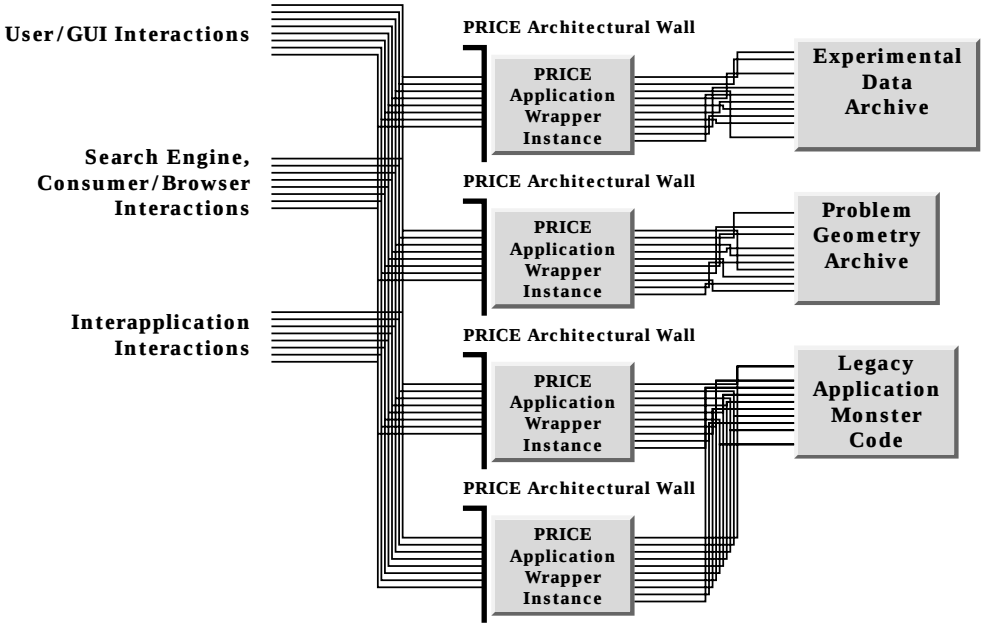
Portable, Redesigned Integrated CFD and Experiments (PRICE)

An object-oriented architecture for the wrapping of all forms of engineering experiment, design, analysis, and simulation has been devised.

1. Wrapped applications are made self-revealing.
2. Entire engineering process captured with logs, change histories, and the like.
3. Standardization is the concern of the wrapper, not the resource.
4. CORBA distributed object implementation hides proprietary resources behind an architectural wall.



