ARL-TR-7583 ● JAN 2016

US Army Research Laboratory

TrafficGen Architecture Document

by Chien Hsieh and Andrew J Toth

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-7583 ● JAN 2016

US Army Research Laboratory

TrafficGen Architecture Document

by Chien Hsieh and Andrew J Toth
Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
January 2016	Final	10/2014–09/2015

4. TITLE AND SUBTITLE
TrafficGen Architecture Document

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Chien Hsieh and Andrew J Toth

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Research Laboratory
ATTN: RDRL-CIN-T
2800 Powder Mill Road
Adelphi, MD 20783-1138

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-7583

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Network science experimentation often requires modeling of realistic network traffic specific to the experiment environment and goals. The multi-generator (MGEN) tool, developed by the US Naval Research Laboratory (NRL), reads scripts to generate real-time traffic patterns to load the network with Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) Internet Protocol (IP) traffic. Each node generating network traffic in an experiment executes a copy of MGEN and requires a custom script to represent its traffic patterns. While this approach works well for smaller experiments, managing the scripts and generating meaningful interaction of network node traffic via the scripts becomes more cumbersome as the number of network nodes increases. This report describes the US Army Research Laboratory (ARL) Network Science Research Laboratory (NSRL) TrafficGen application, which eases the task of composing network traffic scenarios by visually representing multiple MGEN scripts simultaneously in a timeline.

15. SUBJECT TERMS
Network Traffic, MGEN

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Andrew J Toth
Unclassified	Unclassified	Unclassified	UU	22	19b. TELEPHONE NUMBER (Include area code) 301-394-2746

Contents

List of Figures

INTENTIONALLY LEFT BLANK.

1. Overview of TrafficGen Application

Network science experimentation often requires modeling of realistic network traffic specific to the experiment environment and goals. The US Naval Research Laboratory (NRL) Protocol Engineering Advanced Networking (PROTEAN) research group has developed a multi-generator (MGEN) to generate real-time traffic patterns to load the network with Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) Internet Protocol (IP) traffic. Each network node generating traffic in an experiment will have the MGEN application installed and will read the traffic patterns from a script. Scripts contain commands to have the network node listen on specific ports and flows describing the start time, stop time, and specific traffic pattern to send. The variety of patterns MGEN offers enables the researcher to model realistic network traffic patterns. MGEN currently runs on various Unix-based (including MacOS X) and WIN32 platforms.[1]

Modeling traffic scenarios using MGEN requires 1 script for each network node that will participate in the experiment. As scenarios grow, managing individual script files becomes cumbersome and can put the researcher in a position similar to that of a playwright composing a scene by writing each actor's part independently, then combining the parts and hoping they align properly. The more actors in the play, the mode difficult the task.

The US Army Research Laboratory's (ARL) Network Science Research Laboratory (NSRL) developed TrafficGen to ease the task of composing network traffic scenarios. TrafficGen presents network traffic in a timeline format with participating nodes arranged vertically and time presented horizontally. Individual traffic flows are represented by horizontal bars indicating the start time, stop time, and specific traffic pattern that will be sent. Traffic flows can be specified as TCP, UDP, or Sink with traffic patterns of Burst, Periodic, Poisson, Jitter, and Clone. Researchers can experience an added dimension of network traffic visualization by pairing TrafficGen with the NRL Scripted Display Tool 3D (SDT3D)[2] to view network nodes and their communication flows overlaid on a topographical map. SDT3D is built on top of the National Aeronautics and Space Administration's (NASA) WorldWind, which provides application programming interfaces (APIs) for displaying and interacting with geographic data.[3] TrafficGen (Fig. 1) interacts with SDT3D by sending node link commands via TCP or UDP to the IP address and port on which SDT3D is listening, illustrating the source to destination traffic flows at the time indicated by the timeline cursor.

