Navy Fire & Emergency Services Loss Modeling Framework

OR 699/SYST 699 Final Report

November 29, 2012

Sponsored by:

Prepared by:
Adam Bever
Megan Malone
Saba Neyshabouri

George Mason University - Fairfax, VA

Table of Contents

List of Figures

Executive Summary

The Navy maintains over seventy bases across the world, which rely on their internal Fire and Emergency Services branch to provide fire prevention and mitigation services. With defense budgets under intense scrutiny, the Navy F&ES department is required to justify their costs and to identify areas where reductions could be made. The general consensus is that reducing F&ES force size will result in increased losses from fires, injuries, and other onsite incidents, but quantifying that expected loss has been a significant challenge.

Following two previous GMU student groups, which have tackled the problem from different angles, the aim of this project is to combine the efforts of both groups and provide a foundation for future development in an incremental and modular way. This report describes both the development process as well as the produced framework. The framework in its current state allows for a simplified set of data compared to previous groups, specifically by allowing the user to enter clusters of homogenous buildings and requiring the response time between stations and buildings or building groups instead of map or grid coordinates for each element in the model. The team identified Submarine Base New London as a candidate for proof of concept analysis and generated two force reduction scenarios for comparison with the baseline installation.

The analysis tool is still in its infancy, but can already be used to generate results for building and vehicle fires and compare expected loss across different scenarios. The proof-of-concept analysis provided intuitive results and demonstrated that the team accomplished the majority of the goals laid out for the project during the short time frame available. During development of the framework, the team also identified several avenues for future development that would increase the fidelity of the model, aid the user in data entry, or provide additional output information that could be used for force size justification.

Introduction

Background

Navy Fire & Emergency Services (F&ES) protects 70+ installations worldwide via four functions: Fire Protection, Fire Prevention, EMS Transport, and Aircraft Rescue & Fire Fighting. However, in a fiscally constrained era, the Navy is required to more carefully budget and trim their resources and the F&ES are not exempt. In order to make decisions regarding a reduction in assets and services, the Navy needs to be able to quantify the risk of loss of infrastructure, property, and lives.

The Navy has hired Mr. Fred Woodaman, Principal Analyst at Innovative Decisions Inc, to inform their decision process regarding reducing F&ES assets. He has created a cost model called Fire and Emergency Service Program and Objectives Memorandum (FESPOM). What he needs is a model that can calculate the expected losses given an installation and a set of F&ES assets. With such a model, he hopes to be able to compare the cost and consequences of reducing F&ES capacity in one installation versus another.

In the Fall 2011 semester, a team of students from George Mason University (GMU) developed an Excel-based simulation of a generic installation with a simplistic loss function driven by historical call data. This model uses loss as a measure of the ability of F&ES assets to reach the location of an emergency. To expand on this model, a second team of GMU students developed a probabilistic loss model of the residential fire scenario during the Spring 2012 semester. This model provides a more realistic simulation of a two story single-family dwelling fire.

Problem Statement

These two previous models provide a basis for quantifying loss on an installation given a reduction in F&ES assets. However, they are currently disconnected and simplistic. The Navy needs a unified tool that can realistically quantify loss.

Objectives

The Fall 2012 GMU team was committed to the following objectives:

- To construct a model of a generalized installation that can be made specific given simple data for a particular installation.
- To build an efficient simulation model that will calculate expected losses using probabilistic loss models of various emergencies.
- To include and expand on the residential fire probabilistic loss model.
- To provide an interface to allow for simple addition of new probabilistic loss models for other emergency scenarios.

In summary, the study team intended to create a simulation model of an installation that was based on simple facts, rather than a building grid, and is capable of calculating partial loss, rather than assuming loss to be binary. This involved essentially combining the two previous models and making the original simulation more closely reflect reality. While the Fall 2012 GMU team was not able to complete a probabilistic loss model for every emergency, they did provide a basis for easy integration of such models in the future.

Scope

This study sought to build a generalized model of an installation. The team confined the model to a simulation of incident mitigation based on assumptions about the composition of the installation. There was no attempt to optimize fire station locations, specify building materials or weather, or model incident prevention.

Limitations

Since the study team did not contain any subject matter experts on F&ES, it was not able to build probabilistic loss models for all emergency scenarios that can occur on an installation within the time constraints of a semester. Therefore, the simulation model includes an interface to connect with future scenario models. The existing model utilizes only the adapted residential fire loss model, which provides infrastructure and property losses and does not attempt to model loss of life as a result of fires. Additionally, the sponsor asked the team to disregard airfields for now,

as the regulations for F&ES force sizes that services airfields are much more strict than those governing other structures.

Methodology

The study team began by researching the structure of Naval installations and the responsibilities of F&ES. This research provided information for building a simulation model of a generalized installation and various emergency situations. The team also planned to explore ways to expand the Spring 2012 GMU team's residential fire probabilistic loss model to include buildings of various types. The resulting hybrid model, written in Visual Basic for Applications in Excel, provides a measure of expected losses for a given installation and its F&ES assets.

Once a simplified base layout was identified, such as the example shown in Figure 1, the simulation applies loss modeling to each building based on the likelihood of a fire event occurring, driven by past service calls for that base, as derived from the Program Compliance Assessment records, such as those shown in Figure 2. Initial loss modeling utilizes a scalable version of the Spring 2012 team's residential model (shown in Figure 3). When and if new models become available, they can be substituted as appropriate.

Name	NSWC Carderock				
Location	Bethesda, MD				
Population	1000				
Total Area	0.356	sq. miles			
Buildings	20				
F&ES Stations	1				
Mutual Aid	3				
Index	**BuildingID**	**BuildingName**	**Type**	**Floors**	**Floor Area (sq ft)**
0	23	Security	Commercial	1	900
1	57	Visitor Office	Commercial	1	900
2	43	Mess Hall	Retail	1	2500
3	113	Tow Tank A	Warehouse	1	100000
4	117	Tow Tank B	Warehouse	1	90000
5	119	Tow Tank C	Warehouse	1	67000
6	32	Code 32	Commercial	3	3000
7	76	Lab B	Commercial	4	3500
8	78	Lab C	Commercial	2	2500
9	79	Lab A	Commercial	2	1700
10	95	Code 95	Manufacturing	1	10000
11	13	Records Room	Commercial	1	2700
12	52	Building 52	Commercial	4	6500
13	9	Steel Shop	Manufacturing	1	12000
14	102	Wood Shop	Manufacturing	1	6000
15	27	Storage A	Warehouse	1	3200
16	28	Storage B	Warehouse	3	4900
17	17	Gym	Commercial	1	6500
18	59	Medical	Commercial	2	4000
19	40	IT Routing	Commercial	1	2000
20	41	Facilities Dept	Commercial	2	2800
Index	**Station**	**Engines**	**Crew/Engine**	**Response Time**	
0	Station 1	1	4	4	
1	Mutual 1	1	4	9	
2	Mutual 2	1	4	11	
3	Mutual 3	1	4	11	

Figure 1 Sample Navy Installation Transformation

Summary By Incident Type

Report Period: 1/1/04 - 12/31/06

Calls By Incident Type	Frequency	Percent Of Total Calls	Mutual Aid None	Mutual Aid Given	Mutual Aid Received	Other Aid Given	Invalid Aid Flag	Exposures	Total Incidents
FIRES									
Building Fires (110-118, 120-123)	25	0.30%	16	0	9	0	0	1	26
Vehicle Fires (130-138)	8	0.10%	5	0	3	0	0	0	8
Others Fires (100, 140-173)	27	0.33%	25	0	2	0	0	0	27
Total Fires	60	0.73%	46	0	14	0	0	1	61
Overpressure Ruptures, Explosion, Overheat (200-251)	2	0.02%	2	0	0	0	0	0	2
RESCUE CALLS									
Emergency Medical Treatment (300-323)	319	3.88%	142	0	177	0	0	0	319
All Others (331-381)	37	0.45%	35	0	2	0	0	0	37
Total Rescue Calls	356	4.33%	177	0	179	0	0	0	356
Hazardous Condition Calls (400-482)	7037	85.51%	7028	0	9	0	0	0	7037
Service Calls (500-571)	43	0.52%	43	0	0	0	0	0	43
Good Intent Calls (600-671)	58	0.70%	57	0	1	0	0	0	58
Severe Weather or Natural Disaster Calls (800-815)	3	0.04%	3	0	0	0	0	0	3
Special Incident Calls (900-911)	6	0.07%	6	0	0	0	0	0	6
Unknown Incident Type (UUU)	0	0.00%	0	0	0	0	0	0	0
FALSE CALLS									
Malicious Calls (710-715, 751)	10	0.12%	10	0	0	0	0	0	10
Other False Calls (700, 721-746)	654	7.95%	652	0	2	0	0	0	654
Total False Calls	664	8.07%	662	0	2	0	0	0	664
TOTAL CALLS	8229	100.00%	8024	0	205	0	0	1	8230

Total Incidents With Exposure Fires	1	Total Fire Dollar Loss	$1,119,584
Total Exposure Fires	1	Total Dollar Loss	$1,620,048

Casualty Summary	Civilian	Fire Service
Fire Related Injuries	3	0
Non-Fire Injuries	3	0
Fire Related Deaths	1	0
Non-Fire Deaths	1	0

Page 2 of 2 NFIRS 5.0 National Reporting System 4/4/07 8:11:23 PM

Figure 2 NAS Key West Incident Reports from March 2007 PCA

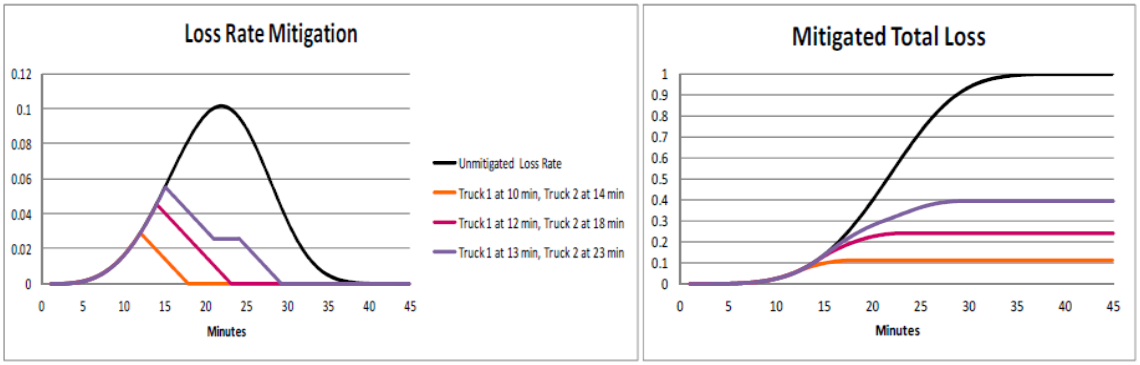


Figure 3 Residential Fire Loss Model (Spring 2012)

Previous Work

Two other teams worked on the different aspects of the F&ES project. This section briefly explains what was done in these past projects.

Fall 2011

In Fall 2011, the project team worked on the following problem statement:
"Develop a mathematical model of the expected loss at an installation given an application of F&ES resources".
This model was built as a supplement to an existing cost model already developed by the stakeholder. In order to develop their model, the project team established the following general requirements for their model and product to address multiple aspects of the system:

- Functional and data requirements
- Look and feel requirement
- Usability
- Ease of learning
- Precision
- Reliability and availability
- Maintainability and portability
- Legal requirements
- Solutions

In order to gather pertinent information about the problem in hand, the team used multiple data sources:

- Navy F&ES 2008
- Fairfax County Fire and Rescue Department 1997-2010
- Interviews
    - Steve Burke, Volunteer fire fighter
    - Cpt. Tom Arnold, Fairfax County Fire and Rescue

The model developed by the Fall 2011 team requires the following data to run:

- X-Y values for the coordinates of each location

- Resources at each existing location (station)
- Frequency of events
- The cost associated with loss and lives

In order to model this problem, a simulation approach at the installation level is used. The focus is on simulating the response of individual vehicles to single event at the installation level. This model considers simultaneous events and it takes into account the locations and the model of vehicles and their capabilities. Microsoft Excel is the platform for the model due to its availability and ease of use to the user.

The team ran a hypothetical case for an installation based on George Mason University Fairfax campus to test the model. In their tests, they considered three different scenarios:

- Three on-site stations with various vehicles + Local community station
- Three on-site stations with various vehicles, but no Local community station
- Two on-site stations with various vehicles + Local community station

In their model, the Fall 2011 team considered the following assumptions:

- The data is to be feasible to collect.
- Simulation includes 100 locations on campus based on the density.
- The number of vehicles available at each station is based on Program Compliance Assessment (PCA).
- Locations of stations are chosen on GMU (No explanation of how)
- Vehicles are unavailable 5% of the time due to maintenance (Source : Interviews)
- High loss low probability events can happen even though there is no history of occurrence.
- Some calls may be false alarms.
- Distances are assumed to be straight line distance and deterministic in time
- Vehicles are fully staffed with properly trained personnel. (No cross manning)
- Events are happen on an exponential distribution and independent of each other and of the time of day/year.
- Not having all the necessary vehicles at the beginning of the event and for the full duration of the event will cause full loss.

- False alarms require the same amount of vehicles but for only 30 minutes.
- Loss is calculated as a step function.

There is no implication of the running time of the model as well as the number of replications in their simulation.

Spring 2012

The second team working on the F&ES project had the following problem statement:
"To model the loss incurred due to an emergency scenario for given F&ES resource condition".
They mention that the scope of this problem is limited to consider the following:

- Sample military installation
- Single family residence fires only
- Measuring generic loss (ratio) without regard to specifying property or dollars

It is important to note that the first bullet point was not addressed in their report.
In the process of data gathering, the Spring 2012 team performed interviews with experienced firefighting personnel. In an interview with Dan Hunt, Federal Fire Fighter, they extracted the fire response procedure as shown in the following chart.