Basic Object Primer

1. Objects are defined in classes, but are worked with as individual instances of their class.

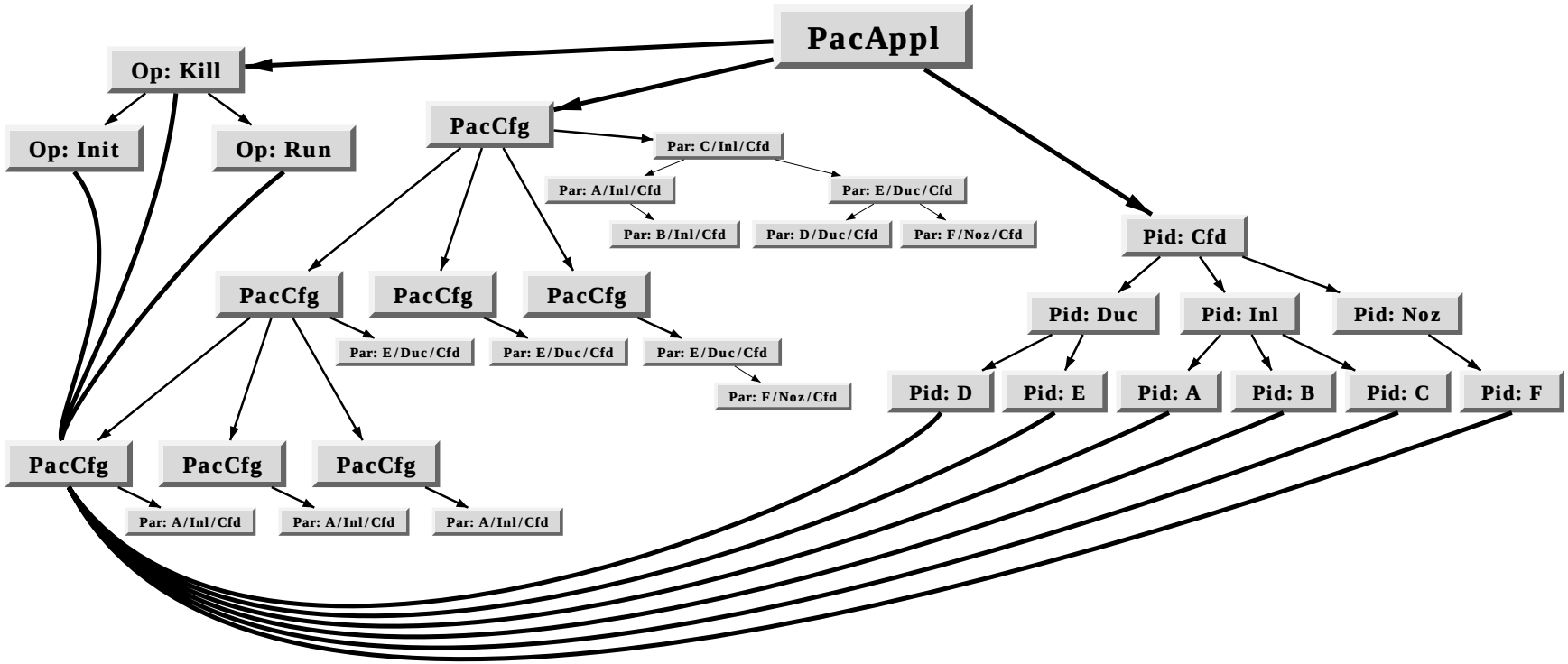
2. Objects usually have functionality and data. Data may be shared among all the instances of a class or may be individual to each instance. Functionality is shared by all instances.

