OpenStatServer

Deploying Custom Statistical Computation to Scientific Clients

Gregory R. Warnes
Associate Director
Pfizer Global Research & Development
2005-05-25

Abstract

The OpenStatServer Project is developing software infrastructure which enables statisticians to easily develop and deploy custom applications to scientific clients.

Our approach aims to maximize the penetration of proper statistical methods while minimizing the cost of development, deployment, and maintenance.

This presentation outlines our work.

Outline

- The Problem: Deploying custom statistical applications
  - Observations
- Proposed Solution: OpenStatServer
  - Project Goal
  - Mechanism
  - System Architecture
  - API Specifications
  - Expertise Separation
- Project Phases
  - Phase 1: ‘**Chaco**’ – Next Step – Early development
    - New Infrastructure (alpha code available)
    - R, SAS, Zope drivers
  - Phase 2: ‘**Chichén-Itzá**’ – TBD – Concept stage
  - Long Term Opportunities
- Organizational Issues
- Project Participants

What is the Problem?

Many scientists perform experiments which are
- Standardized
- Regularly Repeated

for which an appropriate statistical method is
- well defined

but is not available in software that is
- Available to the client
- Customized to the specific task
- Simple to use.

As a consequence proper (re)analysis requires
- Significant statistician time and effort, or
- Even More Scientist user time & effort or
- Is not done (properly)

hence is
- hard to manage/repeat/replicate/track/etc.

Scientist: Client, Physician, Analyst, …

Statistician: Computational Methods expert, Scientific Programmer, …

Etc.

Observations

- Statisticians consult with client to determine proper analysis method → Consultation
- Statisticians can easily code necessary computations using existing statistical software tools (SAS, R, etc) → Module Coding
- Existing statistical software tools are not *end-user* friendly, so a custom wrapper needs to be created for the end user → UI Coding

Statistical software tools need to be connected to UI development tool

OpenStatServer

connect custom statistical computations implemented in any statistical tool to user interfaces implemented in any UI client

OpenStatServer Project: Goal

Make it easy to connect computational modules with UI clients regardless of the specific applications utilized.

…

WebServices++ for developers

AND

Computational Legos for users

OpenStatServer Project: Mechanism

- Middleware Layer
  - Standard Computational Module Interface
  - Standard Client Interface
  - Transformations and data mappings
  - Mapping of task to computation server
  - Movement of data to/from computation server
- Machine-readable meta information formats for
  - Locating appropriate service (‘module’)
  - Integrating computational modules into client software tools
  - Managing computational module lifecycle

