Microsoft View: Clusters, Grids & Web services

Samuel Y. Shen
Sr. Director
Microsoft Taiwan Corporation
Overview

- Clusters
- Web services
- Grids
- .NET Framework
- Web services futures
Clusters www.microsoft.com/hpc

Computational Clustering Tech Preview Toolkit

- Everything you need to get up and running with a cluster on Windows
  - Windows XP Pro, Windows 2000 Advanced Server, VS.NET evaluation version
  - Interix 2.2
  - MPI/Pro 1.6.4 & ClusterController 1.5.1 from MPI Software Technology, Inc.
  - Math Kernel Libraries 5.2 from Intel
  - Computational Cluster Monitor from Cornell Theory Center
  - PLAPACK package (open source software)

.NET can hide the complexities of computing
Goal: To utilize existing commercial supported software and tools to build clusters

Supporting

Computational finance, Computational biology, Bio informatics, Facture Mechanics, Behavioral and social sciences, computer science, engineering, geosciences, mathematics, physical sciences, and business

Clusters

- Run on Windows 2000 and supported commercial apps
  - MPI, PVM, Fortran, MPI enabled MatLab, MP SAS, Excel, SQL
- Reliability — 99.999999%

2 locations

- CTC Ithaca
- CTC Manhattan
Web Services: The Key?

- **Web SERVER:**
  - Given a url + parameters
  - Returns a web page (often dynamic)

- **Web SERVICE:**
  - Given a XML document (soap msg)
  - Returns an XML document
  - Tools make this look like an RPC.
    - $F(x, y, z)$ returns $(u, v, w)$
  - Distributed objects for the web.
  - + naming, discovery, security...

- **Internet-scale distributed computing**
TerraServer – Whole Earth Imagining
Terraserver Architecture

- **Standard Browsers**
- **Smart Clients**
- **Windows Forms**
- **.NET Framework**

- **HTML**
- **Image/jpeg**
- **XML**

- **Map UI Web Forms**
- **Map Server Http Handler**
- **TerraServer Web Service**
- **ADO.NET**
- **OLEDB**
- **SQL 2000 1.0 TB Db**

To the rescue!
TerraService

http://terraservice.net

Give programming interface to the images and data contained on TerraServer

- Launched June 2000
- Web Service created in 3 days

Terra-Tile-Service

- Retrieve meta-data on TerraServer imagery
- Query TerraServer Gazetteer
- Retrieve TerraServer Tiles (imagery)
- Simple Projection conversions

Landmark-Service

Geo-coded data of well-known objects (points), e.g. Schools, Golf Courses, Hospitals, etc.
Polygons of well-known objects (shapes), e.g. Zip Codes, Citys, etc.

Intended to be “overlay” information for Terra-Tile-Service applications

Enables users/service providers to control user interface to TerraServer imagery
SkyServer – a Virtual Astronomy

The images from the SDSS project are organized into **stripes**. These are long images, 24,000 pixels wide (2.5 degrees), and typically over 1 million pixels long (120 degrees). The SDSS covers the northern sky with 45 overlapping stripes. The stripes are designed to avoid the dust in the plane of the Milky Way, shown as the red areas in the diagram at right.

Each stripe is merged from two **runs**, taken on separate nights. The telescope can complete only a single 8-hour run in any given night. Each run consists of **camcels** (camera columns); each camcel corresponds to each array of CCD detectors of our camera. Each camcel takes images in five colors.
SkyServer

http://skyserver.sdss.org/

Give educational access to Sloan Digital Sky Survey

- Based on the TerraServer design
- Live example of Web Services in e-Science
- Access to Federated Astronomy Databases (Fermilab/SDSS, JHU/FIRST, Caltech/2MASS)
- Great test bed for .NET ideas
- SDSS Cutout Web Service
  http://skyservice.pha.jhu.edu/SdssCutout/
- SkyQuery
  http://www.skyquery.net/
- Jim Gray (MSR) & Alex Szalay (JHU)
  http://research.microsoft.com/pubs/
Grids

- Grids are Web Services
  - .NET connected software makes it easy to create Web Services
- DataGrids – Accessing and unlocking the data has huge potential
  - Provide DataSets
  - Bring the computation to the Data instead of bringing the Data to the computation
- Working with Globus & GGF
  - GT 2 on Windows
  - OGSA for .NET
- Commercial adoption barriers
  - Information/computing is a competitive advantage
  - Does Web Services covers enough? (80/20)
.NET Framework

http://msdn.microsoft.com/net
Inside the .NET Framework

- ASP.NET: High-productivity environment for building and running Web services
- Secure, integrated class libraries:
  - Unifies programming models across languages
  - Enables cross-language integration
  - Factored for extensibility
  - Designed for tools

Common Language Runtime (CLR):
- Executes code, maintains security, handles component “plumbing” and dependencies

Operating System
- Base Framework
- Data and XML
- User Interface
- Web Services
- VB, C++, C#, J#, Python, ...

