

Summer 2012

Jet Propulsion Laboratory/California Institute of Technology

Lee, Carlyn-Ann (332H-Affiliate), Mentor: Kar-Ming Cheung and Charles Lee

[REST ARCHITECTURE FOR LINK ANALYSIS TOOLS PORTAL]¹

Abstract: A web portal is under development to improve overall services provided by NASA networks that support near-Earth missions. These networks are managed by Space Communication and Navigation (SCaN) Program of NASA and include the following: Deep Space Network (DSN), Space Network (SN), and Near-Earth Network (NEN). DSN is a network of antennas located in three longitudinally separated sites and supports interplanetary spacecraft missions, radio science, and radar astronomy observations. SN consists of a number of geostationary Tracking and Data Relay Satellites (TDRS) and associated ground stations. NEN consists of both NASA-owned ground stations and those owned by international, commercial, and academic partners. This study investigates different web access techniques for mission users to use link analysis tools that aid in operating NASA networks. We developed a web portal prototype following principles of Representational state transfer (REST) architecture to leverage scalability, independent deployment of tools, and generic interfaces. In REST architectures, requests and responses between clients and servers are built around transferring representations of resources that can be addressed in memory. We demonstrate the feasibility of using REST architecture to facilitate web-service transactions via web-portal.

¹ This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the Year-Round Internship Program and the National Aeronautics and Space Administration.

1. Introduction

This purpose of this study is to (1) investigate different web access techniques for mission users to utilize link analysis tools for operating NASA networks, and (2) to develop a web portal prototype. The web portal prototype development considers the following features: scalability, independent implementation and deployment of tools, and generic and reusable interfaces. To achieve these goals, the prototype development follows the principles of REpresentational state transfer (REST) architecture. In REST architecture, requests and responses between clients and servers are built around transfer of representations of resources that can be addressed in memory.

The objective of this study requires converting a stand-alone link analysis tool developed in Excel/Visual Basic into a web service that can be invoked using HyperText Transfer Protocol (HTTP) and principles of REST. We demonstrate the feasibility of using REST architecture to facilitate web service transactions via web-portal, mainly through keeping the web service decoupled from the portal.

2. Background

The Deep Space Network (DSN) is a network of antennas located in three longitudinally separated sites. DSN supports interplanetary spacecraft missions as well as radio science and radar astronomy observations. The DSN provides vital communications and tracking services that control the spacecraft in deep space, and brings back scientific information collected by the science instruments onboard the spacecraft. Other NASA networks that support near-Earth missions include Space Network (SN) and the Near-Earth Network (NEN). SN consists of a number of geostationary Tracking and Data Relay Satellites (TDRS) and the associated ground stations. NEN consists of both NASA-owned ground stations and those owned by international, commercial, and academic partners. The DSN, SN, and NEN are managed by the Space Communication and Navigation (SCaN) Program of NASA. A web portal is under development to streamline the operations, and to improve the overall services provided by the NASA networks.

For a portal application, it is desirable to minimize coupling between the server and the many clients that operate at different geographically dispersed locations. Decoupling allows for regular updates to the server to be performed without requiring updating the client software. Consistent use of hypertext transfer protocol (HTTP) in REST architecture renders web services that can be easily consumed by a client developer [Fielding]. REST is currently the predominant architecture for distributed

systems such as the World Wide Web.

3. Objectives

This goal of this study is to develop a prototype service for easy web access to link tools. This project is one step in a larger effort to centralize the tools developed to support SCA_N and missions that utilize the SCA_N services. The web portal shall be a point-of-access to the tools required for operations related to the DSN and other antenna networks. The time span for prototype portal development is ten weeks. Figure 1 summarizes the basic requirements for this project with a use case model.