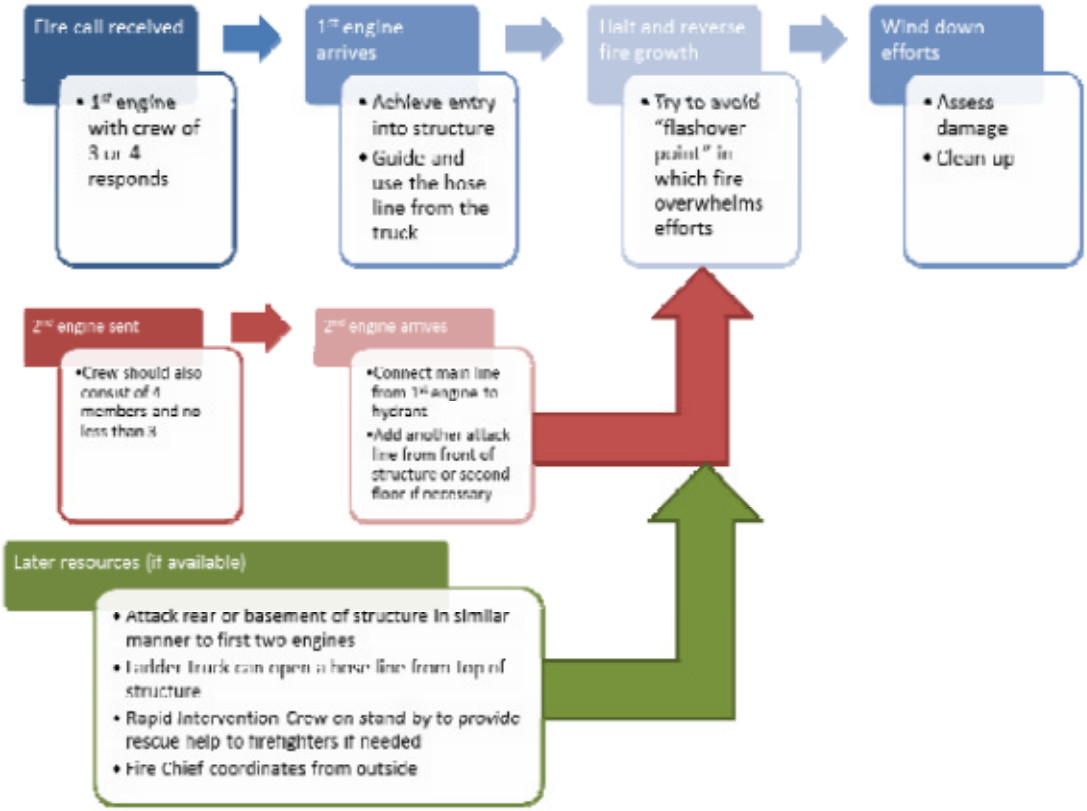


Figure 4 F&ES Response Flow (Spring 2012)

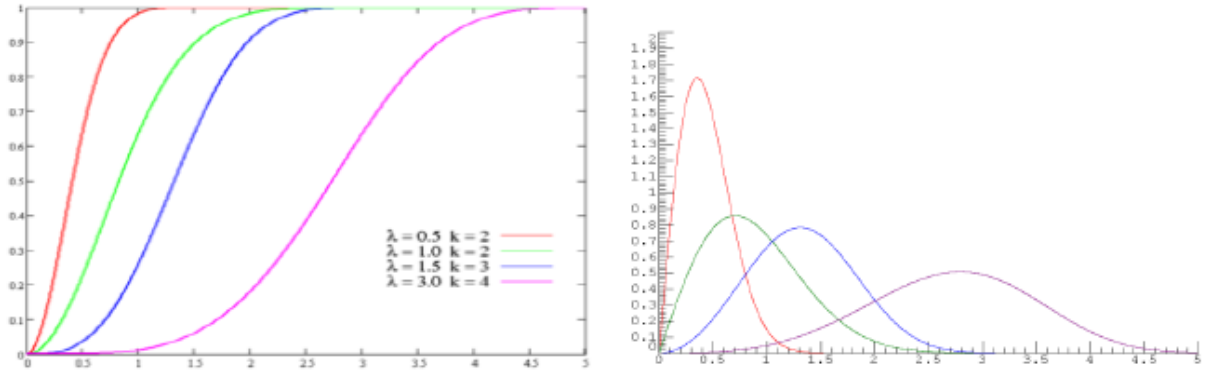
In another interview with Patrick Cantwell, a Systems Engineering PhD. Student at GWU and a volunteer fire fighter, they learned the following:

- The goal of Stafford County, VA department is to respond on scene within eight minutes. (Insurance driven, rather than safety driven)
- Having a crew of three persons instead of four will decrease the effectiveness of the team
- There is a difference between the behavior and rate of loss for downstairs fire versus and upstairs fire.
- Different structures allow fire to spread at different rates.

The team also did a literature review on models for fire spread and they came upon the concept of the "Flashover Point" (FP). This is the point at which a fire breaks relative containment and engulfs the structure to the point that complete damage mitigation of an asset becomes nearly impossible. FP usually occurs around ten minutes after the fire's ignition. FP is a function of the energy release rate which in turn will change the temperature which can set other materials on

fire. They show based on historical data that the effect of timely response in fire mitigation is considerable.

The Spring 2012 team aimed to model the loss caused by fire in cases of various response time. The important assumption in their modeling is that the loss is not a step function. In fact a timely response from the fire department can mitigate the loss greatly. Based on the general shape of the loss function, they decided to use a Weibull Cumulative Distribution Function (CDF) as the general shape for their loss function (function of time) so that the loss rate is modeled as a Weibull Probability Density Function (PDF). Since different fires have different power and burn rates, parameters are chosen from random Gamma distributions based on the different residential fire types. For more information please refer to the Spring 2012 team's report.



General form of the Weibull PDF: $f(x; a, b) = \frac{a}{b^a} x^{a-1} * \exp\left(-\left(\frac{x}{b}\right)^a\right)$

Figure 5 Weibull Distributions

Most of the data and assumptions that are used in the model were based on their Subject Matter Experts (SME). The team tried to characterize the difference between fires that originate in different parts (rooms) of the building by their different probabilities in engulfing the entire building.

It has also been mentioned that not all fires burn at the same rate and not all fires burn down an entire building (some fires are self contained). To address this, the team introduced parameters to create variability in types of fires generated by the simulation.

The Spring 2012 team used the following as the primary assumptions in the model:

- Response to the fire will mitigate the loss rate with a linear form.
- The mitigation of first truck will go on as long as it has water supply from the tank. (Six minutes)

- The second engine to arrive will hook the main line to the hydrant.
- The response time of the second engine has a significant effect on mitigation.
- If the first truck's supply of water is depleted before the second truck can connect to the hydrant, the loss rate stays constant.

The team used a Monte Carlo simulation to analyze the effects of the average response time of the two fire engines and the percentage of crews that are fully manned. They assume the response times for engines are normally distributed with standard deviations of two and four minutes, for the first and second trucks, respectively. It is important to note that the mean response times are inputs to the model.

Fall 2012 Framework Description

This section describes the different components of the analysis framework, such as the installation template, fire loss model as implemented in VBA code, the incident generator for spawning fires and other events over a given time span, the incident response model, and the reporting features. The general flow of the model is shown in Figure 6. The user enters information about a specific installation, which could be cut and pasted from previously used analyses. This includes historical event data for fires, vehicle collisions, severe weather, and other incidents, installation profile of buildings or building groups and onsite F&ES stations as well as offsite F&ES stations for which there are mutual aid agreements in place, and finally the vehicles that are housed in the stations. Once all this information is entered, hitting the Run Simulation button will cause the underlying VBA code to ingest all this data to initialize the model, and then run the specified number of iterations for a one year time frame at one minute time resolution, randomly generating incidents and responses. Once all the iterations have been completed, raw data is dumped to a tab-separated text file, and statistics are computed for display to the user in graphical form.

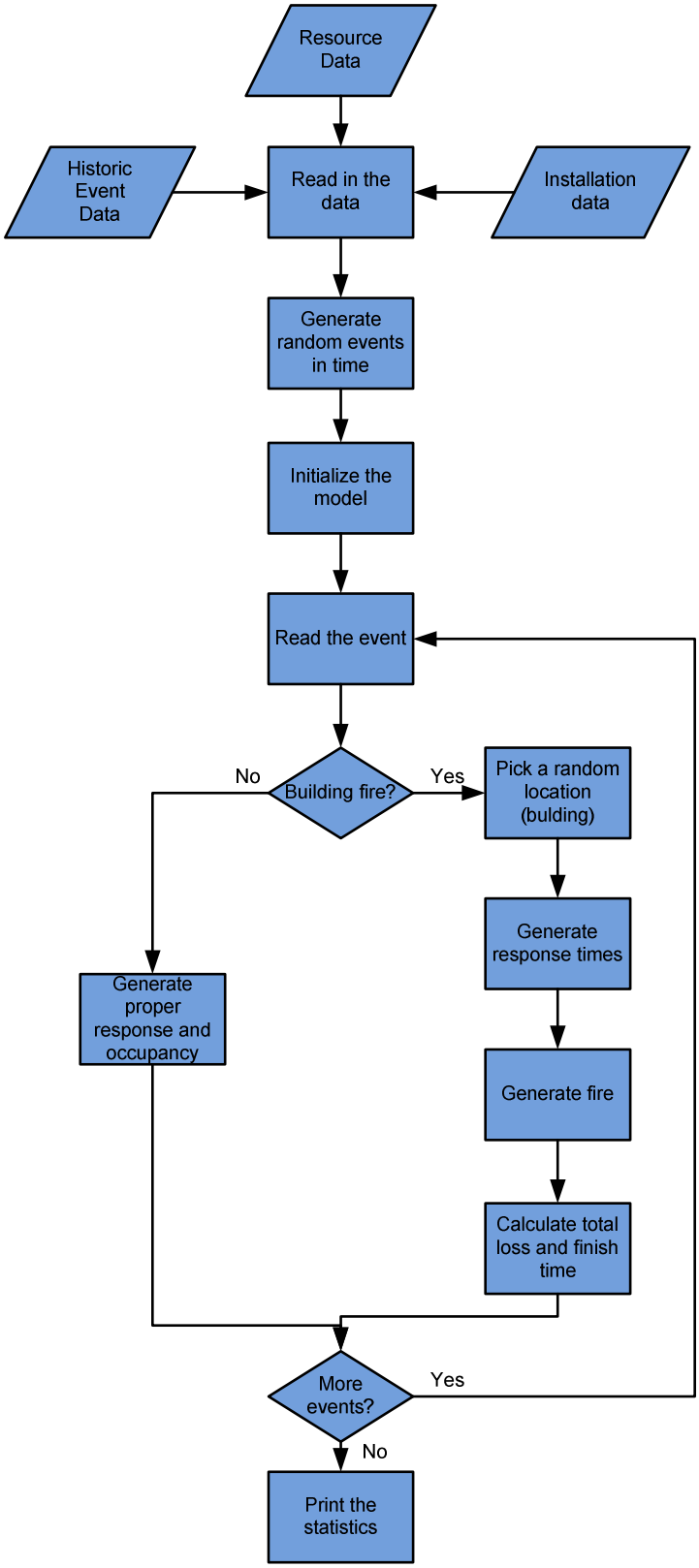


Figure 6 Analysis Framework Flow Chart

Assumptions

The following assumptions were made in the development of the analysis framework:

- Complex and varied Navy installations can be generalized and simplified, such that they can be described adequately by a relatively small set of parameters.
- F&ES forces are relatively small and integral, such that forces cannot necessarily be reduced by a given percentage to match a desired budgetary outcome. (i.e. One cannot reduce a fire truck by 90%.)
- Response time is defined as the start of the incident until a response vehicle arrives at the location.
- Response time to a location within a cluster of buildings from a given fire station will be uniformly distributed within two minutes shorter or longer than the nominal response time to the cluster from the same fire station.
- Incidents are addressed on a first-in, first-out basis. There is no priority given to one type of event over another when events overlap or occur simultaneously.
- F&ES events will occur with the same frequency over the next year as they have on average over the period specified in an installation's PCA report.
- No building is any more likely to catch fire than any other building.
- Any building that catches fire begins in a state of good repair.
- F&ES vehicles must return to their assigned station before responding to another event.
- All vehicles may become unavailable for maintenance for a set period of time with a certain probability. This probability and length of unavailability may be set by the user.
- Vehicles are assumed to be fully manned when needed for an incident response.
- Vehicles at mutual aid stations are always available for use on the installation. However, these vehicles will only be selected for response if there are no onsite vehicles available.
- Loss from fire follows a Weibull distribution using a random draw for parameters of distribution and a given building size.
    - The adoption of fire loss model is based on the previous work done by the Spring 2012 team. Their main reason for choosing the Weibull distribution is that the shape of the CDF is very close to the total loss function that exists in the literature. For more details please refer to the Spring 2012 team's final report.

- If the first fire company to respond is a tanker truck, it only uses water on the truck, allowing a limited amount of fire-fighting time. The second company to respond, regardless of truck type, hooks up to hydrant to assist, and thus has an unlimited supply of water. The third company to respond also connects to a hydrant. Since the NFPA guidelines specify three companies as an adequate response, the framework will assign three companies to each fire. Additionally, vehicles will respond as soon as they are able to do so, meaning three response vehicles could arrive nearly at the same time or one or more could arrive late.
- The existing residential fire model is limited to a fixed number of floors (two) and only four rooms per floor with the ignition profile randomly selected.

Installation Template

The installation template is made up of several components that are separated by object types as listed below.

Header

The header block contains the highest level of information about the installation, such as a base descriptor or name and its location. These are only used in the simulation for identification on the output reports. The population field describes the typical number of people onsite, including Navy personnel, civilian workers, and family members. This information is currently unused in the simulation, but could be used as an input into an injury/casualty model as a result of fires, vehicle collisions, or other onsite incidents. Likewise, the area field describes the overall footprint of the installation and is currently unused, but could be used for building and/or population density calculations or for randomly generating incidents in locations other than buildings. The building field instructs the model how many building entries to expect in the building description section. This can include individual buildings or groups of buildings, and should also include F&ES buildings. The next two fields for onsite F&ES stations and mutual aid stations jointly describe how many station entries the module should expect to find in the station listing section. The number of vehicles entered in the header similarly tells the model how many vehicle entries to expect the in vehicle description section. The vehicle counts are broken down in more detail in the station list and vehicle list sections. Finally, the maintenance

period entry describes the percentage of time that a vehicle is typically unavailable due to maintenance. This value is used to determine the availability of vehicles when assigning responders to generated incidents. In conjunction with this, the Maintenance Time field specifies how many minutes a piece of equipment will be unavailable when maintenance is required. The last two elements in the header section are not related specifically to the installation itself, but have to do with the simulation model. The Replications field determines how many one-year iterations the simulation should run before reporting its results. The Random Seed field provides a starting point for random number generation. This can be set to any positive integer, allowing the user to generate runs for the same installation using the same set of randomly generated numbers. The Random Seed can also be set to zero to enable Excel to utilize the current time as its random seed.

Building List

Each entry in the building section can be either a single unique building or a group of roughly homogenous buildings. This creates some flexibility in the installation model by allowing the user to enter information for similar buildings only once and have the simulation model produce the duplication. The user may enter as many or as few buildings as they wish, but the better the model represents the actual base, the more realistic the results will be. F&ES buildings should also be included in this list even though they will also be listed separately under the station section. Incidents requiring F&ES response can happen even at F&ES locations!

The Index field is automatically populated based on the number of buildings specified in the header section. It is used as the primary reference designator by the simulation model.

The BuildingID field and the BuildingName field are descriptive identifiers used in the simulation output for more user-friendly building identification. The reason for two separate fields is that most Navy bases assign building numbers to all structures on a base, which will be unique, even when some buildings share a name. So having both the Navy's reference designator ID as well as a name makes the building entry readable by both military personnel as well as civilian contractors.

The Type field specifies what type of building this entry is, and must be from a predetermined list of building types: Residential, Commercial, Manufacturing, Warehouse, Laboratory, Retail, Dock, or Airfield. These type designators inform the simulation model of which type of loss modeling to perform. When new models are developed, they can be tied to these types to allow

the simulation model to handle each type differently. The user should utilize their best judgment when assigning building types as some buildings may comprise multi-use spaces. Some examples of building type assignment are shown in Figure 7.

BuildingType	Defining Characteristics	Examples
Residential	Areas which are generally vacant except for private meals and sleep.	Barracks, apartments, condos, single family homes, hotels
Commercial	Structures with high usage during daytime hours, typically those with office, conference rooms, or other non-retail gathering spaces.	Office buildings, classroom facilities
Manufacturing	Facilities for assembly, alteration, or transformation of raw materials into finished goods, typically containing tooling equipment.	Metal shop, wood shop, welding shop, smelters, auto body shop
Warehouse	Storage facilities for non-hazardous materials such as files, books, and raw materials.	Storage facilities for inert substances
Laboratory	Structures which contain high-value test equipment	Storage facilities for hazardous substances, medical offices, hospitals, equipment testing facilities
Retail	Facilities which primarily are used for the sale of goods, including food service.	Clothing store, Navy Exchange, galley, bars
Dock	Any facilities used primarily for service of water craft.	Piers, dry docks
Airfield	Any facilities of an onsite airport.	Runways, hangars, control towers, passenger gate areas

Figure 7 Examples of Each Modeled Building Type

The Floors field specifies how many floors the building has, and the Floor Area field represents the typical area of a single floor of the building, in square feet. For cases where buildings contain floors of different sizes, the user should enter either the area of the ground floor or the weighted average floor size.

