The Deep Impact Network Experiment Operations Center Monitor and Control System

Shin-Ywan (Cindy) Wang, J. Leigh Torgerson, Joshua Schoolcraft, Yan Brenman
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109

Abstract—The Interplanetary Overlay Network (ION) software at JPL is an implementation of Delay/Disruption Tolerant Networking (DTN) which has been proposed as an interplanetary protocol to support space communication. The JPL Deep Impact Network (DINET) is a technology development experiment intended to increase the technical readiness of the JPL implemented ION suite. The DINET Experiment Operations Center (EOC) developed by JPL’s Protocol Technology Lab (PTL) was critical in accomplishing the experiment. EOC, containing all end nodes of simulated spaces and one administrative node, exercised publish and subscribe functions for payload data among all end nodes to verify the effectiveness of data exchange over ION protocol stacks. A Monitor and Control System was created and installed on the administrative node as a multi-tiered internet-based Web application to support the Deep Impact Network Experiment by allowing monitoring and analysis of the data delivery and statistics from ION. This Monitor and Control System includes the capability of receiving protocol status messages, classifying and storing status messages into a database from the ION simulation network, and providing web interfaces for viewing the live results in addition to interactive database queries.

Keywords-Deep Impack Network (DINET); Delay/Disruption Tolerant Network (DTN); Interplanetary Overlay Network (ION); Bundle Protocol (BP); Experiment Operations Center (EOC); Asynchronous Message System (AMS); Protocol Technology Lab (PTL).

I. OVERVIEW OF DELAY AND DISRUPTION TOLERANT INTEROPERABLE NETWORKING (DTN)

A delay-tolerant network is a network architecture designed to operate effectively as an overlay on top of regional networks, or as an interplanetary internet. The interaction among regional networks can be characterized by long delay, asymmetric data rate, intermittent connectivity and high error rate due to the extreme environments and distances encountered in deep space communication at an interplanetary scale. In order to overcome stated problems, DTN mainly uses store-and-forward service to move data from the source to the destination. The DTN bundle protocol (BP) forms an end-to-end message-oriented overlay network between the transport layer and the application layer. Devices implementing the bundle layer are called DTN nodes. This network layer uses bundles as the message data units to accomplish the hop-by-hop transfer, reliable delivery and optional end-to-end acknowledgement through a DTN. The bundle protocol also includes the diagnostic and management features, a flexible naming scheme capable of encapsulating different naming and addressing systems, and a basic security model.

II. OVERVIEW OF THE DEEP IMPACT NETWORK EXPERIMENT

NASA is working to raise the Technology Readiness Level of DTN for the purpose of infusing it into space exploration missions and supporting a ground and space-based relay infrastructure. The JPL Deep Impact Network Experiment (DINET) was a technology development experiment intended to increase the technical readiness of the JPL DTN implementation, Interplanetary Overlay Network (ION) software. It was the first DTN experiment performed in deep space.

DINET utilized the Deep Impact spacecraft as a flying testbed for DTN validation. In the DINET topology, the Deep Impact spacecraft acted as a router to pass the payload data from one simulated space region to another. There three regions were Earth, Mars, and Phobos and included three end nodes to produce and consume the payload data as shown in Figure 1. In the network, the data delivery starting from one end node was first routed to the ground system of that end node region, then to the Deep Impact spacecraft via RF links, then to the ground system of the destination region, eventually arriving at the destination end node.

Figure 1. DINET virtual topology

Copyright 2009 California Institute of Technology. Government sponsorship acknowledged.
The simulated network regions and associated hardware were located within the Experiment Operations Center (EOC) where DINET operations were performed. The ground system terminating each region with the spacecraft is referred to as DSOT (Data System Operations Team). The DINET DTN node connectivity is depicted in Figure 2. More details about network connectivity and data delivery are contained in section III.