Common Language Runtime
- Unifies programming models across languages
- Enables cross-language integration
- Factored for extensibility
- Designed for tools
List of Languages

Ada
APL
Basic (Visual Basic)
C#
C
C++
COBOL
Component Pascal
   (Queensland Univ of Tech)
ECMAScript (JScript)
Eiffel (Monash University)
FORTRAN
Haskell (OGI)
Java Language
Icc
   (MS Research Redmond)
Mondrian
   (Utrecht)
ML
   (MS Research Cambridge)
Mercury
   (Melbourne U.)
Oberon
   (Zurich University)
Oz
   (Univ of Saarlandes)
Perl
Python
Scheme
   (Northwestern U.)
SmallTalk
Dim s as String
s = "authors"
Dim cmd As New SqlCommand("select * from " & s, sqlconn)
cmd.ExecuteReader()

string s = "authors";
SqlCommand cmd = new SqlCommand("select * from "+s, sqlconn);
cmd.ExecuteReader();

String *s = S"authors";
SqlCommand cmd = new
SqlCommand(String::Concat(S"select * from ", s),
sqlconn);
cmd.ExecuteReader();
String s = "authors";
SqlCommand cmd = new SqlCommand("select * from " + s, sqlconn);
cmd.ExecuteReader();
var s = "authors"
var cmd = new SqlCommand("select * from " + s, sqlconn)
cmd.ExecuteReader()
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
REPOSITORY.
   CLASS SqlConnection AS "System.Data.SqlClient.SqlConnection".
DATA DIVISION.
WORKING-STORAGE SECTION.
  01 str PIC X(50).
  01 cmd-string PIC X(50).
  01 cmd OBJECT REFERENCE SqlCommand.
  01 sqlconn OBJECT REFERENCE SqlConnection.
PROCEDURE DIVISION.
  -> Establish the SQL connection here somewhere.
MOVE "authors" TO str.
STRING "select * from " DELIMITED BY SIZE,
   str DELIMITED BY " " INTO cmd-string.
INVOKE SqlCommand "NEW" USING BY VALUE cmd-string sqlconn RETURNING cmd.
INVOKE cmd "ExecuteReader".
DclFld MyInstObj Type( System.Data.SqlClient.SqlCommand )
DclFld s Type( *string )
s = "authors"
MyInstObj = New System.Data.SqlClient.SqlCommand("select * from "+s, sqlconn)
MyInstObj.ExecuteReader()

assembly_external(name="System.Data.SqlClient.SqlCommand")
sqlcmdcharacter*10 xsqlcmd
Cmd x='authors'
cmd = sqlcmd("select * from "/x, sqlconn)
call cmd.ExecuteReader()
end
s←String.New ‘authors’
cmd←SqlCommand.New (‘select * from ‘, s.ToString σ) sqlconn
cmd.ExecuteReader

|s| := 'authors'.
|cmd| := SqlCommand('select * from ' + s, sqlconn).
cmd.ExecuteReader().
(let* (  (s "authors")  
    (cmd (new-SqlCommand (string-append "select * from " s) sqlconn)))  
(execute-command cmd))

Scheme

local
    s: STRING
    cmd: SQLCOMMAND
do
    s := "authors"
    create cmd("select * from " + s, sqlconn)
    cmd.ExecuteReader()
end

Eiffel

SqlCommand = create System.Data.SqlClient.SqlCommand(String,
    System.Data.SqlClient.SqlConnection);
query = sqlconn -> let{ s = "authors"; } in {
    cmd <- SqlCommand ("select * from "+s, sqlconn);
    cmd # ExecuteReader();
};;
ASP.NET

- Server-side *Web Forms* and *Web Services* components separate code from content
- Provides factored architecture
- Compiled
- No-touch deployment system
- Intelligent state management
.NET For Devices

- Same programming model
- Same set of tools
- Same application architectures
Visual Studio.NET