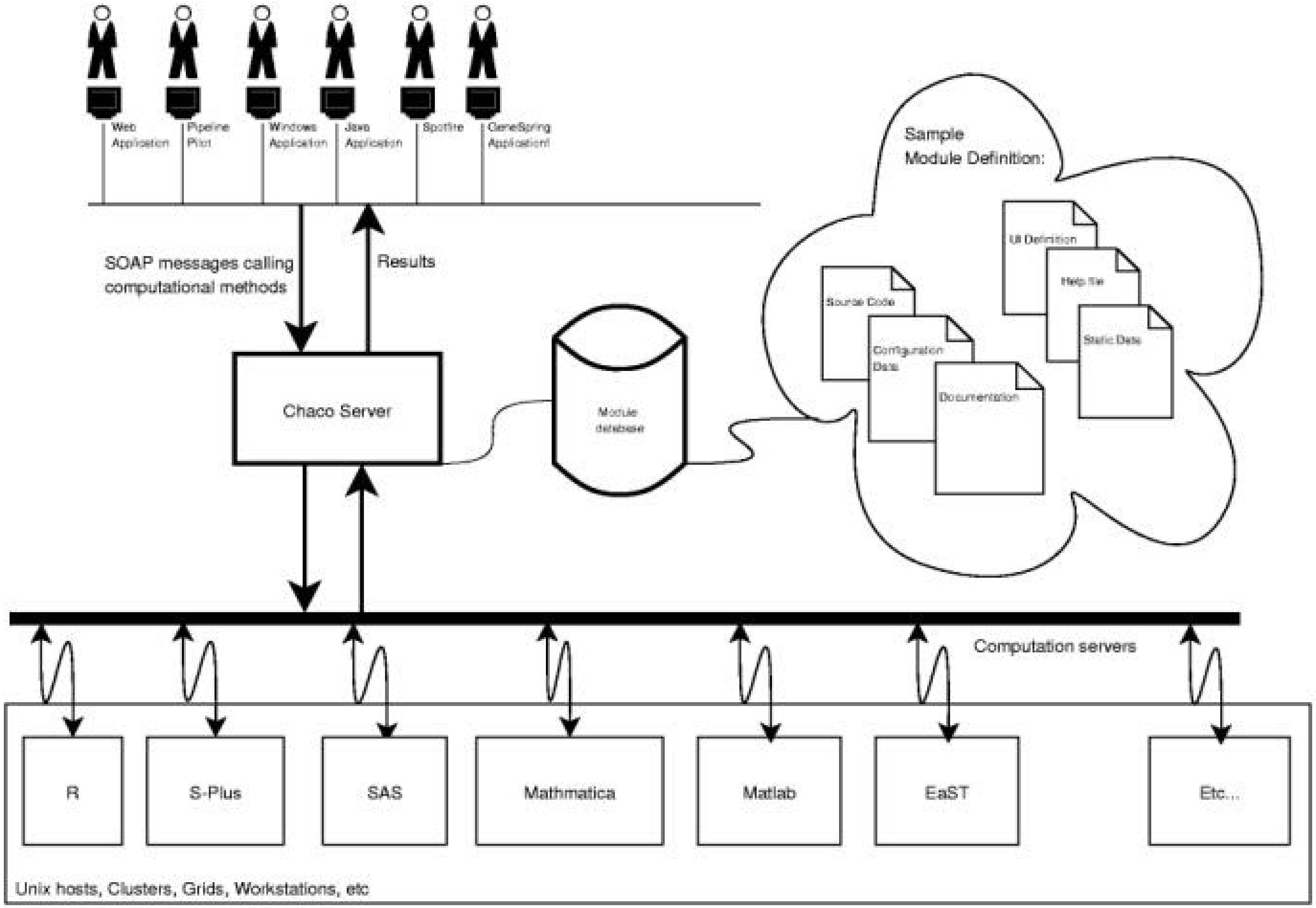
(via existing XML web standards when available)

- Computation Tool Adaptors

  (R, SAS, S-Plus, MATLAB, …)
- UI Tool Adaptors

  (Pipeline Pilot, Spotfire, EJB, BEA Web Objects, MS-Excel, …)

OpenStatServer: System Architecture



OpenStatServer Project: API Specifications

- Communication Protocol:
  - SOAP
  - Synchronous and Asynchronous calls supported
- Meta information:
  - WSDL: API call details (W3C standard)
  - XForms: API parameters definitions, validation (W3C standard)
  - XHTML: Presentation information (W3C standard)
  - XML: Additional module lifecycle and detail information
    - Code version
    - Validation level: devel, test, production, validated, peer-reviewed).
    - Author
    - Release Date
    - Algorithm details
    - Software dependencies
    - Computational resource requirements
    - Computational resource costs
    - …

OpenStatServer Project: Expertise Separation

- Client:
  - Task specification requirements
  - UI requirements
- Methods expert (Statistician):
  - Computational modules
    - Code in appropriate computational system (R, SAS, C, …)
    - Provide complete API, usage, lifecycle information
  - Specification of data flow to/among modules
- OpenStatServer:
  - Computational system adaptors
  - UI adaptors
  - Mechanism for identification of relevant modules
- UI designer:
  - UI interface
  - UI documentation

OpenStatServer Project: Project Phases

Phase I – ‘Chaco’

1. Separate UI and Computation layers
2. Multiple clients (Zope, Java, Pipeline Pilot)
3. Multiple compute server tools (R, SAS)
4. Distribute computations