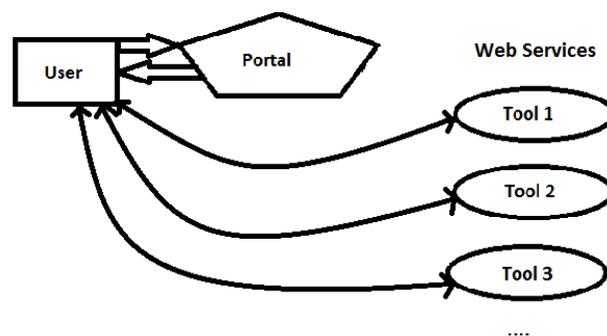


Figure 1: Use Cases Model for the web portal prototype. The portal shall serve as a point of access for the user to interact with a number of tools. These tools already exist as excel workbooks and Matlab scripts.

4. Approach

The first tool that was made available from the portal is the Easy Link Tool V0-17b. This tool exists as an excel worksheet with Visual Basic for Windows Applications (VBA) macros that perform RF link calculations from user input. The stand-alone tool followed a model-view-controller pattern. The logic layer of Easy Link Tool V0-17b was converted from VBA to VB.NET and a service layer was developed so that the tool can be accessed as a web service via REpresentational State Transfer (REST) Application programming interface (API) calls. In this implementation of REST, the portal generates a Uniform resource identifier (URI) from the user input/file and returns the URI to the user. The user can invoke REST calls to the tools directly using the URI, such that the portal and tools are decoupled. This is advantageous if tools need to be modified in the future. Furthermore, tools can be distributed to multiple users and users that have flexibility to script function calls directly to the tools.

One advantage of rendering link analysis tools as APIs is the ability to integrate their services into existing software. Services can be utilized

from any device connected to the internet, including mobile devices. Functions can be implemented in loops, plots, and used for more complex analyses. In addition, service redundancy can be employed to improve performance. A server farm can be built with redundant web service functionalities for load-balancing if multiple clients require the same features. The proposed architecture is summarized in the diagram in figure 2.

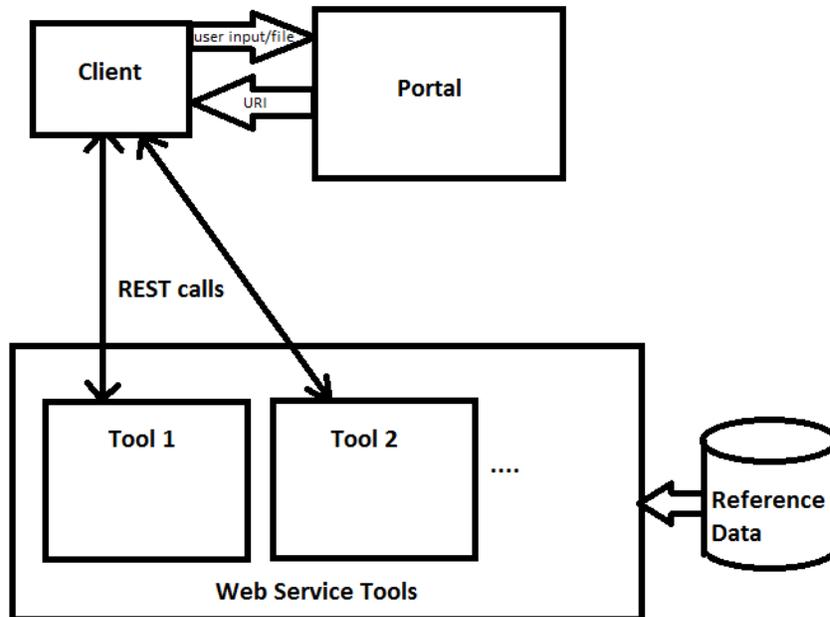


Figure 2: A diagram of the portal's architecture.

The most straightforward approach to converting the Easy Link tool into a web service was to rewrite VBA macros in the logic layer with VB.NET. The conversion from VBA to VB.NET required syntax changes and modifications of variable scopes. One caveat with using .NET framework tools is that the service must be hosted on a machine with Microsoft Internet Information Services (IIS).