3. Classes may be derived from classes, inheriting the attributes of the parent class.

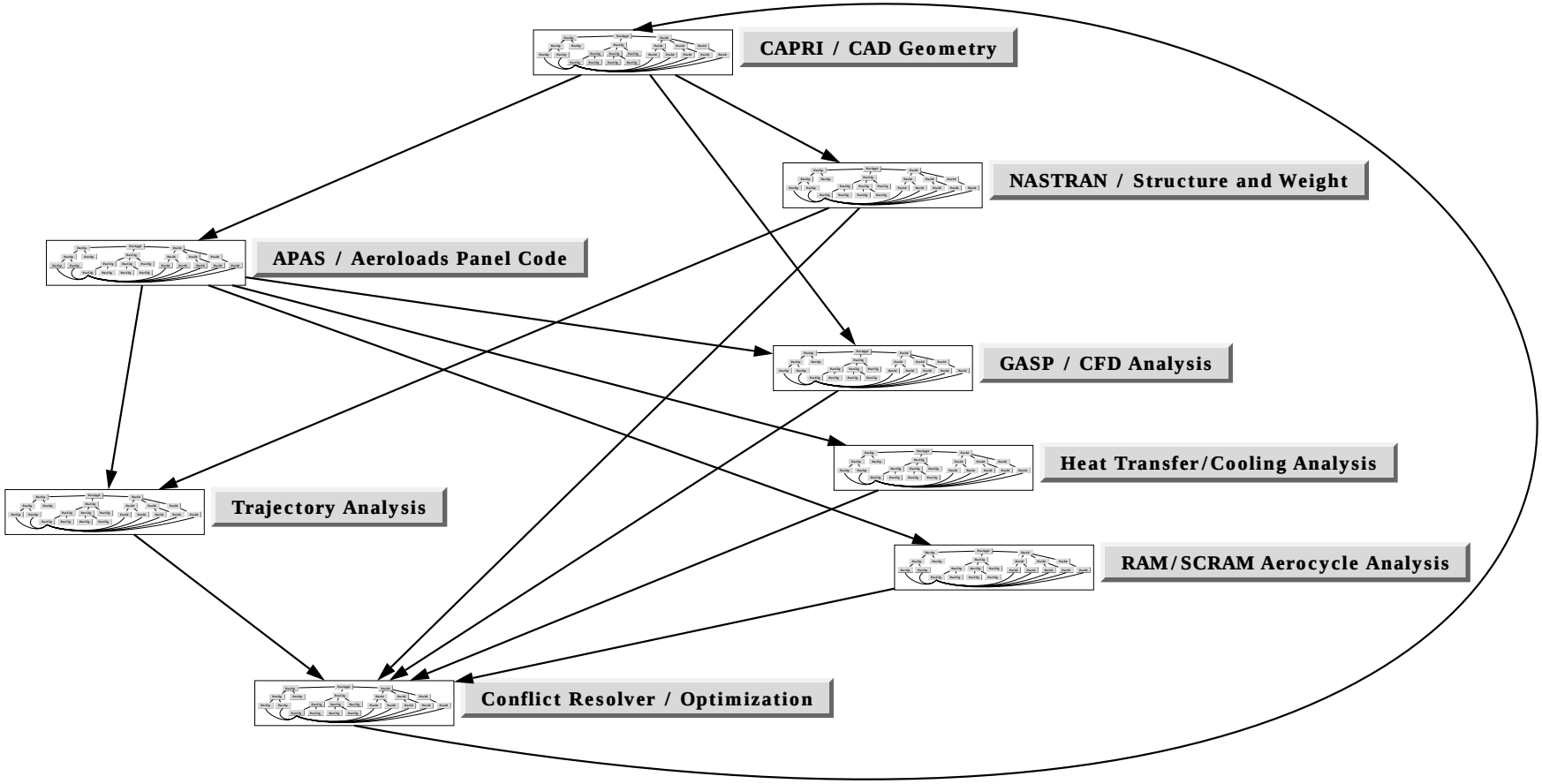
4. Derived classes may add to, override, alter, extend, turn off, or otherwise mangle inherited functionality.

5. The correct functionality is obtained without regard to whether or not the program "knew" the exact kind of object in use.