Finally, the Units field represents the multiplicity of the building entry. A value of 1 indicates that the building is a single structure. A value greater than 1 indicates that the building entry

describes multiple buildings of similar size and type that are roughly co-located such that F&ES response time to any building of the group is approximately the same. This last caveat is necessary as the entire building group will be assigned a single nominal response time for each F&ES station in the installation model, though actual response times to each building in the group will be randomly assigned during the incident response portion of the simulation.

Station List

The stations section contains entries for both onsite F&ES stations and mutual aid stations. Onsite stations should be listed first, followed by stations for which a base has mutual aid agreements in place.

The Index field is automatically populated based on the number of stations specified in the header section. It is used as the primary reference designator by the simulation model.

The Station field provides a place to enter the station name or other reference designator for usage in the output data.

The Vehicles field describes how many vehicles are assigned to this station. This value must match the number of vehicles in the Vehicles description section that are actually assigned to this station.

The Crew field is used to determine how many F&ES crew are assigned to this station. At the current time, all crew are assumed to possess the necessary skills to operate any piece of F&ES equipment and are available to be assigned to any vehicle being dispatched. This value should account for all crew members assigned to the station, not just the staff currently on-duty. Future expansions to the model could add individual skill requirements or delineate off-duty versus on-duty crew, such that off-duty crew might be available for backup but require additional time to deploy.

The Type field can be set to either Mil or Civ to denote whether this station entry is an onsite (Mil) F&ES station or a mutual aid (Civ) F&ES station.

Vehicle List

The vehicles section contains entries for both onsite F&ES stations and mutual aid stations. Entries can be listed in any order, though for clarity, the user may wish to group vehicles assigned the same station together.

The Index field is automatically populated based on the number of stations specified in the header section. It is used as the primary reference designator by the simulation model.

The Vehicle field provides a place to enter the vehicle name or other reference designator for usage in the output data.

The Type field specifies what type of vehicle this entry is, and must be from a predetermined list of building types: Pumper, Tanker, EMS, HazMat, Command, or Auxiliary. These type designators inform the simulation model of which type of functions a vehicle can provide, whether it is a firefighting apparatus or a medical transport, etc. If there is any doubt about the type of firefighting vehicle at a station, enter it as a Pumper type, which allows it to respond to nearly all emergency dispatch situations, but does not grant it the instant fire response capabilities of a Tanker type, which carries water onboard. Command and Auxiliary types currently serve no F&ES function, but could be used in assigning vehicles to incidents that do not necessarily require immediate response, such as Service Calls or Good Intent applications.

The Location field informs the simulation model which F&ES station the vehicle entry is assigned to and must be set to the index of the corresponding station from the Stations section.. Additionally, the number of vehicles assigned to each station in this section must match the expected number of vehicles specified in the Vehicles field of the Stations section.

The Crew On Duty field defines the number of F&ES crew required to operate this vehicle. When an incident occurs, if there is a vehicle available, the current model assumes that there will always be enough crew on-duty to staff it. A future update could utilize the number of crew per vehicle and the crew on duty at the station to determine if a vehicle could actually be deployed. As mentioned in the Stations section, the current model assumes that all crew have the necessary skills to perform any F&ES duty. A future expansion to include crew skills would have to be reflected here as well in terms of detailing not only what skills are required, but how many crew members with each skill are required for a given vehicle.

Response Time Matrix

The Response Time section is an NxM matrix, where N is the number of building entries and M is the number of combined onsite and mutual aid stations, both specified in the Header section. The Building Index and Station Index are automatically populated from these values, and the Building Name and Station Name fields are then pulled from their corresponding sections as a reference. The user must then enter the time, in minutes, for a typical F&ES response between each building and every F&ES station. If F&ES stations have been included in the building list, take care to correctly enter the response time between a station and itself. The response times

entered here should be the typical response times, usually found in the Program Compliance Assessment (PCA) report for the installation. These values serve as nominal response values in the simulation model. For building clusters, the response time is subjected to random fluctuations when an incident is generated that affects an individual building in the cluster to allow for variance in the arrival time of F&ES equipment, reflecting marginal location differences between buildings in the cluster.

Call Data

The Call Data section is mostly comprised of actual incident data from an installation's PCA report. This represents real-world data accumulated over a fairly long time span and is used to generate the baseline incident probabilities in the simulation model. The Navy F&ES Handbook can provide more details on what constitutes each specific type of incident and what type of response is necessary. From a user perspective, they need only enter the required data for the time period, incident counts, and injury counts. Currently, only the incident counts are used by the simulation model and incidents are assumed to be uniformly distributed throughout the specified time period. The number of days in the period is computed automatically from the start and end dates. The total number of calls is also computed automatically as a sum of all entered incidents. Neither of these values needs to be entered by the user.

Fire Loss Model

The main incident that is modeled in the simulation is a residential building fire. The fire generation module allows for the simulation to generate a fire incident as well as generating the loss statistics based on the response to the fire. The module is based on the previous research, done by the Spring 2012 team. In their model, ten different fire types were identified, based on the fire origin and final spread, shown in Figure 8.

Fire Origination	Final Spread Limited to:
Ground Level	Original Room
Ground Level	2 rooms on same floor
Ground Level	3 rooms on same floor
Ground Level	1 Room in Other Floor
Ground Level	Whole House
Upper Level	Original Room
Upper Level	2 rooms on same floor
Upper Level	3 rooms on same floor
Upper Level	1 Room in Other Floor
Upper Level	Whole House

Figure 8 Fire Ignition and Spread Scenarios

Based on the probabilities of fire spread from previous studies they have calculated the different probabilities of each specific incident happening. Having different probabilities for each incident will allow the model to generate a random fire based on the random draw from a Uniform (0,1) distribution.

Based on the assumptions in the previous project, not all the fires burn the same way. In order to introduce variability in the fire generated by the module, the parameters of the Weibull distribution are chosen from a random draw, from the Gamma distribution. The parameters of the Gamma distribution for each type of fire are assumed to be known from the subject matter expert.

In order to generate a response procedure to a fire, following assumptions are made in addition to those made by the Spring 2012 team:

- All the trucks responding to a fire are pumpers which are hooked up to the fire hydrants
- Fire trucks will start the mitigation process and after the start of the mitigation, the fire loss rate (which has the shape of the Weibull PDF) will decrease with a linear rate in time.
- For the mitigation rate of one fire truck is assumed to be 0.004. This rate is chosen based on the previous model assumptions.
- Unlike the previous model, extra fire trucks will expedite the mitigation process.

- Each extra fire truck will increase the mitigation rate by 0.004.
- Three fire trucks (pumpers) will respond to a building fire incident.

The following is a summary of the response procedure:

1. The fire generator module will be supplied with up to three ascending response times corresponding to the response times associated with the responding pumpers.
2. The fire generator module will randomly choose a fire scenario from the list of different fire types.
3. Based on the specified fire, parameters for that fire are randomly generated.
4. The total loss and loss rate up to the time that first truck responds is calculated.
5. The time that the fire could be put out with the first truck is calculated.
    a. If the finish time with truck one is smaller than the arrival time of the second truck then the total loss and finish time are reported.
    b. If the finish time is greater than the response time of the second truck but less than the response time of the third truck, then the new mitigation rate, finish time with the second truck and total loss are calculated.
    c. The same calculations take place for the last fire truck.

Here is an example of the results generated by the fire generation module:

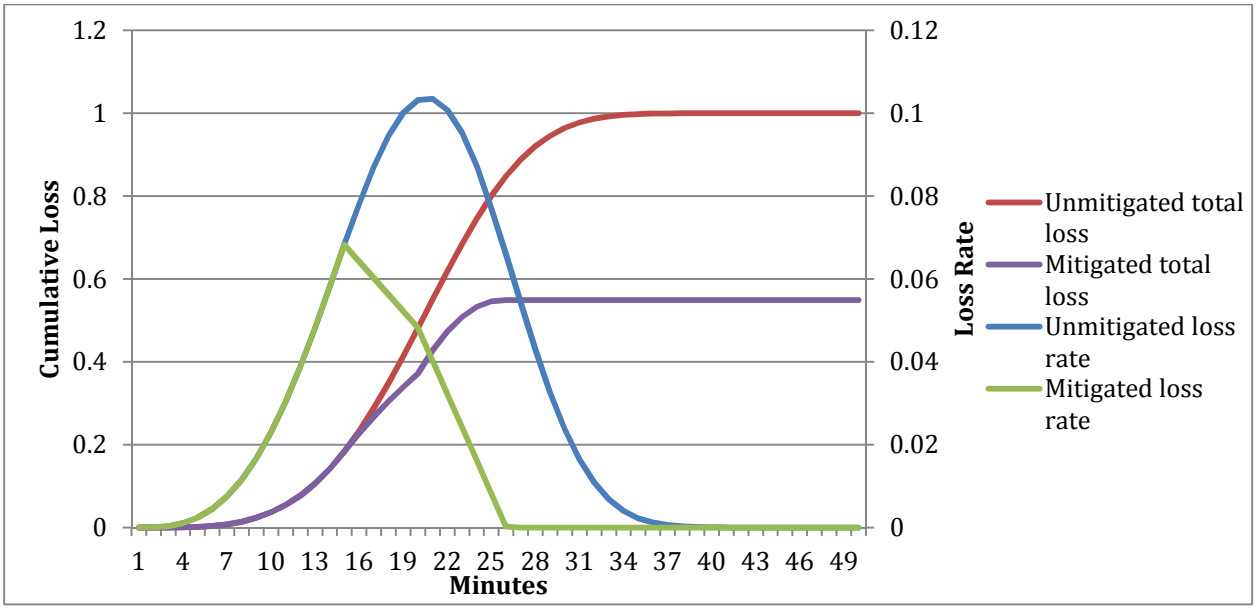


Figure 9 Fire Mitigation Over Time

The left Y-axis shows the total loss while the one on the right corresponds to the loss rate. The X-axis shows the time in minutes. The generated fire is a type that engulfs the whole house if not mitigated, which will have the total loss of 1. In the response procedure, the first truck starts the mitigation at time 15 with its associated mitigation rate while at time 20 the second truck starts the mitigation. On the graph, around the 20 minute mark, the mitigation rate increases which in turn will reduce the total loss on the building. In this case the fire was out at time 27, with the total loss of 0.54.

Incident Generator

The model generates an event list at the beginning of each replication in order to conduct a discrete event simulation. It uses the call data provided by the user in the spreadsheets as an average or expected frequency for each event type. The inverse of this rate becomes the parameter for an exponential interarrival time. The model generates a new parameter and creates all events of a particular event type for the replication before continuing to the next event type. The team chose the exponential distribution because it is commonly used for interarrival times. Given more detailed data regarding the timing of events, a future team could fit a statistical distribution that more accurately reflects reality.

Incident Response

When the simulation advances to the next event on the event list, it calls on the appropriate incident model to adjudicate the event. Currently, the only fully functioning model is the residential fire generation model created by the Spring 2012 team. To demonstrate the capability of the simulation, the team has added some hardcoded data to the vehicle fire model and the fire generation of buildings other then residential. This is merely to demonstrate how future teams can interface with the main simulation as they build other incident models.

When the event is a building fire, the simulation will choose a location from the installation using the uniform distribution. It will then determine how far this location is from each station serving the installation and prioritize the stations. Stations on base are prioritized over those off base and then stations are prioritized by response time to the location of the incident. With this information, the simulation can search for available vehicles. The incident model provides the number and type of vehicles required. If when searching through stations in priority order, no

available vehicles can be found, the simulation will look for the next returning vehicle and mark it for the current incident. In addition to being already assigned to an incident, vehicles can also be unavailable due to maintenance. The simulation handles this by taking in a percentage, provided by the user on the Header tab, and comparing it to a uniform random number between 0 and 1. If the random number is less than the percentage, the otherwise available vehicle will be considered under maintenance for the period of time indicated by the user on the Header tab. After choosing the location of the incident and the vehicles to respond, the simulation informs the incident model of its assets. A hardcoded time of 10 minutes is added to the response time of each truck. This accounts for the time it takes for a person to notice the fire, attempt to deal with it, and call for help and the fire fighters to receive notification and get to their trucks. The model can then adjudicate the incident and provide the simulation with the total time for the response, from the arrival of the first vehicle to the time that the vehicles are able to depart the scene, and the amount of damage that occurred. The simulation then records these data points to be used in summary statistics and the log file at the end of the run.

Reporting Features

The analysis framework tool currently has two output modes. The first is a set of graphical information depicting loss and response time distributions, as these are the measure of effectiveness and measure of performance, respectively. The second is a generated text file with comma-separated values containing all the event information, including duration, response times, affected buildings, and loss amounts. This file could be utilized for further, more-detailed analysis of individual events or groups of event types across iterations. The tab delimited text file format is a common data format that can be opened with most statistical tools, including Excel, Matlab, R, and Octave.