- “Drag and Drop” Web application and services development
- Fully supports .NET Framework
- Multi-language support
- Open, extensible development environment
  - Languages focus on compiler – not UI
  - Consistent set of tools across languages
Standards

- .NET is based on open Internet standards
  - HTTP, HTML, XML, SOAP, WSDL
- .NET itself is an open standard
  - ECMA - [http://www.ECMA.ch](http://www.ECMA.ch)
  - ECMA 335 - "Common Language Infrastructure"
    All core .NET Framework Protocols, Formats, & Interfaces
  - ECMA 334 – C# Language Specification
- UDDI: Directory for web services
- Co-Sponsors with Microsoft - HP, Intel, IBM, Ariba...
- WS-I: Web Services Interoperability Org
Shared Source Implementation of the CLI (ROtOR)

- source and binaries
- ref implementation of ECMA specs
- includes C# & JScript compilers
- subset of the V1 .Net Frameworks
- on both Windows and FreeBSD
- intended for research
- liberal license, no tainting
- available now
Web Services — What’s next?

Global XML Web Services Architecture (GXA)

- GXA defines a framework that augments the basic Web service with generic higher-level services required by many distributed applications and are not specific to a particular problem domain

- Design Principles:
  - Decentralization and Federation
  - Modularity
  - XML-based Data Model
  - Transport neutrality
  - Application domain neutrality

- Proposed standards - Microsoft, IBM, and others
Global XML Web Services Architecture (GXA)

http://msdn.microsoft.com/webservices

- **WS-Inspection**
  Aggregates references to pre-existing service description documents

- **WS-Attachments**
  Mechanism for encapsulating a SOAP message and zero or more attachments in a DIME message.

- **DIME - Direct Internet Message Encapsulation**
  Submitted a revised DIME spec as an IETF Internet Draft

- **WS-Referral**
  Enables the routing strategies used by SOAP nodes in a message path to be dynamically configured.

- **WS-Routing**
  Simple, stateless, SOAP-based protocol for routing SOAP messages in an asynchronous manner over a variety of transports like TCP, UDP, and HTTP.

- **WS-Security**
  Enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication.

- **WS-Coordination**
  Extensible framework for providing protocols that coordinate the actions of distributed applications.

- **WS-Transaction**
  Describes coordination types used with WS-Coordination
Web Services Enhancement

New .NET library for building ASP.NET Web services that use more advanced protocols

Includes support for:

- **WS-Security**
- **WS-Routing**
- **WS-Attachments** and **DIME**
The Microsoft View

- Web services will subsume the Grid
  - The Grid will be more about data and services than renting cycles
  - Therefore we need to evangelize web services in the e-Science community

- Data Grids will stress test web service infrastructure and tools
  - So we can do cool research at the same time!
Globus announced OGSA earlier this year

...proposed evolution of the current Globus Toolkit towards a Grid system architecture based on an integration of Grid and Web services concepts and technologies...

...during 2002 and into 2003, we will evolve the current Globus Toolkit towards an OGSA-compliant Globus Toolkit 3.0, working with our partners to address backward compatibility and transitioning issues...

Lots of encouragement from industry (Microsoft, IBM, Oracle, Sun)

GGF as forum for discussion

Challenges:

Keeping OGSA compatible with mainstream

Filling in gaps
Engagement with Grid Community

- **Goal:** GXA as infrastructure for Grids
- **Working with Globus & GGF**
  - Funding work at Argonne National Lab (Globus)
  - Globus Toolkit 2 on Windows
  - OGSA for .NET (prototyping)
    - [http://www.globus.org/ogsa/](http://www.globus.org/ogsa/)
  - GGF6
    - OGSA for .NET BOF
- **HPC .net kit – see**
  - [http://www.microsoft.com/HPC](http://www.microsoft.com/HPC)
    - Part of .net server scale out development
Data Grid Research

- MSR Working with key DataGrid projects
  - Jim Gray and Tom Barclay leading the charge
  - Intellectual leadership in application of database technology, web services and .net power tools

- There’s research to be done too - challenging problems are
  - publishing data
  - Advanced query techniques e.g., spatial indices
  - Visualization
Summary

Willing to partner with researchers willing to explore the use of Windows and .NET software in Clusters and Grids.

Interested in working with other scientific communities that have large datasets or processes to expose as Web Services.

Encourage you to evaluate and track commercial tools.
Thank You