6. Usually, the first thing provided is a way to find out the kind of any given object.

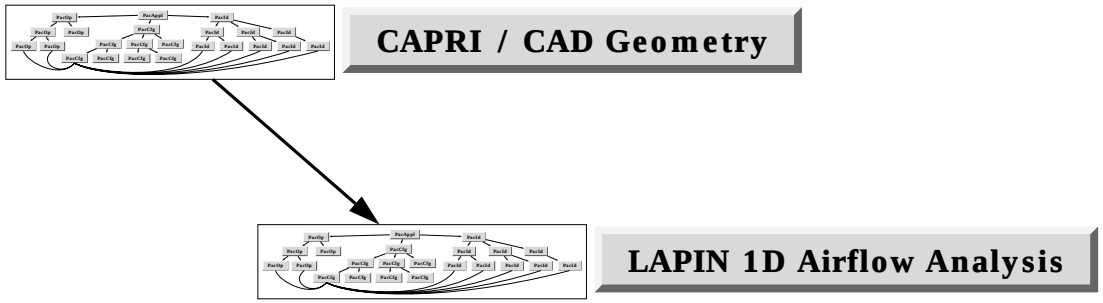


PRICE Application Structure

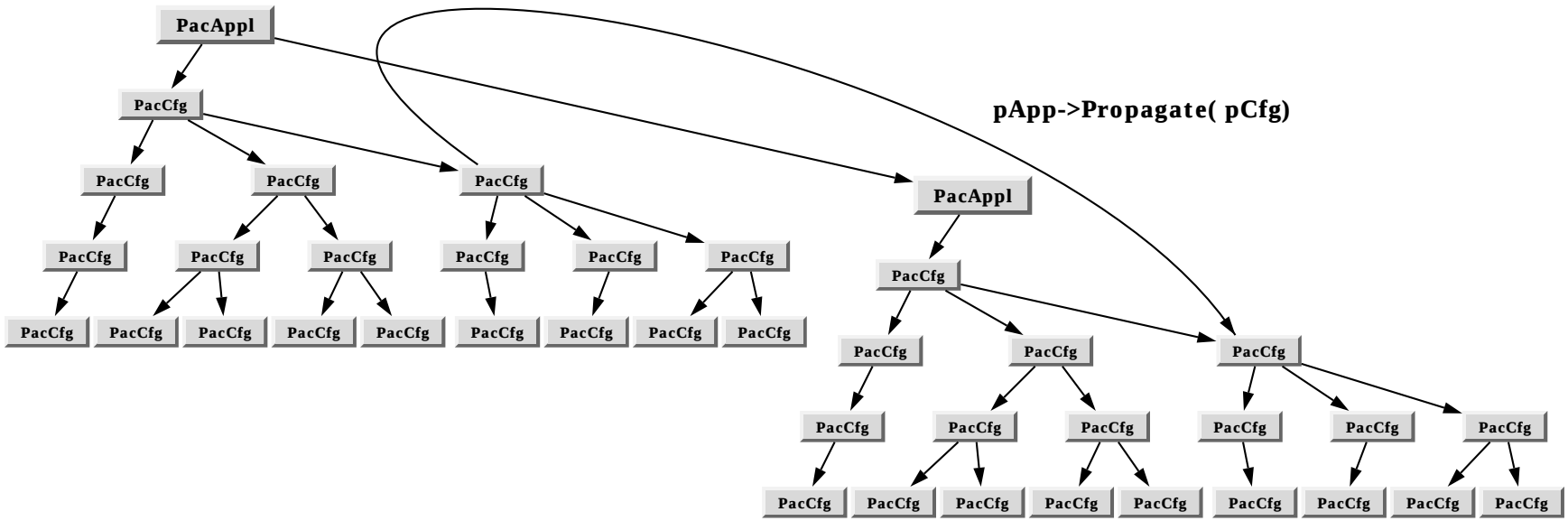


Approximate Rocket-Based Combined Cycle Engineering Solution Flow

Coordinated Automatic Recirculation of Design Feedback



C++ Environment: Inter-Application Operation Demonstration



Data Configurations After Propagation has Occurred

Semantically-Meaningful Dimensional Parameter Objects

1. Dimensionality consists of three encapsulated elements:
   a) A measurement system (English, metric, etc.) identified by code number,
   b) A computable dimensional characteristic vector giving the participation of elemental dimensional components (length, mass, time, etc.), and
   c) A power of application for that dimensional characteristic.

2. Parameters provide system-of-measurement-sensitive access functionality.

3. Operator overrides enforce conventional dimensionality rules. For example, addition and subtraction require aggregate dimensional congruence.

4. Assignment replicates all three dimensional components, rather than causing a dimensional conversion. Assignment does not introduce numerical conversion noise.

5. Objects designating a kind of dimensionality require only characteristic congruence. Thus, a unit vector may still be encapsulated in a length vector object. Since the application of a unit vector's length characteristic is zero, the unit vector becomes invariant between systems of measurement.

Geometry Parameter Objects

1. Based where appropriate upon the dimensionally-sensitive parameter classes.

2. Follows the structuralization of CAD information provided by the CAPRI (Haimes/MIT) effort.

3. Provides additional functionality to identify an arbitrary closed plane section of a geometric surface (if possible) and compute the area of that section. (A RBCC need.) The section is limited to tesselation accuracy.