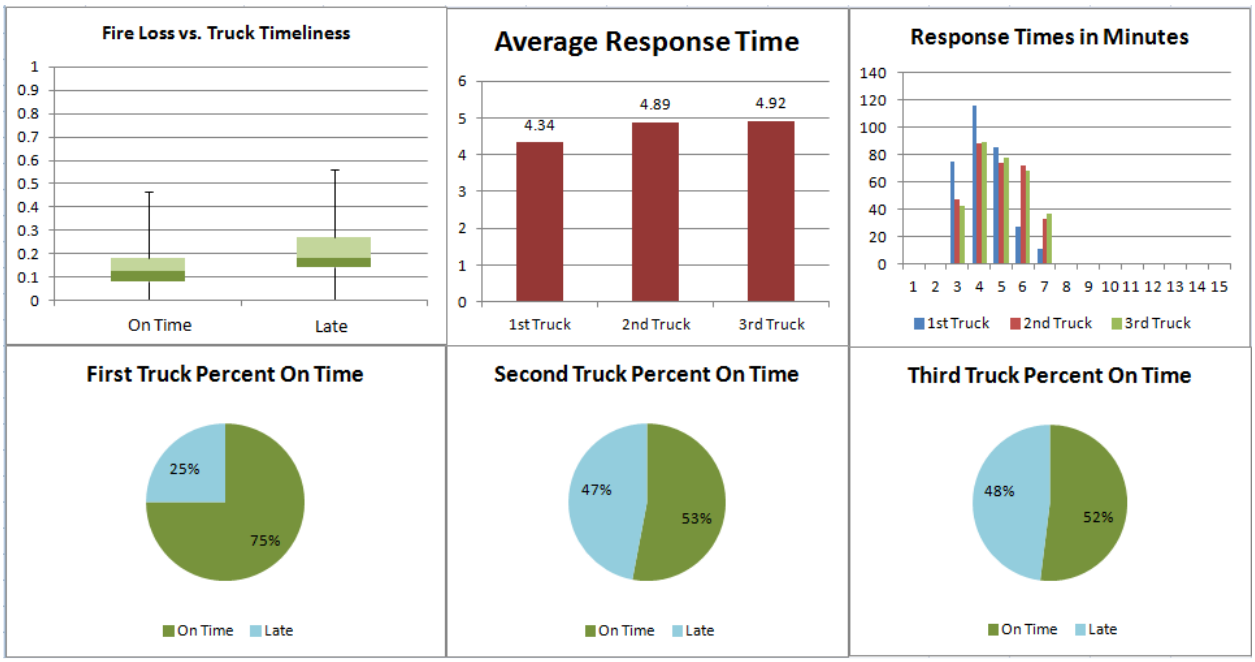


Figure 10 Sample Graphical Model Output

Event Type	Event Number	Replication	Start Time	Duration	Truck 1 Response Time	Truck 2 Response Time	Truck 3 Response Time	Location ID	Location	Damage
Building Fires	3	8	4515.42	78	5.37	5.82	5.82	PPV	PPV Housing	0.14
Building Fires	4	8	4599.57	79	3.11	6.24	6.24	PPV	PPV Housing	0.23
Vehicle Fires	5	8	5228.77	90	5.35	6.26	9999	PPV	PPV Housing	0.75
Building Fires	6	8	5389.54	80	4.9	4.9	4.9	PPV	PPV Housing	0.38
Vehicle Fires	7	8	5390.9	90	5.94	4.88	9999	PPV	PPV Housing	0.75
Building Fires	8	8	5559.83	90	4.81	4.81	4.81	CH 905	Morale Welfare & Recreation (MWR)/Youth Center	0.46
Building Fires	9	8	6320.04	77	4.45	4.45	4.45	PPV	PPV Housing	0.12
Building Fires	10	8	6629.69	76	3.93	3.93	3.93	PPV	PPV Housing	0.12
Building Fires	11	8	6959.73	74	3.15	3.15	3.15	PPV	PPV Housing	6.61E-002
Building Fires	12	8	7879.49	76	3.83	3.83	3.83	PPV	PPV Housing	0.17
Building Fires	13	8	7895.09	90	3.88	6.09	6.09	483	Public Works Dept	2.63E-003
Vehicle Fires	14	8	8074.95	90	5.32	6.41	9999	PPV	PPV Housing	0.75
Building Fires	15	8	8087.16	78	4.3	4.3	4.3	PPV	PPV Housing	0.22
Building Fires	16	8	8168.55	81	4.57	4.57	4.57	PPV	PPV Housing	0.4

Figure 11 Sample Raw Model Output

Analysis

Since the scope of this project was limited to the creation of an analysis framework and not focused on the analysis itself, this section primarily discusses the development of the framework and the proof-of-concept testing.

Framework Development

The analysis plan for the framework was to unit test each development component individually to ensure expected functionality before integrating it with other components. In this way, problems could be isolated and corrected before potentially being lost in the larger overall model. To this end, first the installation template was developed using a set of dummy data that contained at least one instance of each building type, station type, and vehicle type. Once that data was successfully entered into the template, it was then imported into the top-level of the model and mapped into the classes in VBA that would actually execute the model. Testing was conducted to confirm that all data in the template was imported successfully based on information in the header portion. The incident generator was tested using dummy call data to confirm that the distribution and random number generator were performing as expected and producing values in the expected ranges for given event types. The fire model was tested individually over a range of time scales to verify that the data it produced was compliant with the Weibull distribution established by the Spring 2012 project team. In doing so, some discrepancies were noted between the model and the documentation of the model. In particular, if the response time for the first fire truck was longer than the flashover point of the generated fire, the summing of the Weibull PDF and the mitigation function actually resulted in prolonging the fire. During this period, the intensity of the fire is beginning to die off on its own, so firefighting efforts should logically accelerate the extinguishing of the fire. Some logic was added to the model to account for such a case and modulate the mitigation function accordingly. Fortunately, this is a rare case, and one that is unlikely to occur in running the model if utilizing real-world response time data as inputs. Typical PDF rolloff times are around 20-25 minutes, requiring that all trucks either be assigned elsewhere or undergoing maintenance during that entire time.

One of the other goals of this analysis framework effort was to expand the applicability of the previous residential fire loss model to other building types and various building sizes. The existing model assumes a two story building with four rooms per floor, which is certainly adequate for modeling a typical single-family home. However, for modeling higher-density residential options such as barracks or high-rise apartments, or for modeling an office building or a large warehouse, the model simply couldn't be used in its existing state. Unfortunately, because the parameters of the model are largely driven by historical residential fire data on

ignition probabilities and spread rates, and the team could not locate such data due to the scarcity of incidents (compared to residential fires) to support populating the residential model with different parameters to model other structure types. Instead, other building types chosen for a fire are assigned a random damage percentage, depending on when the first fire engine arrives. If the first truck responds on-time (five minutes or less), the damage is assigned using a uniform random variable between 0 and 0.5. If the first company to respond is late, the damage is assigned using a uniform random variable between 0 and 1. While this may not be an accurate representation of realistic fire spread, it does serve the purpose of generating randomized loss data, and could be substituted for more representative models in the future.

Proof of Concept Analysis

To provide insights into the tool's application, two installation scenarios were generated. The first is a "dummy" model with a rudimentary set of buildings and vehicles meant to capture all the necessary elements for the simulation. The second is a more robust installation that approximates as closely as possible a real-world Navy installation. For the latter model, Submarine Base New London (SUBASE NLON) in Groton, Connecticut was selected. This particular base was selected because it contains a wide variety of building types, multiple fire stations, both onsite and via offsite mutual aid agreements, and a significant population size. In addition, the base does not contain an airfield, which greatly simplifies the simulation since airfields are governed by an entirely separate set of rules and regulations with respect to F&ES force size that are much less flexible than those applicable to other areas of a base.

The SUBASE NLON example provides the customer with a template for generating his own data sets while at the same time demonstrating the features of the analysis framework. For this semester's project, the team generated a set of loss data for the installation given the currently staffed F&ES forces, and then generated two alternative outcomes for comparison. The first scenario involved removal of a single fire engine from the first fire station and the second involved the removal of the entire second fire station, which primarily protects the residential area of the base. The results of each scenario are presented below for comparison.

Baseline Results

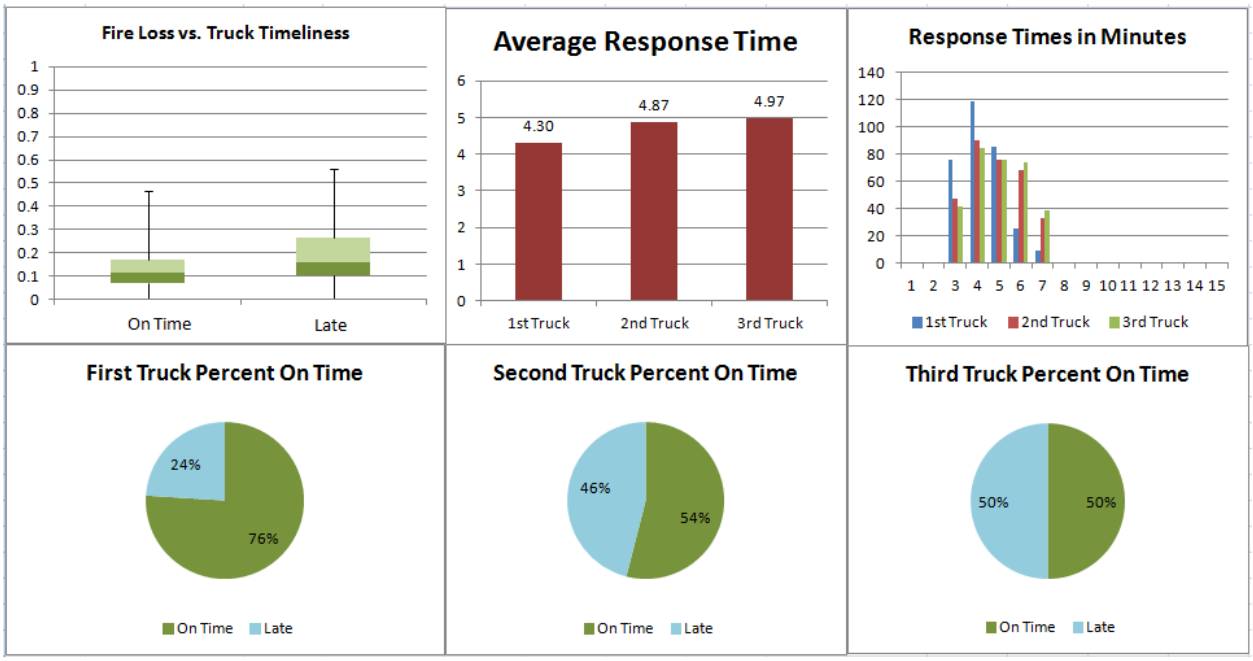


Figure 12 SUBASE NLON Baseline Results

F&ES Reduction 1 Results

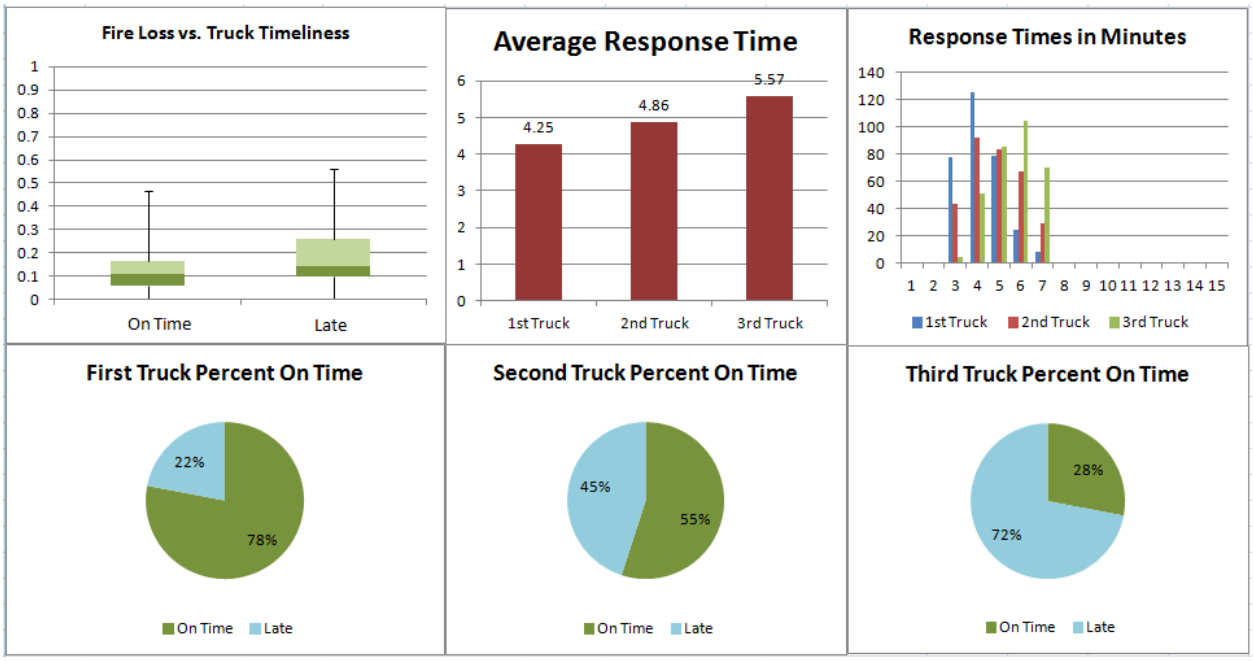


Figure 13 SUBASE NLON F&ES Reduction 1 Results

F&ES Reduction 2 Results

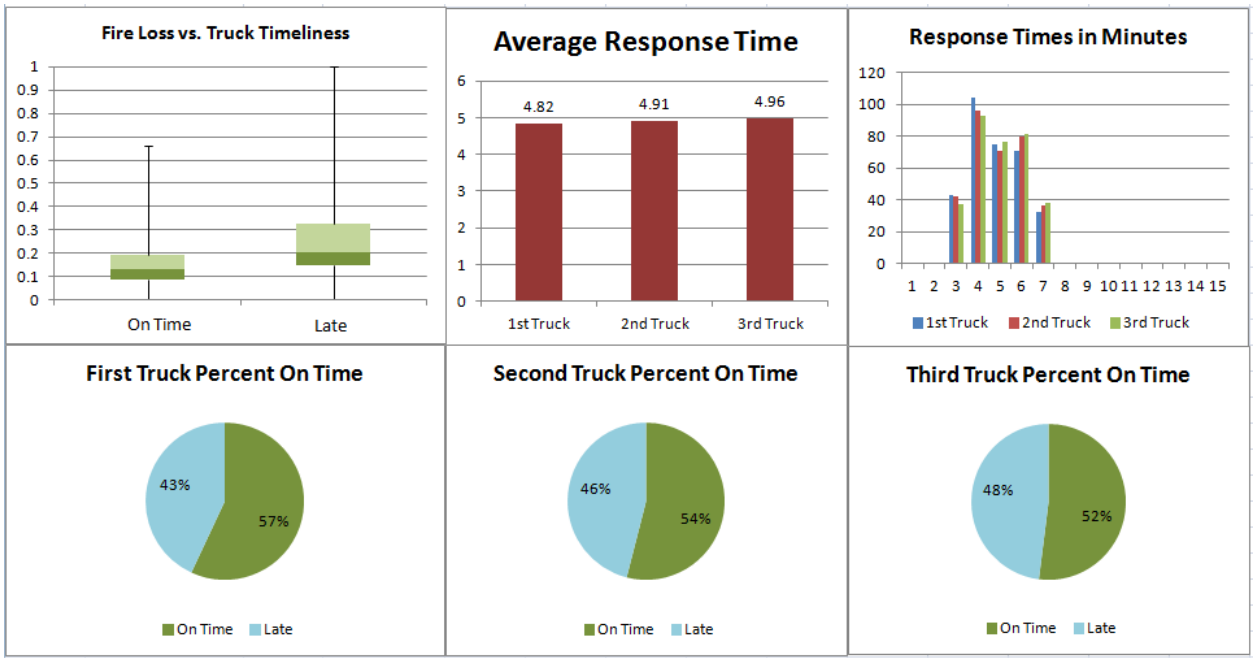


Figure 14 SUBASE NLON F&ES Reduction 2 Results

Comparative Analysis

The table below shows the detailed outcome of the three scenarios, compared side-by-side. There is very little difference between the baseline case and the first reduction case, where a single fire engine company was removed from onsite Station 1. This equipment removal does result in a minimal increase in the average arrival time of each responding company, causing a corresponding decrease in the on-time arrival percentages for the second and third responders. However, the effect on the overall loss during the period is essentially unchanged. The loss distribution shifted upward slightly, but the increase is statistically insignificant. This would suggest, at least within scope of the assumptions taken, that the third fire company at on-site station one may not be necessary. Since F&ES crews are only responding to fires in the model at this point in time, modeling additional events might drive up the loss as the demand for the vehicle increases. This outcome is expected, since there are still two fire companies at the station one, and one fire company at station two, plus four more fire companies at the mutual aid stations.

