The Deep Space Network (DSN) is an international network of antennas that supports interplanetary spacecraft missions and radio and radar astronomy observations for the exploration of the solar system and the universe. DSN provides the vital two-way communications link that guides and controls planetary explorers, and brings back the images and new scientific information they collect. In an attempt to streamline operations and improve overall services provided by the Deep Space Network a DSN-wide portal is under development. The project is one step in a larger effort to centralize the data collected from current missions including user input parameters for spacecraft to be tracked. This information will be placed into a principal repository where all operations related to the DSN are stored. Furthermore, providing statistical characterization of data volumes will help identify technically feasible tracking opportunities and more precise mission planning by providing upfront scheduling proposals. Business intelligence tools are to be incorporated in the output to deliver data visualization.

The Deep Space Network (DSN) the largest and most sensitive scientific telecommunications system in the world. It is composed of three spacecraft tracking centers situated approximately 120 degrees apart around the world in Goldstone California, Madrid Spain and Canberra Australia. Each center consists of a 70 meter satellite antenna and a few more 34, and 26 meter ones to enable and ensure continuous communications for multiple spacecrafts. Among the functions sustained by the DSN is to acquire telemetry data from spacecrafts, transmit commands to spacecraft, track spacecraft position and velocity, perform very-long-baseline observations, measure variations in radio waves for radio science experiments, gather science data, and monitor and control the performance of the network. While NASA's scientific investigation of the Solar System is being accomplished mainly through the use of unmanned automated spacecraft, each center tracks spacecraft up to the far reaches of space 24 hours/7-days/week and is manned by human operations.

In an attempt to rationalize operations and improve overall services provided by the Deep Space Network a DSN-wide portal is under development. One of the services envisioned for the portal is to provide a high-resolution planning service to flight projects supported to understand the potential return data volumes and associated variances of DSN provided communications. As the number of missions has increased, the capability to automate and precisely plan for the use of network services will become ever more paramount. Additionally, the high resolution predictive information will support a higher degree of automation thereby allowing for a reduction in human labor involved with translating from planning phases to the execution phase/usage of the NASA tracking networks.

The solution will make use of the SPS infrastructure which supports a readily scalable compute intensive architecture. Mission planners and designers will eventually be able to submit hypothetical trajectories and telecommunications profiles and receive detailed reports and analysis with regard to predicted data communications supportable by the DSN. With proper metadata tagging etc., DSN customers will in the end be able to indicate the potential communications plans as the basis for drawing up service agreements and/or developing a schedule of services with the DSN.

The problem is not being solved by this development alone. This is a piloting effort to gauge the potential for such a solution. The essential problem is to provide a high-resolution planning
service to flight projects supported by the DSN to understand the potential return data volumes and variances to support utilization of DSN provided communications. However, currently there is no closed loop, from an information system perspective, between planning and capture of service agreements and development of schedules. In other words, the information gleaned from planning cannot be used to seed subsequent service management processes for the DSN, especially so in any automated fashion. This final solution, will eventually, in contrast, allow for precise predictive information to be used as the basis for drawing up service agreements and developing a subsequent schedule of services.

DSN information composed from the above mentioned centers is collected into a centralized data warehouse, and used to analyze the feasibility of existing as well as future missions. However, it is currently being accessed by two separate interfaces a portal that provides a summary of data from current missions, and a separate interface that enable users to input parameters for desired spacecraft to be tracked.

The focus of the development over the past ten weeks is to provide the necessary data processing to merge and report on two internally DSN produced information objects. These are objects that are currently computed by the DSN and, with sufficient modification and subsequent presentation can be combined into a much more useful information object for the DSN customers. In particular, this is a coalescing of the mutual visibility predictions calculations and the telemetry predictions capabilities, and then treating this as raw data for a business intelligence reporting platform to provide detailed visualization for DSN customers.

The project outcome was to combine the two sources of information into one centralized interface that will enable all operations related to the Deep Space Network to be conducted within a principal repository. While the large scale project will provide true planning capabilities the prototype operates on a data sets already calculated and documents previously produced, the initial version aims to combine the communication geometry and data rate estimates. Thus, based on what has already happened we can draw up a prediction of what it is to be expected. Although on a smaller scale, combining the two separate data channels will help lay the foundation for more precise mission planning and the ability to provide upfront scheduling proposals listing the most feasible time for a mission to be tracked.

The image below graphically describes the objective of the project completed during the past ten weeks. Taking the communication geometry contained within the ViewPeriod file and combine with the data rate estimates found in the Telemetry Prediction file in attempt to produce a more cohesive planning report showcasing certain elements found in the above two documents. Both data sources output information in XML raw format, further formatted to HTML for better human readability and more user-friendly environment. The output utilizes the flexibility and purity of XML as well but uses more sophisticated and modern techniques.
The individual steps taken during the development are described below:

The first step taken to accomplish the objective of the project was to familiarize myself with Downlink Equipment Monitor and Service Preparation Subsystem Interface documentation as well as the DSN Interface to Flight Projects for Viewperiod Predictions documentation. In addition I have acquainted myself with SPS portal, and more specifically how to obtain specific mission information from Viewperiod and UTP files and find developer resources. Seeing the specific data in those files helped me gain a better understanding of possible data manipulation through code and a format of the desired output.