4. Will also identify one or more arbitrary open plane sections.

5. A method for computing the surface area of an arbitrary part of a geometric surface (another RBCC need) has been identified and will be implemented in the next round of improvements.

6. Several approaches for handling translating centerbody geometries (yet another RBCC need) have been identified. One will probably be selected and implemented in the next round or two of improvements.

Gas Parameter Objects

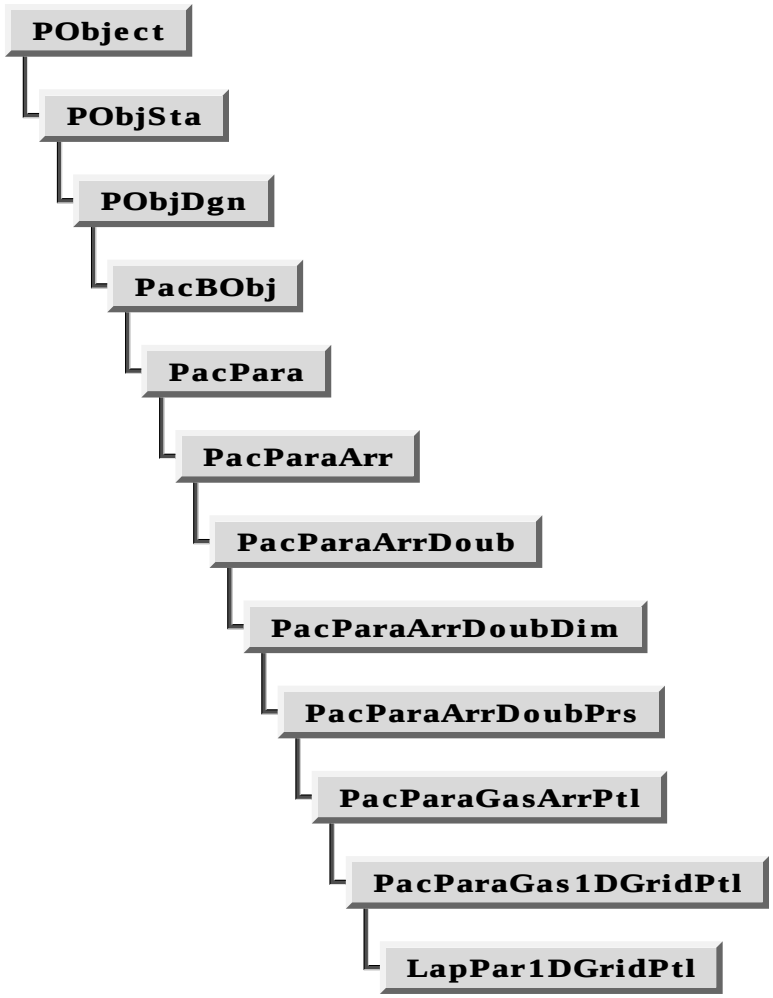
1. Begins by defining common CFD types: total and static pressures, Mach number, CFL number, and the like.