For convenience, the portal was written with Active Server Pages (ASP), which requires IIS. The portal provides forms for user input and submission buttons for making web service calls. However it is possible to utilize other solutions to a portal. Due to its minimal coupling with the web service, the portal can rest on a different server than the web service. A mock up portal prototype with access to forms is shown in figure 3

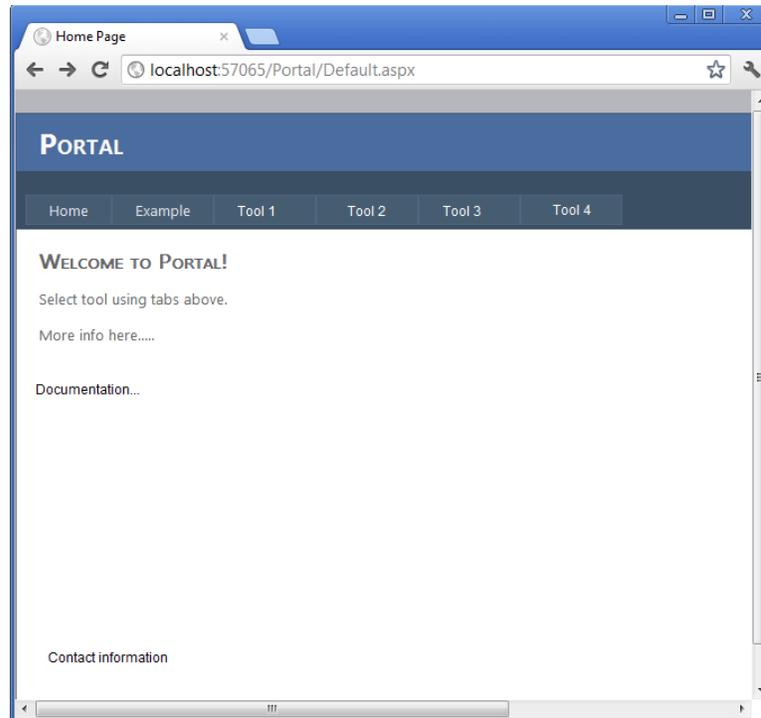
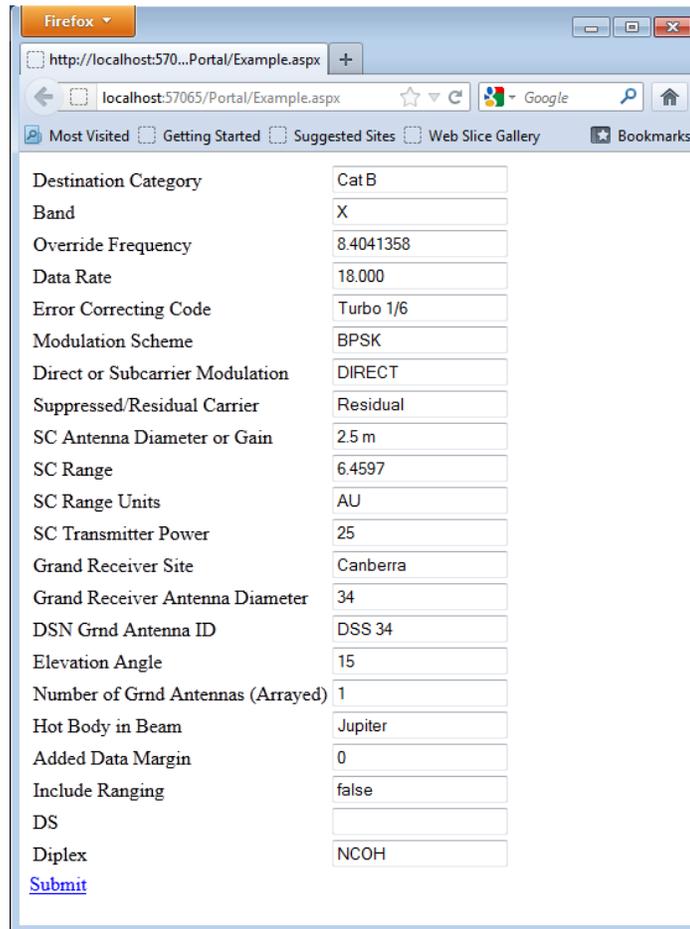


Figure 3: Mock-up of portal home page contains general information, documentation, contact information. Forms for available tools can be accessed via hyperlinks listed along header.