Phase II – ‘Chichén-Itzá’

1. Module Reuse (Add module Meta-information)
2. Automatic computation wrappers for UI systems (More module meta-information)
3. More clients + servers

Phase III – TBD, Ideas include

Ontology of Computations

Data type ontology + automatic translations

OpenStatServer Server: Alpha “Chaco” code

http://openstatserver.org

- Complete:
  - APIs
    - Computation Request + Response (both synchronous & asynchronous)
    - Task Assignment to Node (Auction/bid mechanism)
    - Task Execution
    - Task State Query
  - Module repository
  - Data marshalling
  - Logging
  - Appropriate error handling
  - Python Compute Module adaptor
- Placeholder implementations:
  - Module Query API
  - Task bidding (node side resource evaluation)
  - Task assignment (master)
  - R Compute Module adaptor
  - Python Client adaptor

OpenStatServer: “Chaco” release To-Do

- Nail down initial definition of XML metafile file formats
- Module Query API & Code
- UI Client Adaptors
  - Zope
  - Pipeline Pilot
  - Spotfire
  - BEA Weblogic
  - Apache Tomcat
  - Synapsia / GeneSpring / SigNet (Agilent)
- Computational Module Adaptors
  - R driver + translator for R manual page to XML metafiles
  - SAS driver
  - Command-line wrapper
  - Interactive tool for generating XML metafiles
  - Synapsia / GeneSpring / SigNet (Agilent)
- Policy Managers
  - Access Control (Security)
  - Task bidding (node)
  - Task assignment (manager)
- Improved Community Web Site (partially complete)
- Improved Documentation

OpenStatServer: “Chichén-Itzá” Release

- Computational Infrastructure Support
  - LSF
  - Globus
- Additional Computational Server Adaptors
  - S-Plus driver + translator for S-Plus manual page to XML metafiles
  - MATLAB driver
  - Mathematica driver
  - Java Beans
- Additional Client UI Adaptors
  - MS-Excel
- Online Repository for Computational Modules
- Extend SOAP with richer data objects: DataSet, TimeSeries, …

OpenStatServer: Long Term

- Syntax for specifying module resource requirements
  - → Allow selection of module appropriate to specific data & parameters at hand
- Ontology of computational tasks
  - → Allocation of an appropriate module from a class providing equivalent functionality (t-test < ANOVA < linear model < generalized linear model < …)
- Mechanism for 'pipelining' or 'pooling' module calculation requests
- Mechanisms for parallelization of module calculations

- Formal Non-Profit Organization?
- Submission to Standards Bodies?
- …

OpenStatServer: Next Steps - Organizational

- IP Issues
  - Software License
    - Current License is GPL
    - Apache 2.0 might be more appropriate
  - Contributor Agreement
    - Avoid later conflict
    - Something like Python Software Foundation Contributor Agreement
- Meetings
  - Host: Greg Warnes, Pfizer
  - Secretary: Nitin Jain, Pfizer
  - When: 1st and 3rd Friday of each month at 11am Eastern Time.

OpenStatServer Project: Participants

- Pfizer
  - Gregory Warnes (NonClin Stats, organizer)
  - JC Lawrence (primary programmer)
  - Nitin Jain (NonClin Stats, organizer
  - Phillip Ross (IT)
  - Ron Menner (IT)
  - Jim Rogers (NonClin Stats)
  - Larry Zaccaro (Strategic Alliances)
  - Steve Matteson (IT)
- Spotfire
  - Dan Nathan
- Scitegic
  - M Hassan
- Yale
  - David Tuck (Pathology)
  - Jay Cowan (Pathology)
  - Eileen Ye (Pathology)
  - Peishen Qi (Computer Science)
  - Martin Schultz (Computer Science)
- IBM
  - David Martin (IBM Healthcare & Life Sciences)
  - John Toppa (IBM Healthcare and Life Sciences)
  - Chris Cooper (IBM Healthcare and Life Sciences)
- Platform Computing
  - Chris Smith
- Agilent
  - Jordan Stockton

Finis