2. Extends these types into grids, for example a 1-dimensional grid of total pressures.

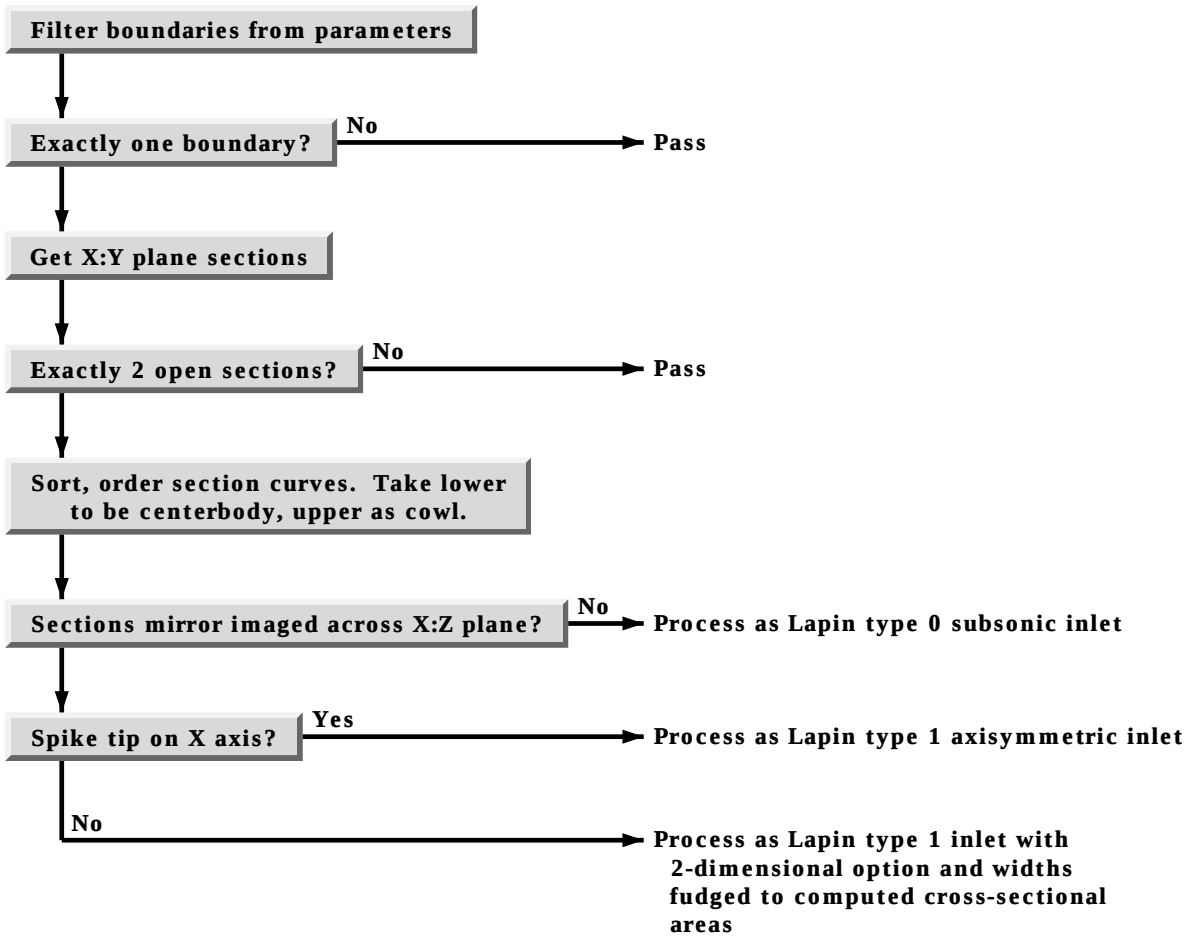
3. The grid parameter types associate a grid position parameter through the key parameter mechanism.

4. The grid parameters may then be further arranged into time histories of known types. For example, a time history of 1-dimensional grid total pressures.

5. Time histories then associate through the key parameter mechanism a parameter giving the times of each history element.