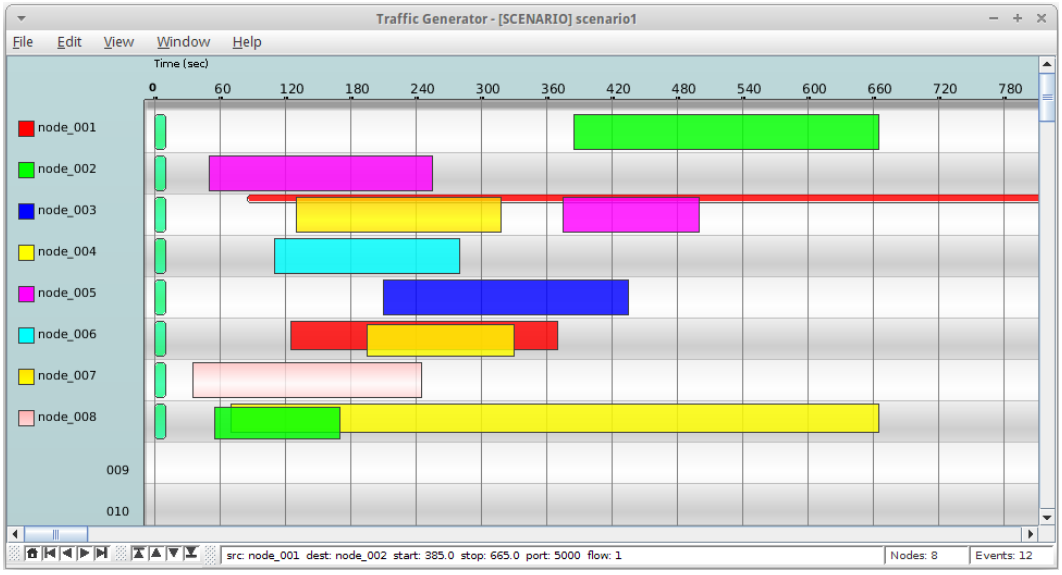

Fig. 1 TrafficGen user interface

The TrafficGen user's guide details specific features and their use. TrafficGen is available for download on the ARL public web site.[4]

2. Modules

Fig. 2 depicts the various modules that constitute this application. It shows its internal interactions and dependencies, as well as the external entities with which the application works.

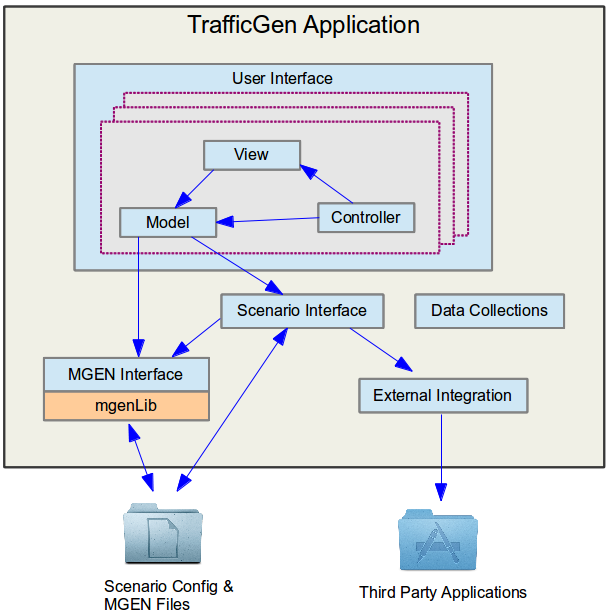

Fig. 2 TrafficGen modules

A key design concept in use here is the model-view-controller (MVC) pattern. In general, the MVC design pattern separates a user interface application into 3 interconnected components: the model, the view, and the controller. The model component represents the data and persistence layer of the application, while the view is the output representation of the data. The controller is the logic that tracks and converts user inputs into changes to the model and the view. By separating these concerns, the responsibilities of the classes are better focused and easier to manage. MVC facilitates reuse by reducing and formalizing coupling between model components and the user interface. The same TrafficGen model is the foundation for the timeline-based user interface and for generating commands for integration with external applications.

3. User Interface

The user interface of this application is comprised of several sets of MVC classes and other support classes. Working together, they present the visual components to the user and manage data updates according to user actions. Each set of MVC classes controls the visual representation of a specific set of information and handles modifications to that information.

To implement the MVC pattern in this application, the model classes extend the Java *Observable* class, and the view classes that generate visual representation of the model class implement the *Observer* interface. The view classes are added to the list of observers of the model class they wish to observe, and will be notified when the model class experiences changes. The controller classes provide event handlers for their corresponding view classes and can manipulate the model class when required.

3.1 Top-Level MVC Classes

The top-level MVC classes control the workspace of the application, specifically the menu and the way the nodes and events are rendered visually.

3.1.1 *TrafficGenView*

This is the principal frame of the graphical user interface (GUI), through which the user can view and interact with data representing the individual traffic flows described in the MGEN files. Its visible elements are the menu bar, scenario view, and the status bar. Menu events are sent to and handled by the *TrafficGenController*.