**III. DINET EXPERIMENT OPERATION CENTER (EOC)**

The DINET (EOC) developed by JPL Protocol Technology Lab (PTL) as shown in Figure 3 was critical to the DINET experiment. EOC, containing all end nodes of simulated spaces and one administrative node, exercised the publish and subscribe functionalities for payload data among all end nodes to verify the effectiveness of data exchange over the ION communications stack. The ION based Monitor and Control System is an innovative,
intranet-based software application which provides live network health and performance statistics. With this system, as illustrated in Figure 2, these ION software diagnostic messages and protocol status messages issued by network nodes are collected, analyzed, stored into a database in real-time. This application also provides a web interface for viewing the data update and topology updates in real-time and for interactive database queries.

IV. MONITOR AND CONTROL SYSTEM

The Monitor and Control System is designed as a multi-tier client-server web application. A middleware process services the data requests between the database and the user. The multi-tier hierarchical structure makes the application extensible and distributive. The implementation of the Monitor and Control System in the EOC utilized the technologies of Java, JavaScript, AJAX, JSP (JavaServer Page), and a MySQL database to provide the means for the DINET experiment operator to monitor ION bundle delivery and operational status effectively. Moreover, the devised infrastructure of the DINET Monitor and Control system can be extended to monitor the operation status, data transmission and to update the display of the topology for other space mission simulations.

A. Software Modules

Figure 4 shows that the Monitor and Control System in the DINET EOC consists of the following major executable pieces: DINETLogProcessor, DINETAdm, DINETQueryAdm, DINETTopology, Update_Receiver and Servlet. DINETAdm and DINETQueryAdm are the implementation of a Web GUI running on the client side. Update_Receiver and Servlet run on the Web server. Servlet is primarily designed for use with the web-based HTTP protocol to process and return client requests. For the devised Monitor and Control System, the client-side Web GUI sends a request to the server’s servlet. The servlet then interacts with all relevant computational libraries, analyzes the data, and returns the result to the client GUI. DINETAdm and DINETQueryAdm dynamically update the GUI contents as network operations proceed. The Update_Receiver which is started at the Web server initiates a TCP socket connection to receive message and topology update notices respectively from DINETLogProcessor and DINETTopology. It then responds to Servlet’s request if any message or topology update notification has been received. DINETLogProcessor sets up a TCP/IP socket connection to all nodes to receive the ION status messages, parse and categorize the messages, and deposit messages contents into the SQL database. Additionally, DINETLogProcessor sends signals to the Web server via TCP/IP socket as notification of the arrival of new status messages. DINETTopology is a topology description generator which dynamically updates the network topology description in xml format based on the ION network configuration, which is initialized prior to mission operations and updated during the course of the mission whenever the network’s connectivity changes. Figure 5 shows an example of topology description created by DINETTopology. The Web GUI of the Monitor and Control System displays the DINET topology in a matrix style as
depicted in Figure 6. The row and column location of each node on the matrix is specified in the description file. Each connected link which contains two end nodes is also described in the xml description file. Similar to DINETLogProcessor, DINETTopology notifies the Web server of any changes in the network topology via TCP/IP socket. DINETAdm is a web-based graphic display system which updates the display of the network topology and the ION status messages in real-time by sending the request to and getting the returns from the server’s servlets. As a companion to DINETAdm, DINETQueryAdm provides another web-based graphic display which interactively accepts and responds to users’ queries on ION status messages.

B. Supporting Libraries

Under the major executable pieces, there are several libraries constructed to support specific functions. They are LibDatabase, LibLogProcessor, LibSocket, LibLogProvide and LibTopology. The functions of each library are described as follows.

LibDatabase
- provides functions to manipulate and interface with the database, such as
  o setting connections to the database,
  o creating tables in the database,
  o inserting items into database tables,
  o deleting items from the database tables,
  o querying table content by giving entry criterion.

LibLogProcessor
- builds the database and tables in order to store the contents of received log messages
- parses and classifies messages,
- stores message contents into the database.

LibSocket
- initiates the socket connection on the server side
- receives messages from remote clients.

LibLogProvider
- interprets queries issued from the application, i.e. DINETQueryAdm,
- sends queries to the database by interacting with LibDatabase.
- returns the query result to the application.