Inheritance of the Lapin 1-D Grid Total Pressure Class



Information Propagation: Lapin Geometry Consumption Decision Flow

CORBA Migration

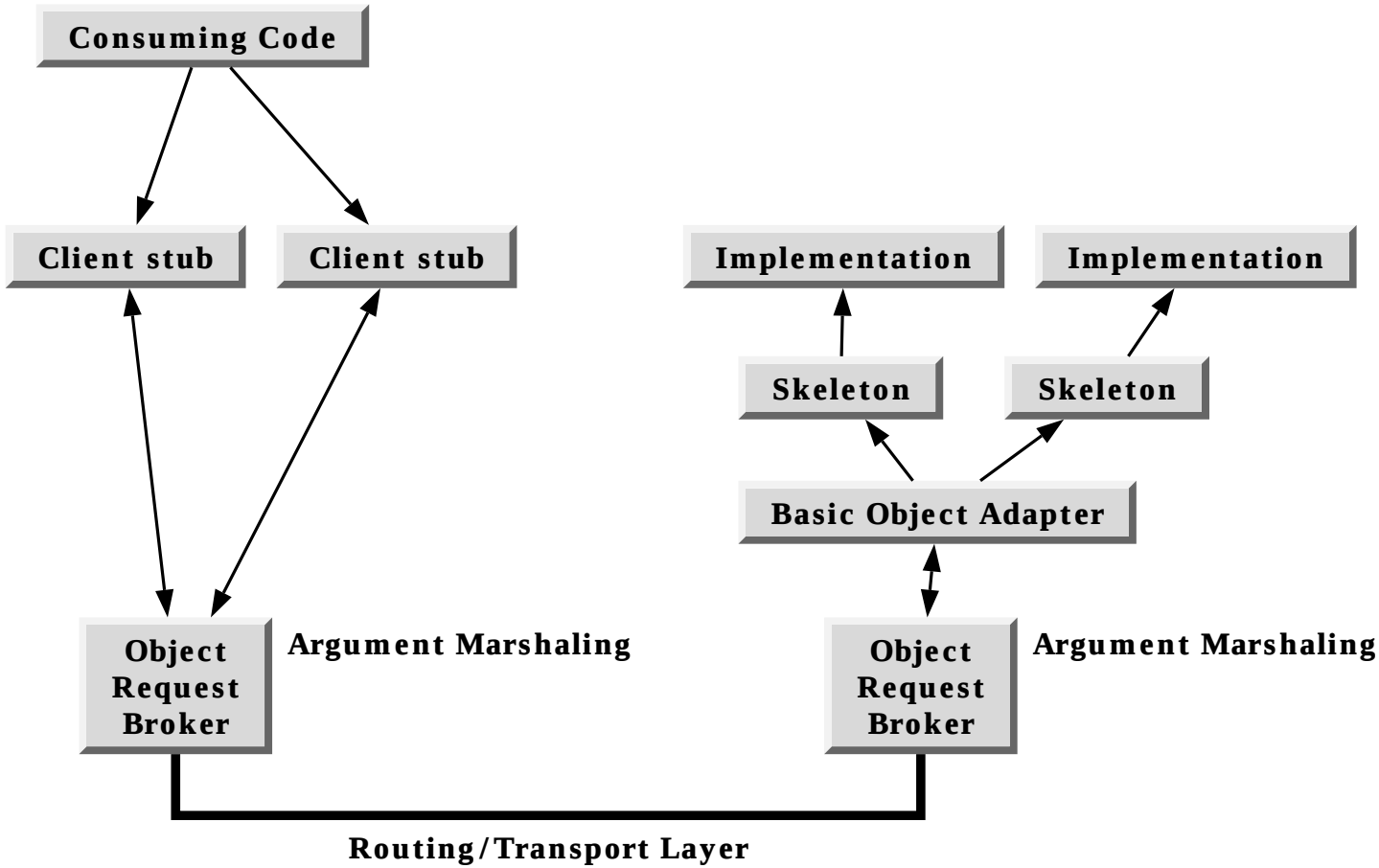
Real life requirements dictated a migration to the CORBA distributed-object environment.

1. Real applications require enormous data capacities (10 Gigabytes and up) which cannot be accommodated in a single C++ virtual address space. Service activated objects (Visigenic feature?) work around this problem.

2. A need exists to provide for multiple accessors, allowing for both generators and browsing consumers of information.

3. A cross-language capability is needed to allow consuming tools (such as GUIs) to be written in languages such as Java.

4. Some application providers wish to provide service without revealing implementing technology.



CORBA at a Glance