Measure	Baseline	Case 1	Case 2
Maximum loss (on-time arrival)	46.5%	46.5%	66.2%
Upper quartile loss (on-time arrival)	17.0%	16.6%	19.1%
Median loss (on-time arrival)	11.7%	10.7%	12.9%
Lower quartile loss (on-time arrival)	6.8%	6.0%	8.6%
Maximum loss (late arrival)	56.4%	56.4%	100.0%
Upper quartile loss (late arrival)	26.5%	26.2%	32.7%
Median loss (late arrival)	15.9%	14.5%	20.5%
Lower quartile loss (late arrival)	10.5%	9.9%	14.6%
Average response time (truck 1)	4.30	4.25	4.82
Average response time (truck 2)	4.87	4.86	4.91
Average response time (truck 3)	4.97	5.57	4.96
On time arrival percentage (truck 1)	76%	78%	57%
On time arrival percentage (truck 2)	54%	55%	54%
On time arrival percentage (truck 3)	50%	28%	52%

Figure 15 SUBASE NLON Comparative Analysis Table

For the second case, removing a fire station entirely clearly has more adverse affects than simply removing a single vehicle. The average arrival time for each responding company increased and the on-time percentages for the first responders fell significantly, which is the most important factor driving the loss. The later the first company arrives, the longer a fire will burn unmitigated, resulting in drastic loss increases. Correspondingly, the loss distribution for both on-time and late arrivals shifted upwards in this scenario. This loss increase would be even more drastic if the remaining onsite station was not able to cover the entire base with a nominal response time of five minutes, or if the two mutual aid stations offering fire services were farther away than five to six minutes. However, even with all of this working in favor of the installation, removing an onsite fire station is clearly detrimental to the expected loss. An interesting quirk in the output for this scenario is that the average response time of the third company and the percentage of on-time arrivals actually improved. This is due to a heavier utilization of the mutual aid stations once the on-site assets have been exhausted.

Overall, the model behaved as expected, yielding results that make some intuitive sense. That is, reducing the available emergency service assets tends to produce a slower response to incidents, which directly drives the expected loss from events such as fires. As a proof-of-concept test case, SUBASE NLON provided a good middle ground between smaller sites such as NSWC Carderock with primarily commercial and laboratory spaces and no residential buildings, and

sprawling Navy complexes such NAVSTA Norfolk with onsite airfields, massive manufacturing plants, and large supply depots.

Conclusions

Overall, the analysis framework as developed satisfies most of the objectives set forth for the semester.

Objective	Status
• To construct a model of a generalized installation that can be made specific given simple data for a particular installation.	• Objective met. Buildings can be entered individually or in homogenous groups. Call data and response times can be obtained directly from PCA reports.
• To build an efficient simulation model that will calculate expected losses using probabilistic loss models of various emergencies.	• Objective met. The default value of 30 iterations of the 1-year simulation at 1-minute resolution takes less than 2 seconds. 100 iterations can be run in less than 3 seconds.
• To include and expand on the residential fire probabilistic loss model.	• Partially met. The team was unable to locate large enough data sets to generate the required parameters for expansion of the existing residential loss model to cover alternative building types.
• To provide an interface to allow for simple addition of new probabilistic loss models for other emergency scenarios.	• Objective met. The VBA code contains placeholders for modeling other incident types with clear comments indicating where to plug in new elements and what inputs and outputs should be.

Figure 16 Fall 2012 Objective Status Table

The fundamental premise of the framework is that rather than providing answers to a limited set of pre-defined questions, it allows the Navy to ask many different questions related to the fire and emergency services. The framework allows for varying most of the inputs to examine the effect on the overall installation's expected loss presented from different angles. Additionally, the established measures of effectiveness and performance of expected loss and response time

are featured on the primary graphical outputs, making it easy for the user to examine the effects on those key elements. The modular nature is an important strength and allows significant room for future growth and increased model fidelity.

Future Expansion

During brainstorming and framework development, many ideas for future work and more granular analysis were discussed, but ultimately tabled in the interest of focusing on developing a functional framework first. Listed here are the ideas that the team identified as the most promising areas for framework expansion.

- Convert installation template into XML. XML could still be imported into Excel for simulation model usage, but decoupling the data from a proprietary format will make it easier for future access and modification.
- Consider optimization analysis for ideal F&ES station placement. If reducing the F&ES force size results in unacceptable expected loss, it might be possible to mitigate some or all of that loss by relocating or reallocating the remaining resources.

  There can be multiple optimization methods used. Basically the problem we are facing is the facility location problem in which facilities are fire stations and demand points are houses or incidents.

  Multiple objective functions can be considered for this general problem. In our case, coverage, costs, etc. can be the objective function. The problem F&ES is facing is reduced budget and what they need to figure out is that how reducing the budget will affect the loss in assets and lives. The dual to this problem is to minimize the cost of locating facilities while satisfying certain service level. However, deterministic approaches cannot be used to model this problem since the data on losses from fires and accidents are very random. Therefore, having the data for losses (which can be generated using simulation tools such as ours) we can use stochastic optimization approaches such as chance-constraint programming. In our setting, the objective is to minimize the cost while trying to satisfy the demand for emergency calls. The decisions are where to locate facilities from the potential facility locations. The general formulation can be set as follows:

$$\begin{aligned} &Min \quad Cost = cx \\ &s.t. \\ &P(Tx \le L) \ge p \\ &x \in B^n \end{aligned}$$

Figure 17 General Form for Cost Minimization Function

The objective function is to minimize the cost of locating facilities at each location. The general format for constraints states that the probability of having loss for locations greater than some threshold vector L should be less than 1-p. In this setting p states the service level defined for the system.

In this setting, the matrix T, is the loss matrix which is stochastic. It means that the loss associated with locations, due to fires and incidents are not known and are random. Scenarios can be generated using simulations or by SMEs.

The mathematical framework for solving stochastic optimization with random technology matrix has been developed recently by Lejeune. Please see the paper listed in the references section or more information.

In this case, changing the service level and solving the problem will help the decision maker to identify the tradeoff between service quality and costs.

The second approach for getting a better solution for locating facilities is to use simulation-optimization approach. The framework needed for this setting is to have the potential facility locations. There needs to be a method for calculating the average costs for losses and operations for different location allocations. The simulation then has to change the allocation of stations and run replications to get an estimate for the costs. The location allocation decisions can be made using heuristic methods.

- Expand the fire loss model to incorporate data on different building types. To generalize the fire loss model a graph theoretic approach can be used. For example, a building can be modeled as a graph G(V,E) in which V corresponds to set of nodes (vertex set) and E is the set of edges connecting those nodes. In this setting, each room in the building can be considered a node that belongs to set V. Edges are connecting rooms that are connected directly to each other which are either adjacent to each other or connected through the roof. The fire can start from a room (node) and each room has some material which can be considered as the fuel. For burning materials we can consider different

burning rate and probabilities for progressing to other rooms (based on connecting arcs) based on the stage of the fire and the heat that is released. Different graphs for different buildings can be made and simulation for the spread of fire in the buildings can be made. This can give a better estimation of loss to a building.

- Construct loss models for other incident types, such as vehicle collisions or medical treatments, that would include some other response models besides fire, such as paramedic response time in relation to personnel injuries or fatalities. These models, especially those for more prevalent incidents, would also lend further detail to the asset allocation aspects of the problem and continue to flesh out the utilization analysis of available equipment.
- Expand capabilities modeled by F&ES crew to a more discrete level, to include firefighter, EMT, etc, such that if crew members are available for a specific incident response, they must also possess the requisite skills. An EMT cannot be expected to drive a hook and ladder truck for instance, nor could a fire inspector necessarily be expected to perform CPR.
- Add in rules-based modules for handling airfield incidents.
- Add priority lists for determining which incident types take higher priority and whether or not a deployed F&ES company should be reassigned to a different incident that takes higher precedent. The current model operates purely on a first-come first-served basis.
- Add some randomization to the response time for each event individually, based on the nominal response time between the affected location and the station sending an F&ES vehicle to the scene.
- Build in parameters to represent how often mutual aid vehicles are unavailable due to servicing the local community as well as how often onsite vehicles are unavailable due to assisting mutual aid station emergency calls, if applicable.

References

- Butler Williams and Associates, LLC. "US Navy Submarine Base New London." Fire & Emergency Services Program Compliance Assessment. June 28, 2007.
- Hannan, Mosquera, and Vossler. "Navy Fire & Emergency Services: Model and Simulation of Structure Loss Due to Fire." George Mason University. May 7, 2012.[1]
- Coronado, Duda, Foroudi, and Tarakemeh. "Right Sizing Navy Fire and Emergency Services." George Mason University. December 1, 2011.[2]
- M. Lejeune. "Pattern-Based Modeling and Solution of Probability Constrained Optimization Problems." Operations Research. In Press, 2012[3]
- A Kogan, M. Lejeune. "Threshold Boolean Form for Joint Probabilistic Constraints With Random Technology Matrix." Submitted for publication, 2012.
- Chief of Naval Operations. "Shore Activities Fire Protection and Emergency Service Program." OPNAV Instruction 11320.23F Change Transmittal 2. May 28, 2004.
- Fire Data Analysis Handbook, Second Edition. FEMA FA-266. January 2004.

[1] http://seor.gmu.edu/projects/SEOR-Spring12/NavyFE/Project%20Documents/Navy%20F&ES%20Spring%202012%20Report%20Final.pdf
[2] http://seor.gmu.edu/projects/SEOR-Fall11/NFES/NFES_Final_Report.pdf
[3] http://www.academia.edu/539074/Pattern-Based_Modeling_and_Solution_of_Probabilistically_Constrained_Optimization_Problems

Appendix A: Acronyms and Definition of Terms

CDF	Cumulative Distribution Function
Company	A firefighting unit, consisting of a vehicle and the crew required to operate it.
F&ES	Fire and Emergency Services
FES	Fire and Emergency Services
FESPOM	Fire and Emergency Service Program and Objectives Memorandum
FP	Flashpoint
GMU	George Mason University
GWU	George Washington University
IDI	Innovative Decisions, Inc.
Mutual Aid	An agreement with offsite fire stations to provide and receive assistance in firefighting efforts.
NAS	Naval Air Station
NAVSTA	Naval Station
NLON	New London
NSWC	Naval Surface Warfare Center
PCA	Program Compliance Assessment
PDF	Probability Distribution Function
Pumper	A firefighting vehicle that must connect to a local water source upon arrival at the scene of a fire in order to carry out its duties.
SME	Subject Matter Expert
SUBASE	Submarine Base
Tanker	A firefighting vehicle that carries water onboard for immediate access upon arrival at the scene of a fire.
VBA	Visual Basic for Applications

Appendix B: Deliverables List

Deliverable	Due Date	Due To
Project Proposal and Presentation	10/4/12	Sponsor, Instructor
Status Report	10/11/12	Instructor
Webpage Design	10/11/12	Instructor
Interim Progress Report	10/18/12	Sponsor, Instructor
Final Presentation Draft	11/1/12	Instructor
Simulation Model	11/29/12	Sponsor, Instructor
Final Report	11/29/12	Sponsor, Instructor
Final Website	11/29/12	Instructor
Faculty Presentation	12/7/12	Sponsor, Instructor

Figure 18 Deliverables List

Appendix C: Work Breakdown Schedule

1. Project Management
    1.1. Project Kickoff
    1.2. Develop Project Definition
        1.2.1. Interview sponsor
        1.2.2. Present problem definition
    1.3. Develop Project Plan
        1.3.1. Assign tasking
        1.3.2. Layout project schedule
        1.3.3. Generate plan slides
        1.3.4. Submit project plan slides
        1.3.5. Present project plan
    1.4. Develop Project Proposal
        1.4.1. Write proposal
        1.4.2. Submit project proposal
    1.5. Prepare 1 page status update
        1.5.1. Write status update
        1.5.2. Submit status update
    1.6. Prepare IPR presentation
        1.6.1. Generate IPR slides
        1.6.2. Submit IPR presentation
        1.6.3. Present IPR
    1.7. Prepare final deliverables
        1.7.1. Prepare final report
            1.7.1.1. Generate final report
            1.7.1.2. Submit final report
        1.7.2. Prepare final presentation
            1.7.2.1. Generate final slides
            1.7.2.2. Submit presentation draft
            1.7.2.3. Dry run presentation
            1.7.2.4. Present project results
        1.7.3. Prepare project website
            1.7.3.1. Design website layout
            1.7.3.2. Submit website design
            1.7.3.3. Write HTML
            1.7.3.4. Upload files
2. Analytic Framework Development
    2.1. Conduct background research
        2.1.1. Break down previous models
            2.1.1.1. Identify inputs and outputs
            2.1.1.2. Identify assumptions made
            2.1.1.3. Identify underlying algorithms
        2.1.2. Identify common installation features
    2.2. Design framework
        2.2.1. Capture requirements

2.2.1.1. Identify stakeholder requirements
2.2.1.2. Decompose into system requirements
2.2.1.3. Generate SRS
2.2.2. Design modular loss model interface
2.2.3. Design generic installation descriptors
2.2.4. Design data input modality
2.2.5. Design user interface
2.3. Develop framework
2.3.1. Develop modular loss model interface
2.3.1.1. Code model interface
2.3.2. Develop generic installation descriptors
2.3.2.1. Code installation handler
2.3.3. Develop data input modality
2.3.3.1. Code data handler
2.3.4. Develop user interface
2.4. Test framework
2.4.1. Develop test case from existing base information and call data