LibTopology
- parses xml topology description
- draws the network topology using a Java applet.
C. Types of Log Messages

The Monitor and Control System in the EOC received messages from all nodes in DINET during mission operations. The messages included ION software status messages, ION bundle status report messages, and node publish/subscription messages. All messages are formatted in consistent plain text style prior to arriving at the processing unit of the destination, i.e. DINETProcessor at the administrative node in the EOC.

Bundle status reports - administrative messages in the form of DTN bundles as defined in bundle protocol - are used to inform of the bundle processing status and are issued by all nodes in the network and destined for the administration node via the ION stack. At the administration node, the 'ipnadminep' software converted bundle status reports into the text-style log messages, and further redirected them to the processing unit of the DINET Monitor and Control System. Bundle status reports provide information about the deletion of any undelivered bundle in the network and thus allow the administrative node to initiate some specific action in reaction upon reception of the message.
ION software status messages are logged by all nodes and programmatically redirected to the administrative node by ION logging functions via TCP/IP backchannel, not via ION stack. ION status messages are of five general types: informational, warning, diagnostic, communication statistics, and other (miscellaneous) status messages. Informational messages mainly inform of the occurrence of events nominal but significant software events. The warning messages inform of the occurrence of events that are off-nominal but are likely due to configuration or operational error rather than software failure. Diagnostic messages inform of the occurrence of events that are off-nominal and might be due to software errors. A statistics message provides network performance statistics over a stated interval. An “other status message” is seen in response to specific operator-initiated activity. The publish/subscription messages denote the publish and subscription status from the endpoint nodes involving the action of publish/subscription.

Messages issued by the Deep Impact spacecraft were encapsulated in EVR telemetry packets. As the ground system received these telemetry packets, it extracted the log messages from the encapsulation and re-transmitted the messages to the administrative node via TCP/IP socket connection.
All these status messages and statistics helped the mission operations staff keep abreast of the network data flow and experiment status.

D. Web GUI

The Monitor and Control System primarily provides two types of web GUI pages, the main GUI page and the query GUI page. They allow the user to activate experiment operation, monitor operational status, and send queries to get stored messages.

Figure 6 is a screen shot of the main GUI page. In this page, either an existing experiment name or a new experiment name is selected for executing operations. There are 11 tables on the page which display live messages received from DINET nodes. The message display is started by selecting the “start/update” button and temporarily paused by selecting the “pause” button. All messages received and stored in the database can be dumped into a text file by selecting the “save” button and entering a file name. The top and middle of the page shows the network topology which is updated dynamically as the connectivity changes.

Figure 7 shows an example of the GUI query page. In order to query the messages stored in the database, the user specifies an experiment name, (optional) message type, message source node, number of messages, and period of message creation time. When the “query” button is clicked, the messages stored in the database which meet the query criterion are extracted to be displayed in the query page, and those resultant messages can also be dumped into a text file by clicking the “save” button and entering a file name. The table content and topology display on both web GUI pages are locally and dynamically updated without interfering with the overall display of both pages. If the total amount of messages which meet the query criterion exceeds a certain amount, the messages returned from the Web server may slow down the GUI display. Therefore, the servlet at the Web server is designed to send the information of the qualified messages in separate batches once the message return exceeds the aforementioned certain value for the purpose of displaying the Web GUI content efficiently.

VI. FUTURE DIRECTION

The Monitor and Control system is designed as a multi-tier client-server web application. The multi-tier hierarchical structure makes the application extensible and distributive. For example, the number of operational nodes, log message type, performance data instrumentation and storage can be expanded. The GUI page displays can be rapidly modified to suit based on the current multi-tier infrastructure.

This devised infrastructure of the Monitor and Control system can be further customized to monitor the operation status, data transmission, and the topology updates for varied simulated space network missions. The topology update can be enhanced to be in more animated and realistic style in which not just the connectivity of the topology but also the relative location and status among network elements is displayed. In addition, a distributed network recovery system can be further developed based on diagnostic messages received from network nodes to check the health of the network and heal repairable errors in an automated fashion.

ACKNOWLEDGMENT

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology and the National Aeronautics and Space Administration.

REFERENCES