3.1.2 *TrafficGenModel*

This is a singleton object that provides the data model that powers the GUI. Internally, it maintains a collection of nodes, which map to the opened MGEN files, and furthermore, each of the nodes tracks events that relate them to other nodes.

In addition to the node and event data, there are also methods that provide linkage to the scenario and MGEN interfaces that allow the model class to access the scenario configuration and MGEN files in the file system.

3.1.3 *TrafficGenController*

The controller extends from Java *ActionListener* class, and it responds to events initiated by the user through the menu bar. Examples of those events include New, Open Scenario, Save, Add Node, Add Flow, etc. The event handler in this class, *actionPerformed*, is responsible for invoking the appropriate methods in the model, which, in turn, executes the desired logic.

3.2 Main Workspace Visual Design

Figure 3 shows the visual layout of the *TrafficGenView* and its subcomponents.

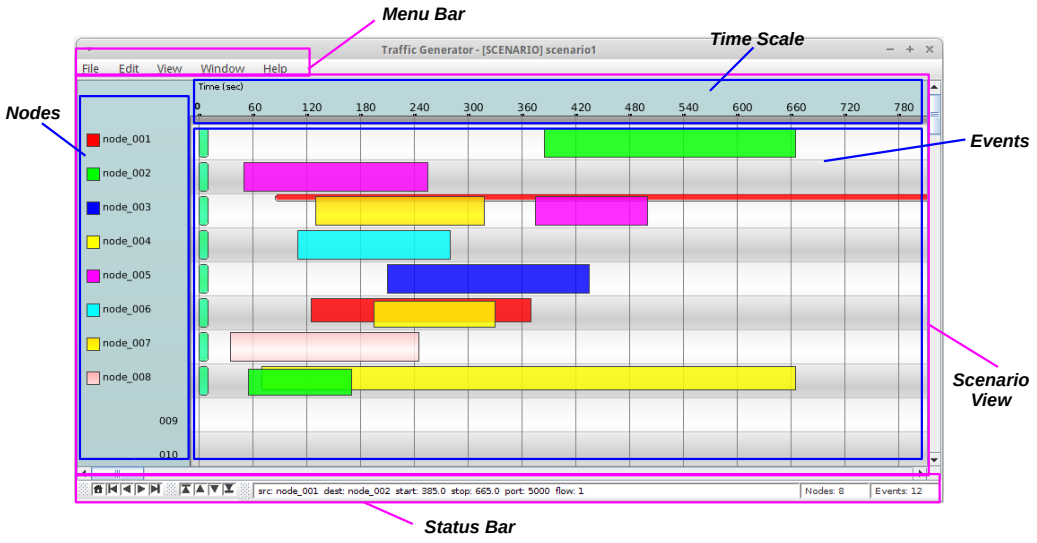

Fig. 3 *TrafficGenView* components

3.3 Initialization

The TrafficGen application is launched by the *Traffic* class, which then initializes the main view (*TrafficGenView*) and its associated model (*TrafficGenModel*) and controller (*TrafficGenController*) classes. Figure 4 describes the startup sequence.