Appendix D: Project Schedule

	WBS	Task Name	Duration	Start	Finish
1	**1**	Project Management	**62 days**	**Wed 9/12/12**	**Fri 12/7/12**
2	1.1	Project Kickoff	0 days	Wed 9/12/12	Wed 9/12/12
3	**1.2**	Develop Project Definition	**11 days**	**Wed 9/12/12**	**Thu 9/27/12**
4	1.2.1	Interview sponsor	7 days	Wed 9/12/12	Thu 9/20/12
5	1.2.2	Present problem definition	0 days	Thu 9/27/12	Thu 9/27/12
6	**1.3**	Develop Project Plan	**10 days**	**Thu 9/20/12**	**Thu 10/4/12**
7	1.3.1	Assign tasking	3 days	Thu 9/20/12	Mon 9/24/12
8	1.3.2	Layout project schedule	3 days	Mon 9/24/12	Wed 9/26/12
9	1.3.3	Generate slides	3 days	Thu 9/27/12	Mon 10/1/12
10	1.3.4	Submit project plan slides	0 days	Mon 10/1/12	Mon 10/1/12
11	1.3.5	Present project plan	0 days	Thu 10/4/12	Thu 10/4/12
12	**1.4**	Develop Project Proposal	**5 days**	**Thu 9/27/12**	**Thu 10/4/12**
13	1.4.1	Write proposal	5 days	Thu 9/27/12	Wed 10/3/12
14	1.4.2	Submit project proposal	0 days	Thu 10/4/12	Thu 10/4/12
15	**1.5**	Prepare 1-5 page status update	**1 day**	**Wed 10/10/12**	**Thu 10/11/12**
16	1.5.1	Write status update	1 day	Wed 10/10/12	Wed 10/10/12
17	1.5.2	Submit status update	0 days	Thu 10/11/12	Thu 10/11/12
18	**1.6**	Prepare IPR presentation	**4 days**	**Fri 10/12/12**	**Thu 10/18/12**
19	1.6.1	Generate IPR slides	3 days	Fri 10/12/12	Tue 10/16/12
20	1.6.2	Submit IPR presentation	0 days	Wed 10/17/12	Wed 10/17/12
21	1.6.3	Present IPR	0 days	Thu 10/18/12	Thu 10/18/12
22	**1.7**	Prepare final deliverables	**46 days**	**Thu 10/4/12**	**Fri 12/7/12**
23	**1.7.1**	Prepare final report	**29 days**	**Fri 10/19/12**	**Thu 11/29/12**
24	1.7.1.1	Generate final report	29 days	Fri 10/19/12	Wed 11/28/12
25	1.7.1.2	Submit final report	0 days	Thu 11/29/12	Thu 11/29/12
26	**1.7.2**	Prepare final presentation	**30 days**	**Fri 10/26/12**	**Fri 12/7/12**
27	1.7.2.1	Generate final slides	24 days	Fri 10/26/12	Wed 11/28/12
28	1.7.2.2	Submit presentation draft	0 days	Thu 11/1/12	Thu 11/1/12
29	1.7.2.3	Dry run presentation	6 days	Thu 11/29/12	Thu 12/6/12
30	1.7.2.4	Present project results	0 days	Fri 12/7/12	Fri 12/7/12
31	**1.7.3**	**Prepare project website**	**40 days**	**Thu 10/4/12**	**Thu 11/29/12**
32	1.7.3.1	Design website layout	5 days	Thu 10/4/12	Wed 10/10/12
33	1.7.3.2	Submit website design	0 days	Thu 10/11/12	Thu 10/11/12
34	1.7.3.3	Generate HTML	5 days	Thu 11/22/12	Wed 11/28/12
35	1.7.3.4	Upload files	0 days	Thu 11/29/12	Thu 11/29/12
36	**2**	Analytic Framework Development	**46 days**	**Thu 9/13/12**	**Thu 11/15/12**
37	**2.1**	Conduct background research	**11 days**	**Thu 9/20/12**	**Thu 10/4/12**
38	**2.1.1**	Break down previous models	**11 days**	**Thu 9/20/12**	**Thu 10/4/12**
39	2.1.1.1	Identify inputs and outputs	11 days	Thu 9/20/12	Thu 10/4/12
40	2.1.1.2	Identify assumptions made	11 days	Thu 9/20/12	Thu 10/4/12
41	2.1.1.3	Identify underlying algorithms	11 days	Thu 9/20/12	Thu 10/4/12
42	2.1.2	Identify common installation features	11 days	Thu 9/20/12	Thu 10/4/12
43	**2.2**	Design framework	**26 days**	**Thu 9/13/12**	**Thu 10/18/12**
44	**2.2.1**	Capture requirements	**21 days**	**Thu 9/13/12**	**Thu 10/11/12**
45	2.2.1.1	Identify stakeholder requirements	11 days	Thu 9/13/12	Thu 9/27/12
46	2.2.1.2	Decompose into system requirements	5 days	Fri 9/28/12	Thu 10/4/12
47	2.2.1.3	Generate SRS	5 days	Fri 10/5/12	Thu 10/11/12
48	2.2.2	Design modular loss model interface	6 days	Thu 10/11/12	Thu 10/18/12
49	2.2.3	Design generic installation descriptors	6 days	Thu 10/11/12	Thu 10/18/12
50	2.2.4	Design data input modality	6 days	Thu 10/11/12	Thu 10/18/12
51	2.2.5	Design user interface	6 days	Thu 10/11/12	Thu 10/18/12
52	**2.3**	Develop framework	**15 days**	**Fri 10/19/12**	**Thu 11/8/12**
53	**2.3.1**	Develop modular loss model interface	**10 days**	**Fri 10/19/12**	**Thu 11/1/12**
54	2.3.1.1	Code model interface	10 days	Fri 10/19/12	Thu 11/1/12
55	**2.3.2**	Develop generic installation descriptors	**10 days**	**Fri 10/19/12**	**Thu 11/1/12**
56	2.3.2.1	Code installation handler	10 days	Fri 10/19/12	Thu 11/1/12
57	**2.3.3**	Develop data input modality	**10 days**	**Fri 10/19/12**	**Thu 11/1/12**
58	2.3.3.1	Code data handler	10 days	Fri 10/19/12	Thu 11/1/12
59	2.3.4	Develop user interface	5 days	Fri 11/2/12	Thu 11/8/12
60	**2.4**	Test framework	**5 days**	**Fri 11/9/12**	**Thu 11/15/12**
61	2.4.1	Develop test case from	5 days	Fri 11/9/12	Thu 11/15/12

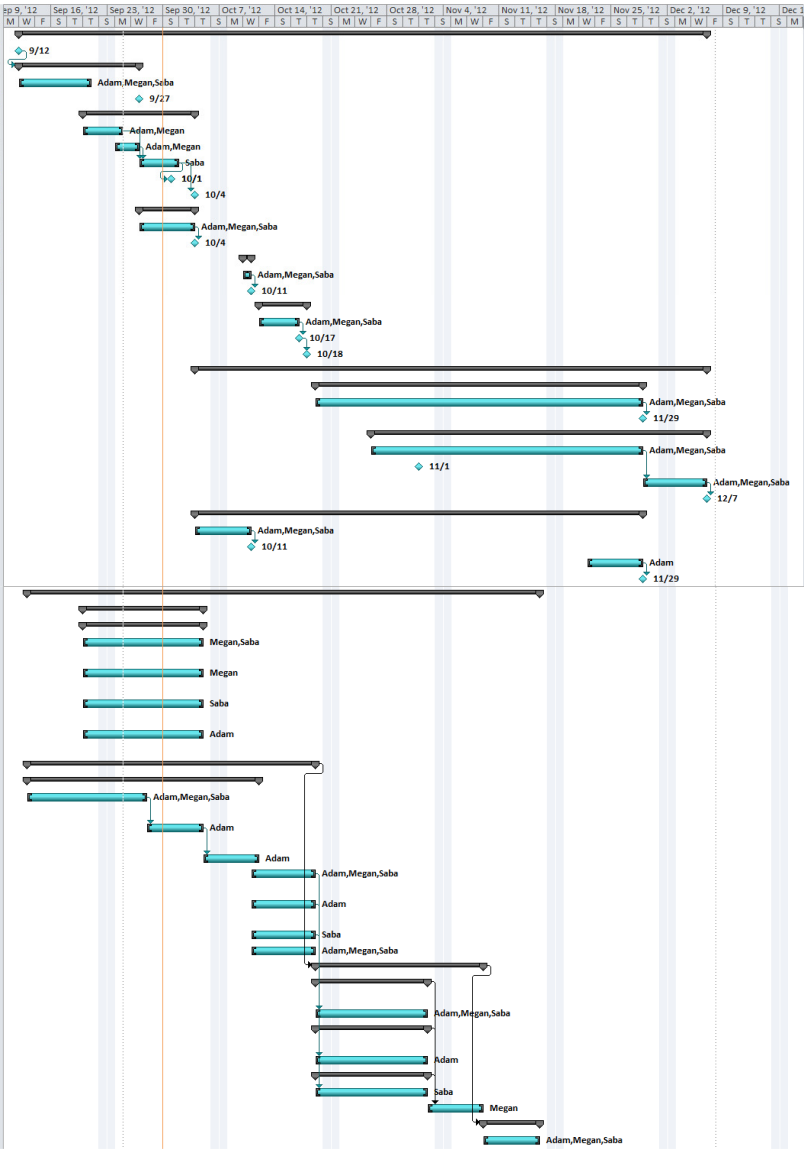


Figure 19 Project Schedule

Appendix E: System Requirements

The system defined by the following requirements consists of the analysis tool, algorithms, and external interfaces that will allow expected annual loss to be computed probabilistically for various installation configurations, including variable F&ES force sizes.

1. System Objective – The system shall consist of a modular analysis tool that enables the Navy to establish an estimated annual expected loss for any given installation as a function of the level of Fire and Emergency Services available to that installation.
    1.1. User Interface Requirements
        1.1.1. System Modality - The user interface shall be in Microsoft Excel.
        1.1.2. Input Requirements
            1.1.2.1. Data file – the system shall utilize a data file describing the installation to be analyzed. The data file will include building type and size parameters.
            1.1.2.2. Force size – the system shall allow the user to vary the force size
        1.1.3. Output Requirements
            1.1.3.1. Expected loss – the system shall output expected annualized loss estimates for the installation as a percentage. The final total will be an aggregate of the losses from each individual building within the installation.
            1.1.3.2. Data file – the system shall output a raw data file in a plain text format containing information on all generated events and response parameters.
            1.1.3.3. Graphical results – the system shall present summarized results in graphical format. The presented results will include at least a box plot of loss percentages, a bar graph of average response vehicle arrival times, a minute-resolution histogram of all arrival times across all iterations, and pie charts depicting on-time versus late arrival for average response times.
    1.2. Algorithm Requirements
        1.2.1. Loss Model Requirements
            1.2.1.1. The model shall compute expected loss for a single building for a single event.
            1.2.1.2. Input Requirements
                1.2.1.2.1. The algorithm shall use at least the following items as inputs: building size, number of floors, estimated fire team response times.
            1.2.1.3. Output Requirements
                1.2.1.3.1. The algorithm shall output the percentage loss of the building and the total time required to complete the incident response.
        1.2.2. Installation Model Requirements
            1.2.2.1. Input Requirements
                1.2.2.1.1. Installation Metrics
                1.2.2.1.2. Building Metrics
                1.2.2.1.3. Station Metrics– The installation model shall utilize events summary information from Program Compliance Assessment reports.
                1.2.2.1.4. Vehicle Metrics– The installation model shall utilize events summary information from Program Compliance Assessment reports.
                1.2.2.1.5. Call Data – The installation model shall utilize events summary information from Program Compliance Assessment reports.

1.2.2.2. Output Requirements

1.2.2.2.1. All input data shall be mirrored as output data into the simulation. Not all data elements must be used initially, but all must be made available to the simulation for future expansion.

1.2.3. Data Model Requirements

1.2.3.1. The system shall utilize Program Compliance Assessment (PCA) data to establish a baseline risk profile for a given installation.

2. System Training – The project team shall provide training on usage of the system to the Navy or its representatives prior to final delivery.
3. System Maintenance – The system shall be maintained by the Navy or its representatives following delivery.

Appendix F: Installation Template Definition

An installation requires the following data to be input to the template in order for the simulation model to process the expected loss for the base.

- Header
    - Name
    - Location (city, state)
    - Population (residents, civilians, contractors, etc)
    - Area (acres)
    - Buildings (including F&ES buildings)
    - Onsite F&ES Stations
    - Mutual Aid Stations
    - Vehicles
    - Maintenance Period (% unavailability)
    - Maintenance Time (in minutes)
    - Replications
    - Random Seed
- Buildings (list from 1 to N, where N is the number of buildings specified in the header)
    - Index
    - Building ID
    - Building Name
    - Building Type (see Enumerations)
    - Floors
    - Floor Area (square feet)
    - Units (if entry is a building group)
- Stations (list from 1 to N+M, where N is the number of onsite stations and M is the number of mutual aid stations specified in the header, with all onsite stations listed first)
    - Index
    - Name
    - Vehicles (total vehicles per station)
    - Crew (total crew per station)
    - Type (see Enumerations)
- Vehicles (list from 1 to N, where N is the number of vehicles specified in the header)
    - Index
    - Name
    - Type (see Enumerations)
    - Assigned Station (corresponds to station index)
    - Required Crew
- Response Time (a matrix of response times in minutes, sized N rows by M columns, where N is the number of buildings and M is the total number of fire stations, onsite and mutual aid)
- Call Data (extracted from an installation's most recent PCA)
    - Period Start
    - Period End
    - Incident Counts
        - Building Fires

        - Vehicle Fires
        - Other Fires
        - Overpressure Ruptures, Explosion, Overheat
        - Medical Treatment
        - Other Rescue
        - Hazardous Conditions
        - Service Calls
        - Good Intent
        - Severe Weather & Natural Disaster
        - Special Incident
        - Unknown Incident
        - Malicious False Calls
        - Other False Calls
    - Injury Counts (civilian here refers to everyone other than F&ES personnel)
        - Civilian Fire Injuries
        - Civilian Non-Fire Injuries
        - Civilian Fire Deaths
        - Civilian Non-Fire Deaths
        - F&ES Fire Injuries
        - F&ES Non-Fire Injuries
        - F&ES Fire Deaths
        - F&ES Non-Fire Deaths

- Enumerations
    - Building Types: Residential, Commercial, Retail, Warehouse, Manufacturing, Laboratory, Dock, Airfield
    - Vehicle Types: Pumper, Tanker, EMS, Hazmat, Command, Auxiliary
    - Station Types: Mil, Civ