A structured query language (SQL) database was set up to contain all reference tables. Queries to the tables are performed using an adapter class that returns data from the data tables to the data service layer. This adapter class keeps the tools loosely coupled from the data tables, in case modifications need to be made.

In the current implementation, Javascript was used to script minimal client side features. These features include: hard-coded default values, construction of URIs, and submission of API calls to the server. Additional client side features such as drop-down menus, radio buttons, and interactive interfaces can be added in the future. The client form is shown in figure 4 and a sample client request to the server is shown in figure 5.



Destination Category	Cat B
Band	X
Override Frequency	8.4041358
Data Rate	18.000
Error Correcting Code	Turbo 1/6
Modulation Scheme	BPSK
Direct or Subcarrier Modulation	DIRECT
Suppressed/Residual Carrier	Residual
SC Antenna Diameter or Gain	2.5 m
SC Range	6.4597
SC Range Units	AU
SC Transmitter Power	25
Grand Receiver Site	Canberra
Grand Receiver Antenna Diameter	34
DSN Grnd Antenna ID	DSS 34
Elevation Angle	15
Number of Grnd Antennas (Arrayed)	1
Hot Body in Beam	Jupiter
Added Data Margin	0
Include Ranging	false
DS	
Diplex	NCOH

[Submit](#)

Figure 4: Portal provides form for user input. Clicking "submit" generates a URI from user input and invokes API to call function from web service.

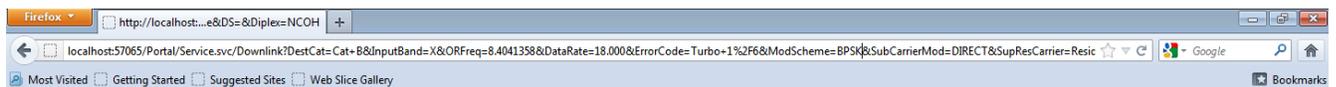


Figure 5: API call to function invoked from URI.

To reduce the overhead included in XML formats described in [Doneva], JavaScript Object Notation (JSON) was used for data-interchange from server to client. Server responses are sent to the client in an easily parsable JSON format shown in figure 6.

```
{
  "calculatedDTEdownlinkResult": {
    "ErrorCondition": false,
    "oAtmosphere": {
      "CD": 0.9,
      "ZenAtten": 0.058,
      "oFreeSpace": {
        "CosmicBackgroundTemp": 2.725,
        "Destcat": "Cat
      }
    },
    "Frequency": 8404135704.0405273,
    "Sep": 966357376931.91052,
    "oGroundAntenna": {
      "AntennaDiam": 34,
      "AntennaEff": 0.74,
      "CircuitLossdB": 0,
      "DiplexMode": "wcom",
      "ElevationAngle": "15",
      "G0ReceivedB": 68.33,
      "G0TransmitdB": NaN,
      "G1dB": 4.5E-
    }
  }
}
```

Figure 6: Function returns JSON string containing output.

Discussion/Conclusions

The model view controller pattern was conducive to developing a web extension of Easy Link Tool V0-17b. Specifically, the logic layer was easily converted from VBA to VB.NET. The service layer for the web service was developed to interface with the client and logic layer. Minimal coupling between the interfaces made the conversion to a web service feasible. In summary, the REST architecture allows for decoupling of services from different clients, web service tools, and portal, which can be hosted on different servers and different locations.

5. Acknowledgements

This project could not have been completed without the help of my mentors Kar-ming Cheung and Charles Lee, as well as Bruce MacNeal and David Morabito, who developed the stand-alone Easy Link tool. I would like to thank George Chang and Hook Hua for their invaluable advice on developing REST architecture, Don Ning for setting up an IP and port forwarding, and Hunter Zhao for numerous and insightful conversations.

6. Bibliography

Doneva, S. (2011). Deep Space Network-Wide Portal Development: Planning Service Pilot Project. Jet Propulsion Lab/California Institute of Technology (Final Report).

Fielding, R.T. (2000). Architectural Styles and the Design of Network-based Software Architectures. (Doctoral dissertation). Retrieved from <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>