Distributed Remote Agents and Global brokers Network (DRAGON) for DSN and Missions

Yeou-Fang Wang, Ph.D.

11/4/02
Agenda

- Motivation/problem definition
- Approach
- Software architecture
- Capabilities
- Current developments
- Conclusion
Acknowledgement

- **Sponsors:** RAPSO, SPS, PTM for MSL (earlier version)
- **Developers (part time):**
  - Yeou-Fang Wang
  - John Baldwin
  - Luke Voss (APT)
Motivation / Problem Definition

• Single computation engine cluster for multiple users/platform/application/devices
  – Users: RAPSO, DSN scheduling, SPS, projects, spacecraft designers
  – Client platforms: Windows, Solaris/UNIX, Mac, Linux
  – Applications: Web browser, GUI apps, …
  – Devices: Desk-top, lap-top, hand-held

• Application linkage / tool integration

• Centralized installation for algorithm changes
Approach

- Service-centric distributed system using Web services (vs. GUI-centric or database-centric)(info vs. data)
- Multi-tier
  - User/program interface
  - Computation engine / business logic
  - Data access
- Encapsulation: between tiers, objects, ...for min dependency.
- Namespaces:
  - Ground system: ground resource, forecast, scheduling
  - Project: mission, event, telecom, power, ....
- Secured Web services: HTTP/XML, standard, text
- Ownership – go to the right source for logic and data, no data stored locally
Agents & Brokers

- **Agents**: Intelligent decision makers
  - Expert in the field (e.g. Scheduler agent to generate schedule)
  - Possible learning capability

- **Brokers**: Connection to legacy software or possible hardware
Access Model

Web pages → Web servers → Web services calls → Dragon server cluster → Data access objects

Legacy system → Broker wrapper 

DB → Files

Legacy system

MS-SQL, Oracle, ...

Text, Excel, Word, ...
Software architecture

Legacy systems

User interfaces
- Multi-platform clients
- Web browsers
- Windows apps

Business Logic (DRAGON)
- Brokers
- Job/workflow manager
- Model Core
- Agents
- Web Services server
- Data Access objects

Data
- DB
- Controlled Files
Current Implementation

- Namespace: JPL.DRAGON
- The concept is platform-independent
- Current implementation is based Microsoft .NET using C#, CLR, and ASP.NET
Current Capability (JPL.DRAGON)

- Time: DST, week/year, current time
- Spacecraft: S/C number, names
- Viewperiod: combining VP data from forecast, project, NSS, ...
- Ground resource: config code, equipment, inventory, ...
- Scheduling: NSS real-time, planning, conflict checking (partial)
- Excel: get value from Excel, compute in Excel
Sample:

Ground Resource Information

The following operations are supported. For a formal definition, please review the Service Description.

- **GetAntennaEquipmentListString**
  Returns the list of all antenna equipment type codes as a hash (#) separated string

- **GetEquipmentString**
  Returns the equipment quantities associated with a configuration code as a string

- **ServiceAvailable**
  Returns true if the service is available, false otherwise

- **GetComplexEquipment**
  Returns an array of equipment structures associated with a complex configuration code

- **GetAntennaEquipmentString**
  Returns the equipment quantities associated with an antenna configuration code as a string

- **GetComplexInventoryString**
  Returns the equipment inventory at a complex as a string

- **GetEquipment**
  Returns an array of equipment structures associated with a configuration code

- **GetAllConfigCodesInfoString**
  Get all config codes and their equipment requirements in a string
Schedule viewing

The Web page updates every few minutes.

Same GUI code for both Web browser and windows app.
Excel Access -- multiplication

(1) A & B

(2) A & B

(3) C = A * B

(4) C = A * B

Excel Workbook (server)
Engineering focus

DRAGON Software focus

Web pages

Applications

Excel (client)
Current developments

- Security (authentication & authorization)
  - SOAP header (user name & password)
  - SSL
  - Role-based
  - Individual function level (rather than Web service level)
  - Excel security: macro, document check-in/check-out

- Admin: user accounts, roles, function access, status, abort

- Workflow: calling sequence, notification, alerting

- Increasing function collection
Conclusion

- Service-centric servers for multiple usages
- Remote access through Web services (SOAP/XML)
- Local application access through library/DLL
- GUI components for assembly
- Welcome others to join JPL namespace development (JPL.NAV, JPL.IMAGE, ...)