Appendix G: Simulation Model Code

Simulation_Engine

```
Option Explicit

Type EventRecord
   EventType As String
   EventNum As Long
   Replication As Integer
   StartTime As Single
   duration As Integer
   truck1time As Single
   truck2time As Single
   truck3time As Single
   Damage As Single
   Location As String
   LocID As String
End Type

Sub main()
'Welcome to the Navy F&ES simulation!  This model is designed to tell the user what might
happen on an installation
'given the call history and the F&ES assets particular to a specific Navy installation.

Application.ScreenUpdating = False

Dim stations() As FireStation, property() As Building
Dim TL As Timeline, k As Integer
Dim i As Integer, j As Integer
Dim RV As RVGen, ID As Long
Dim probMaint As Single, MaintTime As Integer
Dim reps As Integer, line As Long
Dim record() As EventRecord, RandomSeed As Integer

Set TL = New Timeline
Set RV = New RVGen
ReDim record(1 To 1)

Worksheets("Header").Select
probMaint = Range("A1").Offset(8, 1)
MaintTime = Range("A1").Offset(9, 1)
reps = Range("A1").Offset(10, 1)
RandomSeed = Range("A1").Offset(11, 1)

Rnd -RandomSeed
```

```
Worksheets("Buildings").Select
j = 1
ID = 1
For i = 1 To 32000
   If Range("A1").Offset(i, 1) <> Empty Then
      ReDim Preserve property(j)
      Set property(j) = New Building
      Call property(j).DefineBuilding(Range("A1").Offset(i, 1), Range("A1").Offset(i, 2),
Range("A1").Offset(i, 3), _
         Range("A1").Offset(i, 4), Range("A1").Offset(i, 5))
      Let property(j).LB_ID = ID
      Let property(j).UB_ID = ID + Range("A1").Offset(i, 6) - 1
      ID = ID + Range("A1").Offset(i, 6)
      j = j + 1
   Else: Exit For
   End If
Next i

Worksheets("Stations").Select
j = 1
For i = 1 To 32000
   If Range("A1").Offset(i, 1) <> Empty Then
      ReDim Preserve stations(j)
      Set stations(j) = New FireStation
      stations(j).DefineStation Range("A1").Offset(i, 3), Range("A1").Offset(i, 4),
Range("A1").Offset(i, 1)
      Call stations(j).MaintInfo(probMaint, MaintTime)
      j = j + 1
   Else: Exit For
   End If
Next i

Worksheets("ResponseTime").Select
j = 1
Call property(j).NumStations(UBound(stations, 1))
For i = 3 To 32000
   If Range("A1").Offset(i, 1) <> Empty Then
      Call property(j).NumStations(UBound(stations, 1))
      For k = 3 To 20
         If Range("A1").Offset(i, k - 1) <> Empty Then
            Call property(j).SetDistance(k - 2, Range("A1").Offset(i, k - 1))
         End If
      Next k
      j = j + 1
   Else: Exit For
   End If
```

```
Next i

Worksheets("Vehicles").Select
For i = 1 To 32000
   If Range("A1").Offset(i, 1) <> Empty Then
      Call stations(Range("A1").Offset(i, 3)).AddTruck(Range("A1").Offset(i, 2))
   Else: Exit For
   End If
Next i


'Now that all the input data is organized, the simulation can begin.
line = 1
For i = 1 To reps
   Set TL = New Timeline
   Call BuildTL(TL, RV) 'Create a new event list.
   For j = 1 To UBound(stations, 1) 'Make sure all trucks are in their stations.
      Call stations(j).ResetTrucks
   Next j
   ReDim Preserve record(1 To UBound(record, 1) + TL.NumOfEvents - 1) 'Create more space
in the log.
   Call RunSim(property, stations, TL, RV, ID - 1, record, line, i)
Next i

'Summarize and print the statistics and log.
Call SummaryResults(record)
Call WriteToFile(record, RandomSeed)

Application.ScreenUpdating = True

End Sub

Sub BuildTL(TL As Timeline, RV As RVGen)
'The purpose of this sub routine is to create a year's worth of emergencies for an installation.  It
uses
'the exponential distribution to set the interarrival times of events based on the historical
frequency of
'that event type as declared in the call data.

Dim y As Single, days As Integer
Dim hours As Single, currentTime As Single
Dim i As Integer, DataHours As Long

Worksheets("CallData").Select
DataHours = Range("B3") * 24
```

```
'Currently the time horizon is hard coded to be a year and the simulation clock is set to calculate
hours.
hours = 365 * 24

'Each event type has its own freqency and therefore must be handled separately.
For i = 4 To 17

   If Range("A1").Offset(i, 1) = Empty Then Exit For

   If Range("A1").Offset(i, 1) <> 0 Then

      'Each time a new event type is selected, we go back to the first day of the simulation.
      currentTime = 1

      'y is the lambda and is calculated as the inverse of the frequency for this event type.
      y = DataHours / Range("A1").Offset(i, 1)

      'Interarrival times are calculated and events are hung on the timeline.
      Do While currentTime < hours
         currentTime = currentTime + RV.RV("Exponential", y)
         If currentTime < hours Then Call TL.AddEvent(currentTime, Range("A1").Offset(i))
      Loop

   End If
Next i

Call TL.AddEvent(hours, "End")

End Sub

Sub RunSim(property() As Building, stations() As FireStation, TL As Timeline, RV As RVGen,
NumBuild As Long, _
   record() As EventRecord, line As Long, rep As Integer)
'This sub routine is where the primary action happens within a single replication.

Dim currentTime As Single, iBF() As BuildingFire
Dim iOF() As OtherFires, iOREO() As OREO
Dim iMT() As Med_Treat, iOR() As OtherRescue
Dim iHC() As HazCond, iSC() As ServiceCall
Dim iGI() As GoodIntent, iSWND() As SW_ND
Dim iSI() As SpecialIndicent, iUI() As UnkIncident
Dim iMFC() As MalFalseCall, iOFC() As OtherFalseCall
Dim iVF() As VehicleFire, x As Integer
Dim trucks() As Single, j As Integer
Dim i As Integer, eID As Integer
Dim station As Integer, k As Integer
```

```
Dim dist() As Single, response As Integer
Dim t As Integer, tmp As Integer
Dim Location As Integer, TotTime As Integer
Dim pTotLoss As Single, complete As Integer
Dim minResponse() As Integer, NextEvent As String
Dim EndSim As Boolean, eventCount As Integer
Dim z As Single, order() As Integer

EndSim = False
currentTime = TL.Time
ReDim unfilled(1 To 5, 1 To 1)
ReDim iBF(1 To 1)
ReDim iVF(1 To 1)
ReDim iOF(1 To 1)
ReDim iOREO(1 To 1)
ReDim iMT(1 To 1)
ReDim iOR(1 To 1)
ReDim iHC(1 To 1)
ReDim iSC(1 To 1)
ReDim iGI(1 To 1)
ReDim iSWND(1 To 1)
ReDim iSI(1 To 1)
ReDim iUI(1 To 1)
ReDim iMFC(1 To 1)
ReDim iOFC(1 To 1)
ReDim dist(1 To UBound(stations, 1), 1 To 2)

'The way the timeline works, the first time is a blank.  So this step gets that point out of the way.
NextEvent = TL.NextEvent
eventCount = 1

Do While EndSim = False

   'Advance to the next event and time.
   NextEvent = TL.NextEvent
   currentTime = TL.Time

   'Select the correct event type.
   Select Case NextEvent
      Case "Building Fires"

         ReDim Preserve iBF(1 To UBound(iBF, 1) + 1)
         eID = UBound(iBF, 1)
         Set iBF(eID) = New BuildingFire
```

```
        'The array trucks will store information on the ID, response time, and home station of
each truck.
        ReDim trucks(1 To 3, 1 To iBF(eID).nTrucks)
        ReDim order(1 To UBound(trucks, 2))

        'The location of the fire is randomly selected.
        Location = CInt(RV.RV("uniform", 1, CSng(NumBuild)))

        'To handle groups of buildings, each property has a range of building numbers contained
in it location.
        For x = 1 To UBound(property, 1)
           If Location >= property(x).LB_ID And Location <= property(x).UB_ID Then Location
= x
        Next x

        'Find the distance from this location to each station.
        For x = 1 To UBound(dist, 1)
           If property(Location).LB_ID = property(Location).UB_ID - 1 Then
              dist(x, 1) = property(Location).GetDistance(x) 'Not a cluster of buildings.
              dist(x, 2) = stations(x).StationType
           Else: 'One building in a cluster.
              dist(x, 1) = property(Location).GetDistance(x) + RV.RV("uniform", -2, 2)
              If LCase(stations(x).StationType) = "mil" Then
                 dist(x, 2) = 1
              Else: dist(x, 2) = 0
              End If
           End If
        Next x

        'The fire event needs all the information about structure on fire.
        Call iBF(eID).BuildingInfo(property(Location).floors, property(Location).area, Location,
_
           property(Location).bType)

        'Search for the right trucks to send and keep accountability on where they are and when
they can return.
        For t = 1 To UBound(trucks, 2) 'This is the number of trucks required for this event.
           Call FindTruck(stations, property, trucks, RV, dist, station, t, "Pumper", currentTime,
Location, complete)
        Next t

        Call OrderTrucks(order, trucks)

        For t = 1 To UBound(trucks, 2) 'This is the number of trucks required for this event.
           Call iBF(eID).AddTruck(Int(trucks(1, order(t))), Int(trucks(2, order(t))), station)
        Next t
```

```
        'Mark all station options as unvisited for that building.
        Call property(Location).Reset

        'Kick of the fire and get back the percentage of the house that was destroyed and the time
needed for mitigation
        TotTime = iBF(eID).StartEvent(pTotLoss)

        'Note at what time the trucks will return to their stations.
        complete = currentTime + ((trucks(2, 1) + TotTime) / 60)
        For t = 1 To UBound(trucks, 2)
           Call stations(trucks(3, t)).ReturnTime(Int(trucks(1, t)), "Pumper", trucks(2, t) / 60 +
complete)
        Next t

        'Record the event in the simulation log
        Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
trucks(2, order(1)), _
            trucks(2, order(2)), trucks(2, order(3)), pTotLoss, property(Location).name,
property(Location).ID)

        line = line + 1
        eventCount = eventCount + 1

    Case "Vehicle Fires"

        ReDim Preserve iVF(1 To UBound(iVF, 1) + 1)
        eID = UBound(iVF, 1)
        Set iVF(eID) = New VehicleFire

        'The array trucks will store information on the ID, response time, and home station of
each truck.
        ReDim trucks(1 To 3, 1 To iVF(eID).nTrucks)
        ReDim order(1 To UBound(trucks, 2))

        'The location of the fire is randomly selected.
        Location = CInt(RV.RV("uniform", 1, CSng(NumBuild)))

        'To handle groups of buildings, each property has a range of building numbers contained
in it location.
        For x = 1 To UBound(property, 1)
           If Location >= property(x).LB_ID And Location <= property(x).UB_ID Then Location
= x
        Next x

        'Find the variable distance from this location to each station.
```

```
        For x = 1 To UBound(dist, 1)
           If property(Location).LB_ID = property(Location).UB_ID - 1 Then
              dist(x, 1) = property(Location).GetDistance(x) 'Not a cluster of buildings.
              dist(x, 2) = stations(x).StationType
           Else: 'One building in a cluster.
              dist(x, 1) = property(Location).GetDistance(x) + RV.RV("uniform", -2, 2)
              If LCase(stations(x).StationType) = "mil" Then
                 dist(x, 2) = 1
              Else: dist(x, 2) = 0
              End If
           End If
        Next x

        'Search for the right trucks to send and keep accountability on where they are and when
they can return.
        For t = 1 To UBound(trucks, 2) 'This is the number of trucks required for this event.
           Call FindTruck(stations, property, trucks, RV, dist, station, t, "Pumper", currentTime,
Location, complete)
        Next t

        Call OrderTrucks(order, trucks)

        For t = 1 To UBound(trucks, 2) 'This is the number of trucks required for this event.
           Call iVF(eID).AddTruck(Int(trucks(1, order(t))), Int(trucks(2, order(t))), station)
        Next t

        'Mark all station options as unvisited for that building.
        Call property(Location).Reset

        'Kick of the fire and get back the percentage of the house that was destroyed and the time
needed for mitigation
        TotTime = iVF(eID).StartEvent(pTotLoss)

        'Note at what time the trucks will return to their stations.
        complete = currentTime + ((trucks(2, 1) + TotTime) / 60)
        For t = 1 To UBound(trucks, 2)
           Call stations(trucks(3, t)).ReturnTime(Int(trucks(1, t)), "Pumper", trucks(2, t) / 60 +
complete)
        Next t

        Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
trucks(2, order(1)), _
           trucks(2, order(2)), 9999, pTotLoss, property(Location).name, property(Location).ID)
        line = line + 1
        eventCount = eventCount + 1
```

```
        Case "Other Fires"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Overpressure Ruptures, Explosion, Overheat"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Medical Treatment"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Other Rescue"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Hazardous Conditions"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Service Calls"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
```

```
'           eventCount = eventCount + 1

        Case "Good Intent"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Severe Weather & Natural Disaster"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1
'
        Case "Special Incident"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Unknown Incident"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Malicious False Calls"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "Other False Calls"

'           Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, TotTime,
Trucks(2, 1), Trucks(2, 2), _
```

```
'               Trucks(2, 3), pTotLoss, property(Location).name)
'           line = line + 1
'           eventCount = eventCount + 1

        Case "End"

            Call SaveRecord(record, line, NextEvent, eventCount, rep, currentTime, 9999, 9999,
9999, 9999, 9999, "N/A", "N/A")
            line = line + 1
            eventCount = eventCount + 1

            EndSim = True
    End Select
Loop

End Sub

Function FindTruck(stations() As FireStation, property() As Building, trucks() As Single, RV As
RVGen, _
    dist() As Single, station As Integer, t As Integer, tType As String, currentTime As Single, _
    Location As Integer, complete As Integer)
'This function searches for the best trucks to respond to the incident.

Dim i As Integer, j As Integer
Dim tmp As Integer, priority() As Integer
Dim minResponse() As Variant


ReDim minResponse(1 To 2)
ReDim priority(1 To UBound(stations, 1))

Call property(Location).PriorityStation(dist, priority)

'First look for a truck that is in its station.
For i = 1 To UBound(priority, 1)
    'Look through stations in priority order based on the location of the incident.
    station = priority(i)
    trucks(1, t) = stations(station).SendTruck(Location, tType, currentTime, RV)
    If trucks(1, t) <> 0 Then 'Values other than 0 are the ID number of a truck
        trucks(3, t) = station 'Store the home station of that truck.
        trucks(2, t) = dist(station, 1) 'Store the distance from the station to the incident.
        Call stations(station).ReturnTime(Int(trucks(1, t)), tType, 9999)
        Exit For
    End If
Next i
```

```
'If no trucks were in the station, then look for the next one to return.
If trucks(1, t) = 0 Then
   minResponse(1) = 1000
   For j = 1 To UBound(priority, 1)
      station = priority(j)
      tmp = stations(station).BestResponse(tType, Int(trucks(1, t)))
      If tmp > 0 Then
         If tmp < minResponse(1) Then
            minResponse(1) = 60 * (tmp - currentTime) + dist(station, 1)
            minResponse(2) = station
         End If
      End If
   Next j
   trucks(2, t) = minResponse(1) 'This is the response time to incident.
   trucks(3, t) = minResponse(2) 'This is the home station for the truck.
   Call stations(station).ReturnTime(Int(trucks(1, t)), tType, 9999)
End If



End Function

Sub OrderTrucks(order() As Integer, trucks() As Single)

Dim accounted() As Integer, mintime As Single
Dim vehicle As Integer, i As Integer
Dim j As Integer

ReDim accounted(1 To UBound(trucks, 2))

For i = 1 To UBound(order, 1)
   mintime = 100
   For j = 1 To UBound(trucks, 2)
      If trucks(2, j) < mintime And accounted(j) = 0 Then
         order(i) = j
         mintime = trucks(2, j)
      End If
   Next j
   accounted(order(i)) = 1
Next i

End Sub


Sub SaveRecord(record() As EventRecord, line As Long, NextEvent As String, eventCount As
Integer, rep As Integer, _
```

```
    currentTime As Single, duration As Integer, truck1time As Single, truck2time As Single,
truck3time As Single, _
    Damage As Single, Location As String, LocID As String)
'This sub routine logs the events of each incident.  It can and should be expanded as other
incident modules are filled out.

    record(line).EventType = NextEvent
    record(line).EventNum = eventCount
    record(line).Replication = rep
    record(line).StartTime = currentTime
    record(line).duration = duration
    record(line).truck1time = truck1time
    record(line).truck2time = truck2time
    record(line).truck3time = truck3time
    record(line).Damage = Damage
    record(line).Location = Location
    record(line).LocID = LocID

End Sub

Sub SummaryResults(record() As EventRecord)
'This sub routine calculates the statistics of interest that feed charts in the Results worksheet. It
can and should
'be expanded to display results of other incident modules.

'FL = Fire Loss; RT = Response Time; OT = On Time; L = Late; t = truck
Dim FLmin As Single, FLavg As Single
Dim FLmax As Single, FLcount As Integer
Dim RTt1avg As Single, t1count As Integer
Dim RTt2avg As Single, t2count As Integer
Dim RTt3avg As Single, t3count As Integer
Dim OTt1 As Single, Lt1 As Single
Dim OTt2 As Single, Lt2 As Single
Dim OTt3 As Single, Lt3 As Single
Dim i As Integer, RTHist() As Integer
Dim aOT() As Single, aL() As Single
Dim OTmin As Single, OT1quart As Single
Dim OTmed As Single, OT3quart As Single
Dim OTmax As Single, Lmin As Single
Dim L1quart As Single, Lmed As Single
Dim L3quart As Single, Lmax As Single

FLmin = 100
ReDim RTHist(1 To 15, 1 To 3)
ReDim aOT(1 To 1)
ReDim aL(1 To 1)
```

```
For i = 1 To UBound(record, 1)

    If record(i).EventType = "Building Fires" Then

        If record(i).EventType = "Building Fires" Then
            If record(i).Damage < FLmin Then FLmin = record(i).Damage
            If record(i).Damage > FLmax Then FLmax = record(i).Damage
            FLavg = FLavg + record(i).Damage
            FLcount = FLcount + 1

            RTt1avg = RTt1avg + record(i).truck1time
            RTt2avg = RTt2avg + record(i).truck2time
            RTt3avg = RTt3avg + record(i).truck3time
            t1count = t1count + 1
            t2count = t2count + 1
            t3count = t3count + 1
            RTHist(record(i).truck1time, 1) = RTHist(record(i).truck1time, 1) + 1
            RTHist(record(i).truck2time, 2) = RTHist(record(i).truck2time, 2) + 1
            RTHist(record(i).truck3time, 3) = RTHist(record(i).truck3time, 3) + 1

            If record(i).truck1time <= 5 Then
                OTt1 = OTt1 + 1
                ReDim Preserve aOT(1 To UBound(aOT, 1) + 1)
                aOT(UBound(aOT, 1)) = record(i).Damage
            Else:
                Lt1 = Lt1 + 1
                ReDim Preserve aL(1 To UBound(aL, 1) + 1)
                aL(UBound(aL, 1)) = record(i).Damage
            End If

            If record(i).truck2time <= 5 Then
                OTt2 = OTt2 + 1
            Else:
                Lt2 = Lt2 + 1
            End If

            If record(i).truck3time <= 5 Then
                OTt3 = OTt3 + 1
            Else:
                Lt3 = Lt3 + 1
            End If
        End If

    End If
```

```
Next i

FLavg = FLavg / FLcount
RTt1avg = RTt1avg / t1count
RTt2avg = RTt2avg / t2count
RTt3avg = RTt3avg / t3count
OTmin = WorksheetFunction.Quartile(aOT, 0)
OT1quart = WorksheetFunction.Quartile(aOT, 1)
OTmed = WorksheetFunction.Quartile(aOT, 2)
OT3quart = WorksheetFunction.Quartile(aOT, 3)
OTmax = WorksheetFunction.Quartile(aOT, 4)
Lmin = WorksheetFunction.Quartile(aL, 0)
L1quart = WorksheetFunction.Quartile(aL, 1)
Lmed = WorksheetFunction.Quartile(aL, 2)
L3quart = WorksheetFunction.Quartile(aL, 3)
Lmax = WorksheetFunction.Quartile(aL, 4)

Worksheets("Results").Select
Range("E2") = RTt1avg
Range("E3") = RTt2avg
Range("E4") = RTt3avg
Range("H3") = Round(OTt1 / t1count, 2)
Range("I3") = Round(Lt1 / t1count, 2)
Range("H4") = Round(OTt2 / t2count, 2)
Range("I4") = Round(Lt2 / t2count, 2)
Range("H5") = Round(OTt3 / t3count, 2)
Range("I5") = Round(Lt3 / t3count, 2)
Range("L3:N17") = RTHist
Range("Q3") = OTmax
Range("Q4") = OT3quart
Range("Q5") = OTmed
Range("Q6") = OT1quart
Range("Q7") = OTmin
Range("R3") = Lmax
Range("R4") = L3quart
Range("R5") = Lmed
Range("R6") = L1quart
Range("R7") = Lmin
Range("Q11") = OTmax - OT3quart
Range("Q12") = OT3quart - OTmed
Range("Q13") = OTmed - OT1quart
Range("Q14") = OT1quart
Range("Q15") = OT1quart - OTmin
Range("R11") = Lmax - L3quart
Range("R12") = L3quart - Lmed
Range("R13") = Lmed - L1quart
```

```
Range("R14") = L1quart
Range("R15") = L1quart - Lmin

End Sub

Sub WriteToFile(record() As EventRecord, RandomSeed As Integer)
'This sub routine writes the log to a file that will be placed in the same folder as the workbook.

Dim line As String, file As String
Dim filenum As Integer, i As Integer

Worksheets("Header").Select

filenum = FreeFile
'File name includes the name of the base and the random number used in the run.
file = ThisWorkbook.Path & "\" & CStr(Range("A1").Offset(, 1)) & "SimLog_" &
CStr(RandomSeed) & ".txt"

Open file For Output As #filenum

line = "Event Type" + vbTab
line = line + "Event Number" + vbTab
line = line + "Replication" + vbTab
line = line + "Start Time" + vbTab
line = line + "Duration" + vbTab
line = line + "Truck 1 Response Time" + vbTab
line = line + "Truck 2 Response Time" + vbTab
line = line + "Truck 3 Response Time" + vbTab
line = line + "Location" + vbTab
line = line + "Location ID" + vbTab
line = line + "Damage"

Print #filenum, line

For i = 1 To UBound(record, 1)

    line = ""

    line = record(i).EventType + vbTab
    line = line + CStr(record(i).EventNum) + vbTab
    line = line + CStr(record(i).Replication) + vbTab
    line = line + CStr(record(i).StartTime) + vbTab
    line = line + CStr(record(i).duration) + vbTab
    line = line + CStr(record(i).truck1time) + vbTab
    line = line + CStr(record(i).truck2time) + vbTab
    line = line + CStr(record(i).truck3time) + vbTab
```

```
    line = line + record(i).Location + vbTab
    line = line + record(i).LocID + vbTab
    line = line + CStr(record(i).Damage)

    Print #filenum, line

Next i

Close filenum

End Sub
```