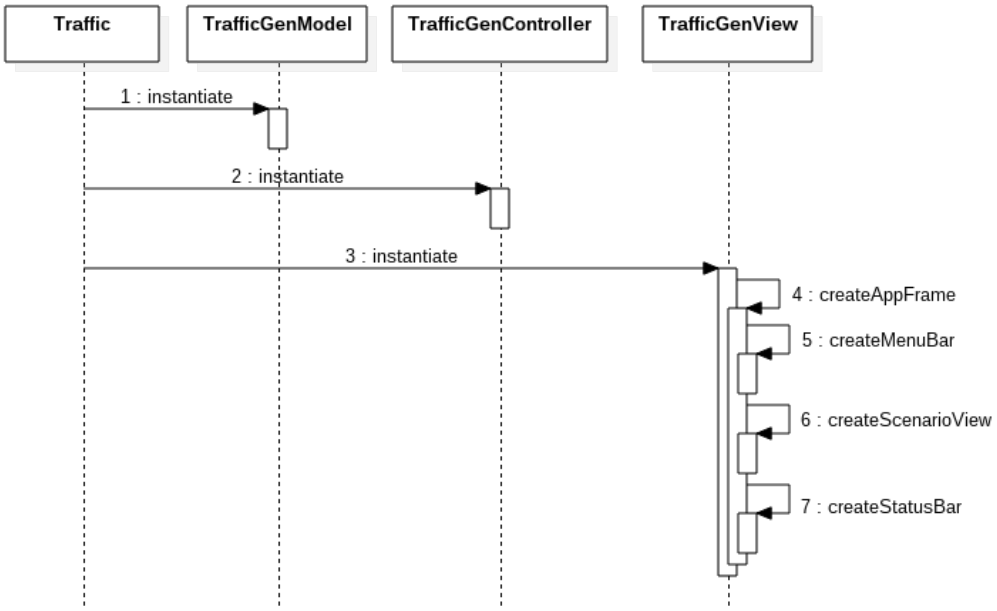

Fig. 4 TrafficGen startup sequence diagram

3.4 Scenario Display and Manipulation

3.4.1 *ScenarioView*

This view class represents the main workspace of the application and is contained within *TrafficGenView*. Its main functions are as follows:

- Set up the layout of the time columns and rows for the nodes.
- Provide the graphical visualization of the MGEN nodes and events found in *TrafficGenModel*.
- Coupled with the *ScenarioMouseListener* class, provide interactivity features that allow users to quickly and easily manipulate the nodes and events.

3.4.2 *ScenarioMouseListener*

ScenarioMouseListener is the controller class that handles mouse events originating from *ScenarioView*. It is responsible for launching editors for node, flow, and reception events when requested. In addition, it processes move and resizing requests of flows and reception events.

3.5 Other Notable Classes

3.5.1 *AppUIContainer*

This is a container class that holds singleton instances of the view classes employed by this application. Because of the way this application is constructed, only 1 instance of each of the view classes is required. The purpose of *AppUIContainer* is to facilitate and standardize the creation and retrieval of those view instances. It is responsible for creating the singleton objects of the views and providing them when requested.

3.5.2 *ScenarioNodeConfigView*

This is the editor class, which allows the user to add or edit an MGEN node to the scenario.

3.5.3 *TransmissionEventConfigView*

This is the editor for the user to add or edit MGEN flows (transmission events). Embedded in this view is another editor that allows user to specify the transmission message pattern.

3.5.4 *ReceptionEventConfigView*

This provides the editor for user to add or edit MGEN reception events.

4. Data Collections

This is a set of composite data types that encapsulate data structures and algorithms that fulfill the need of internal data representation and manipulation of nodes, flows, events, and scenarios. These data types are used extensively throughout this application. The following are the principal classes and brief descriptions:

HostMap	Container class that holds the collection of hosts and their associated MGEN events
HostData	The class that represents a single host and its associated MGEN events
TransmissionEvent	Class that represents a flow (transmission event)
ReceptionEvent	Class that represents a reception event

5. Scenario Interface

The scenario interface is the data persistence layer that is invoked by *TrafficGenModel* to read and write files related to a scenario. Operationally, this interface module is invoked every time the user opens or saves a scenario. While it deals directly with the scenario configuration file, it relies on the MGEN interface module for accessing and manipulating the MGEN files:

ScenarioReader — This class provides the capability to load or import a scenario of MGEN files and synthesize them into data structures that can be consumed by the *TrafficGenModel*.

ScenarioWriter — This is the class responsible for writing configuration and MGEN files related to a scenario

6. MGEN Interface and *mgenLib*

For this application, these 2 components represent the data abstraction and file manipulation layer, through which the application can read, write, organize, and interpret MGEN data and files.

The *mgenLib* library is a framework of classes that reads, parses, and manipulates data from MGEN data files. During file input processing, the library reads a line of MGEN text 1 word at a time. Based on its contents, it will return an object that accounts for every parameter in that line of MGEN text. Conversely, the output logic materializes the content of the data objects into MGEN files.

MGEN interface consists of *MgenReader* and *MgenWriter* classes. They act as interpreters between the *mgenLib* and the other parts of this application. It converts data from *mgenLib* format to something more easily consumable by the model and view classes in the user interface:

MgenReader — This class provides the capability to load or import MGEN files.

MgenWriter — This class is responsible for writing MGEN files during the file save operations

7. External Integration

TrafficGen is designed to interact with third-party applications that can be associated or work with MGEN files.

7.1 SDT3D Integration

The integration consists of transforming the events in the currently open scenario into SDT3D commands, and sending those commands to a running SDT3D application via network connection. TrafficGen interacts with SDT3D by sending node link commands via TCP or UDP to the IP address and port on which SDT3D is listening as configured in the SDT3D View menu option. Nodes represented in SDT3D are either linked or unlinked indicated by the presence of a line between the nodes. The state of links represents the source to destination traffic flows at the time indicated by the timeline cursor on the timescale. A view of TrafficGen traffic flows in SDT3D is shown in Fig. 5. It uses the following classes:

SDTPlayerView User interface that allows user to control the timing of SDT commands and communication options.

