COTS product and Java language and its APIs.
- “Can only get Better!”