The next stage was to decide on specific technologies to be used in order to turn the project plan into reality. I have investigated and experimented with various technologies and tools to assist in accomplishing the goal. Some of these are GlassFish application server, Maven project management tool, XMLBeans, Jersey JAX-RS API, XQueryXPath plugin, Java Persistence API, XQJ API, Saxon and StAX APIs. The final decision to use Apache’s Jersey JAX-RS API library was driven by the fact that the most suitable approach to the problem was to build it using RESRful webservies, and Jersey had everything needed. Used ALTOVA XML Spy Enterprise Edition 2011 for building the XSLT files and reading the existing XML files schemas.

In an attempt to build a sustainable test environment on my local machine I have installed and configured Apache Tomcat 5.5.33 since it is the same as the server machine used in production environment. The choice of backend machine drove my decision to
use NetBeans IDE 7.0 as the development environment since it provided a convenient compatibility and lowered testing efforts by starting the server automatically on each program invocation.

The next phase was starting the actual development by building a RESTful web service within NetBeans using Jersey and Apache Tomcat. The webservice holds all the computation as well as calls to other RESTful webservices to obtain specific files to be parsed and data within used. The parsing is done by using Java’s xPath API package and by building xQueries to extract specific desired data. The main decision driving this choice was fast execution and low memory usage. At the beginning of the project I had all the XML files static on my hard drive and focused on making the coding work regardless of the validity of output. Later I substituted for real mission data by building dynamic URLs based on user selection to call the existing RESTful webservices and parse the returned XML file to extract desired data. This allowed me to test the validity of calculations on a broader range.

I have also created Java classes to mirror major data components and their proper functionality. The main issue here was deciding between creating a highly cohesive objects and doing so only to the point which does not impact performance. The main reason for using xPath/xQuery is the high speed and low memory usage required to parse an XML file. xPath works in a way that as soon as it reads a piece of data it releases all resources used, and by using classes to mirror data components I am storing the above read objects in memory which reduces the main benefit of using xPath. It took some trial-and-error to find the balance between the two.

The processing unit of the project consists of calculating data return bit rate per tracking pass dependent on symbol rate and coding scheme, as well as totals for certain periods of time. Calculated total and average data return volume on various levels, and provided ability to determine and display details for the most and least feasible tracking passes in terms of data volume return.

I have decided on dynamically build an output XML file that contains selected data from View Periods, and Telemetry Predictions files as well as calculated statistical data to be used in the output. This file is being temporary stored and used by XSLT to output the results for the Summary and Details pages (screen shots below). By using multiple XSLTs to be parsed in the output web pages based on one output XML I gained a very good output performance. Multiple XSLTs also allow for various design considerations and open choices in terms of data delivery, readability and improved user acceptance. To put it all together I have creates Java Server Pages to parse the XSLT and display the output as well as control code invocation – this part handles the relation between the web page and its associated XSLT as well as the calls to certain methods to load data needed for output.
Summary of calculated statistical data:

Details for the statistical data calculated:
The initial, login, mission and viewperiod selection pages, are designed to gather user input and thus are built with JSP in plain HTML invoking pieces of Java code when needed. The lack of significant processing drove the decision of a simple solution backed up with great performance.

In addition to the XML output file used to display results I am also building a CSV file to be fed into MicroStrategy and provide various visualization of the data which will help users utilize data more easy and quick. While business intelligence tools like MicroStrategy work best on large data sets, it turned out to be quite useful for a smaller set of data as well. For example the graph below shows how the data return volume varies by a tracking pass(red) as well as how quickly it can build up by showing the cumulative data volume in blue.

Putting all the pieces together has successfully turned into a sustained working unit. Detailed project flow diagram is shown below.
Login Page

User enters username/password.

- Authentication successful?
  - No: Cannot proceed. Controls are disabled.
  - Yes: User selects mission

- Mission selection is valid?
  - No: Error; need valid mission selection.
  - Yes: Make a RESTful call to return, parse and display all ViewPeriod files for that mission.
In conclusion I wish to express my heartfelt thanks to all the people who helped turn this summer into an incredible experience. First and foremost I would like to thank my mentor, Paul Wolgast for bringing me aboard and providing this great opportunity. He also ensured the stay is educational as well as fun. Next, I would like to thank Erik Barkley, my co-mentor, for the great project proposed and incredible help in every step from conception to completion. Bach Bui for helping build the webservice and select the proper tools to make it happen, Shantanu Malhotra for the numerous detailed explanations of antenna operations and data gathering processes, George Chang for providing the RESTful calls URLs as well as providing the solution to SSL authentication and connection through code, Silvino Zendejas for providing documentation and
lessons about MicroStrategy as well as a hands-on help on MicroStrategy specifics. Syed Sadaqathullah for setting up a MicroStrategy development environment for me. In addition I would like to thank Rhonda Lyons for ensuring we have the equipment needed in a timely manner, and the education office for the number of activities organized. I would also like to mention my fellow interns Andrea M Connell, Ben-Han Sung, Alexander Valderrama, Jonathan Yee, and Kenny Lei for great collaboration and enjoyable time together.

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