SDTInterface This class is responsible for establishing network connections and the actual sending of SDT commands.

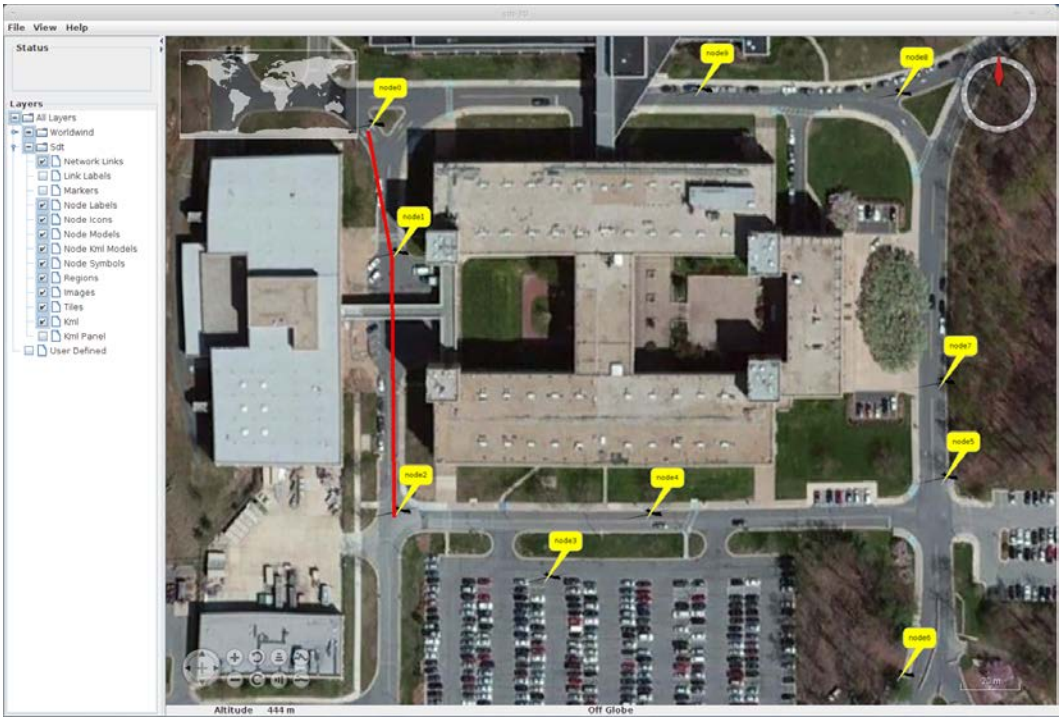

Fig. 5 TrafficGen traffic flows viewed in SDT3D

7.2 Data Exports

TrafficGen provides export features to transform scenario data into other formats that can be ingested by external application at a later time. Currently these functions are handled via the scenario interface.

7.2.1 Export SDT Files

This export feature builds SDT files that contain the same commands that would have gone to SDT3D application in the integration operation as described in the previous section. The exported files can be helpful if the SDT3D application is currently not available or if the user wishes to distribute the commands.

7.2.2 Export MGEN Timeline Script

In this export feature, TrafficGen gathers all MGEN commands destined for the MGEN data files and combines them into a single file with some adjustments in syntax. The exported MGEN timeline script file can then be used as input to the control orchestrator that commands MGEN actors configured within the Common Open Research Emulator (CORE).

8. Class Diagram

Figure 6 is a simplified class diagram of the application, showing the relationships between select classes of this application.

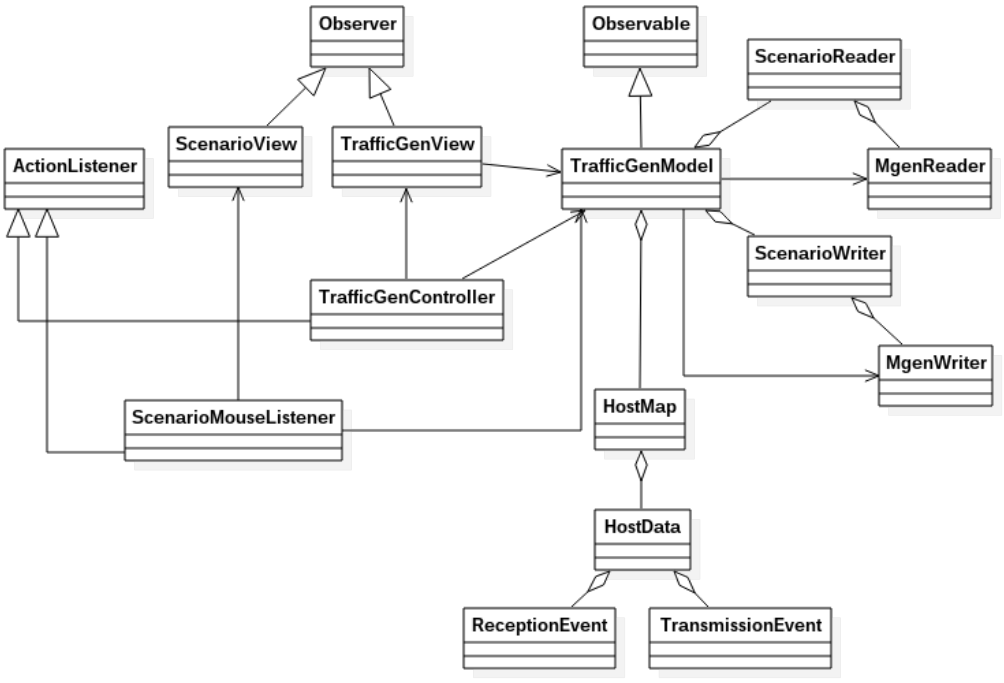

Fig. 6 Simplified class diagram

9. Current and Future Work

TrafficGen continues to evolve as researchers apply the TrafficGen capabilities to a wider variety of scenarios. The following are some of potential features in the future roadmap of the application:

- **Templates for transmission patterns.** The user will be able to define and save transmission message patterns in form of templates. The templates can then be made available for reuse when defining flows. This will greatly improve efficiency and consistency when defining similar types of events.
- **Better support for multicast addresses.** Currently when multicast addresses are used while defining flows and reception events, the user has to enter them manually. We will research and implement a way to better define and organize the multicast addresses so they can be more easily accessible, and thus help improve overall user experience.
- **Expanded third-party application integration.** We will be looking for ways for TrafficGen to work with other applications, in terms of in-process communication, similar to the current SDT3D integration, as well as out-of-process collaboration, via data imports and exports.
- **Orchestration of experimentation events outside of network traffic.** TrafficGen is being considered as the basis for an application that controls experimentation scripts to start and stop processes, network node mobility models, and communications effects between nodes.

10. References

1. US Naval Research Laboratory (NRL) Protocol Engineering Advanced Networking (PROTEAN) Research Group Multi-Generator (MGEN) Tool, http://www.nrl.navy.mil/itd/ncs/products/mgen.
2. Scripted Display Tool 3D [accessed 2015], http://www.nrl.navy.mil/itd/ncs/products/sdt.
3. NASA WorldWind [accessed 2015], http://worldwind.arc.nasa.gov/java/.
4. NSRL area of the ARL Public web site [accessed 2015], http://www.arl.army.mil/nsrl.

List of Symbols, Abbreviations, and Acronyms

API	application programming interface
ARL	US Army Research Laboratory
CORE	Common Open Research Emulator
GUI	graphical user interface
IP	Internet Protocol
MGEN	multi-generator
MVC	model-view-controller
NASA	National Aeronautics and Space Administration
NRL	US Naval Research Laboratory
NSRL	Network Science Research Laboratory
PROTEAN	Protocol Engineering Advanced Networking
SDT3D	Scripted Display Tool 3D (
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Glossary

Continuous Flow	A flow that does not have a stop time defined, which means the streaming of data continues indefinitely.
Flow	Streaming of data (transmission event) from a host to another, as specified by an ON event in an MGEN file. A flow can be terminated by an OFF event.
MGEN file	A script file that contains a sequence of commands and events sent between hosts. Specified parameters include IP, ports, data patterns, and other options.
Reception Event	An event specified in MGEN file that indicates whether a host is actively monitoring network data on a particular port(s).
Scenario	A set of MGEN files and an associated configuration file that in a whole contain a series of related sequences of communication events to describe a particular story.

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL IMAL HRA MAIL & RECORDS MGMT
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
3 (PDF)	US ARMY RESEARCH LAB RDRL CIN T S KREPPS A TOTH B RIVERA