Building

```
Option Explicit
'This is the Building object.  It holds general information about the infrastructure on an
installation.

Public ID As String
Public name As String
Private category As String
Public floors As Integer
Public area As Long
Private distance() As Integer
Public UB_ID As Long
Public LB_ID As Long
Private StationTried() As Integer
Public bType As String

Public Sub DefineBuilding(num As String, called As String, use As String, stories As Integer,
space As Single)

ID = num 'ID numbers sometimes contain letters.
name = called
category = use
floors = stories
area = space
bType = use

End Sub

Public Sub NumStations(num As Integer)

ReDim distance(1 To num)
ReDim StationTried(1 To num)
```

```
End Sub

Public Sub SetDistance(station As Integer, dist As Single)

distance(station) = dist

End Sub

Public Sub PriorityStation(dist() As Single, priority() As Integer)
'This sub routine allows a building to determine which station should be checked for an available truck next.
'Stations are prioritized first by whether or not they are on base and second by the distance from the incident.

Dim mindist As Single, i As Integer
Dim minstation As Integer, j As Integer
Dim visited() As Integer, mil As Integer

'Count how many stations are military run (or on base).
For i = 1 To UBound(dist, 1)
   If dist(i, 2) = 1 Then mil = mil + 1
Next i

'This array ensures that all stations are included exactly once.
ReDim visited(1 To UBound(priority, 1))

'First priortize the stations on base by time.
For j = 1 To mil
   mindist = 30
   For i = 1 To UBound(dist, 1)
      If dist(i, 1) <= mindist And visited(i) = 0 And dist(i, 2) = 1 Then
         mindist = dist(i, 1)
         minstation = i
      End If
   Next i
   priority(j) = minstation
   visited(minstation) = 1
Next j

'Next prioritize stations off base by time and add them after the stations on base.
For j = mil + 1 To UBound(dist, 1)
   mindist = 30
   For i = 1 To UBound(dist, 1)
      If dist(i, 1) <= mindist And visited(i) = 0 Then
         mindist = dist(i, 1)
         minstation = i
```

```
        End If
    Next i
    priority(j) = minstation
    visited(minstation) = 1
Next j

End Sub

Public Function GetDistance(station As Integer) As Integer

GetDistance = distance(station)

End Function

Public Sub Reset()

Dim i As Integer

For i = 1 To UBound(StationTried, 1)
    StationTried(i) = 0
Next i

End Sub
```

BuildingFire

```
Option Explicit
'This is the module for building fire events.  The fire generation function was designed by the
Spring 2012 GMU team.
'It was then coded and refined by the Fall 2012 team.  This function only generates fires for two
story residential buildings.
'All other buildings are assigned a hardcoded duration and total loss.  This will need to be
expanded by future teams.

'n = need; h = have
Public nTrucks As Integer
'Public nEMS As Integer
Private hTrucks() As Integer
'Private hEMS() As Integer
Private floors As Integer
Private area As Long
Private Location As Integer
Private RV As RVGen
Private bType As String

Private Sub Class_Initialize()
```

```
ReDim hTrucks(1 To 3, 1 To 3)
'ReDim hEMS(1 To 1, 1 To 3)
nTrucks = 3
Set RV = New RVGen

End Sub

Public Sub AddTruck(truckID As Integer, responseTime As Integer, stationID As Integer)
'Each truck is added individually as the main simulation determines the next best truck to send.

Dim i As Integer

For i = 1 To UBound(hTrucks, 1)
   If hTrucks(i, 1) = 0 Then
      hTrucks(i, 1) = truckID
      hTrucks(i, 2) = responseTime
      hTrucks(i, 3) = stationID
      Exit For
   End If
Next i

nTrucks = nTrucks - 1

End Sub

'Public Sub addEMS(truckID As Integer, responseTime As Integer, stationID As Integer)
'
'
'Dim i As Integer
'
'For i = 1 To UBound(hEMS, 1)
'    If hEMS(i, 1) = 0 Then
'        hEMS(i, 1) = truckID
'        hEMS(i, 2) = responseTime
'        hEMS(i, 3) = stationID
'    End If
'Next i
'
'nEMS = nEMS - 1
'
'End Sub

Public Sub BuildingInfo(levels As Integer, space As Long, ID As Integer, kind As String)
'This is the general information regarding the infrastructure that is on fire.
```

```
floors = levels
area = space
Location = ID
bType = kind

End Sub

Public Function StartEvent(pTotLoss As Single) As Integer
'This is the main function of the module.  It decides what kind of fire generation function to use
to mitigate
'the fire based on the specifications of the building.

Dim finish As Single
Dim first As Single, second As Single
Dim third As Single, i As Integer

'The first truck to be chosen is not necessarily the first to respond, due to prioritizing stations on
base
'over those on base.  Thus the incident model needs to recognize which truck arrives first.
first = WorksheetFunction.Max(hTrucks(1, 2), hTrucks(2, 2), hTrucks(3, 2))
third = WorksheetFunction.Min(hTrucks(1, 2), hTrucks(2, 2), hTrucks(3, 2))

For i = 1 To 3
   If hTrucks(i, 2) >= first And hTrucks(i, 2) <= third Then
      second = hTrucks(i, 2)
   End If
Next i

If floors = 2 And LCase(bType) = "residential" Then
   Call GenerateFire(first + 10, second + 10, third + 10, pTotLoss, finish)
   StartEvent = Int(finish) + 60
Else: 'All other buildings receive a hardcoded fire event.
   If hTrucks(1, 2) <= 5 Then
      pTotLoss = RV.RV("uniform", 0, 0.5)
   Else:
      pTotLoss = RV.RV("uniform", 0, 1)
   End If
   StartEvent = 90
End If

End Function

Private Sub GenerateFire(Time1 As Integer, Time2 As Integer, Time3 As Integer, TotalLoss As
Single, finish As Single)

'as a procedure this should get a time as the response time and calculate the loss rate
```

```
'and loss percentage upto that time. From that point, depending on mitigation remaining
'loss should be calculated in the main body of the program
Dim MitigationRate As Single
   MitigationRate = 0.004      'mitigation rate is assumed to be linear
   'each truck has that mitigation rate which will add up if there are multiple trucks
'***************Reading the name of the data sheet*********
Dim WorkingSheet As String
   WorkingSheet = "Simulation Scenarios"       'This should be changed to the corresponding
sheet name
   Worksheets(WorkingSheet).Select
'*************************************************************
Dim LossRate As Single
Dim i As Integer, j As Integer, SpreadParamCount As Integer
Dim t As Integer
   t = Time1
Dim LossRateT As Single, TotalLossT As Single
'*************Read in the parameters from the Sheet*************
Dim Param As Variant, SpreadParam As Variant
Dim SpreadScenCount As Integer
Dim ParamRange As Range

Set ParamRange = Range(Cells(3, 7), Cells(12, 13))
   SpreadParam = ParamRange.Value
   SpreadParamCount = Sheets(WorkingSheet).Range("G2").CurrentRegion.Rows.Count - 1

Dim PSpreadRTRG As Single, PSpreadFTFG As Single, PSpreadRTRU As Single,
PSpreadFTFU As Single, PGround As Single
   PGround = Sheets(WorkingSheet).Range("E8")
   PSpreadRTRG = Sheets(WorkingSheet).Range("E11")
   PSpreadFTFG = Sheets(WorkingSheet).Range("E12")
   PSpreadRTRU = Sheets(WorkingSheet).Range("E15")
   PSpreadFTFU = Sheets(WorkingSheet).Range("E16")

'    Debug.Print PGround
'    Debug.Print PSpreadRTRG
'    Debug.Print PSpreadFTFG
'    Debug.Print PSpreadRTRU
'    Debug.Print PSpreadFTFU

'*****Generate Fire*************
Dim Alpha As Single, Beta As Single, Damage As Single
Dim Origin As Single, Spread As Single
Dim FireType As Integer
Origin = Rnd
```

```
For i = 1 To SpreadParamCount - 1
   If Origin >= SpreadParam(i, 4) And Origin < SpreadParam(i + 1, 4) Then
      Debug.Print "Fire originated in", SpreadParam(i, 1)
      Debug.Print "Fire Contained in", SpreadParam(i, 2)
      FireType = i
      Damage = SpreadParam(i, 5)
      Alpha = SpreadParam(i, 6)
      Beta = SpreadParam(i, 7)
      Debug.Print "Fire type is", FireType
   End If
Next i

If Origin >= SpreadParam(SpreadParamCount, 4) Then
   Debug.Print "Fire originated in", SpreadParam(SpreadParamCount, 1)
   Debug.Print "Fire Contained in", SpreadParam(SpreadParamCount, 2)
   FireType = SpreadParamCount
   Damage = SpreadParam(SpreadParamCount, 5)
   Alpha = SpreadParam(SpreadParamCount, 6)
   Beta = SpreadParam(SpreadParamCount, 7)
   Debug.Print "Fire type is", FireType
End If

Spread = Rnd
Alpha = RV.RV("Gamma", Alpha / 0.02, 0.02)
Beta = RV.RV("Gamma", Beta / 0.15, 0.15)



   LossRateT = Damage * (Application.WorksheetFunction.Weibull(((t - 1) / 1.5), Alpha, Beta,
False))
   TotalLossT = Damage * (Application.WorksheetFunction.Weibull(((t - 1) / 1.5), Alpha, Beta,
True))

   LossRate = LossRateT
   TotalLoss = TotalLossT

Dim Flag As Boolean
   Flag = False
Dim LossRateT2 As Single, TotalLossT2 As Single
Dim LossRateT3 As Single, TotalLossT3 As Single
Dim FinishTime1 As Single, FinishTime2 As Single, FinishTime3 As Single ', Finish As Single

   FinishTime1 = (LossRateT / MitigationRate) + Time1
   finish = FinishTime1
   TotalLoss = TotalLossT + (((FinishTime1 - Time1) * 0.5 * LossRateT) * (2 / 3))
If Time2 = 0 Then
```

```
    Debug.Print "Finish Time with 1 truck at", finish
    Debug.Print "At time", t, "Loss rate is", LossRateT
    Debug.Print "At time", t, "Total Loss is", TotalLossT
    Debug.Print "At time", finish, "Total Loss is", TotalLoss

ElseIf (Time2 > 0 And Time2 < FinishTime1) Then
    LossRateT2 = LossRateT - (MitigationRate * (Time2 - t))
    TotalLossT2 = TotalLossT + (((Time2 - t) * 0.5 * (LossRateT + LossRateT2)) * (2 / 3))
    FinishTime2 = (LossRateT2 / (2 * MitigationRate)) + Time2
    TotalLoss = TotalLossT2 + (((FinishTime2 - Time2) * 0.5 * LossRateT2) * (2 / 3))

    If (Time3 > 0 And FinishTime2 > Time3) Then
        LossRateT3 = LossRateT2 - (2 * MitigationRate * (Time3 - Time2))
        TotalLossT3 = TotalLossT2 + (((Time3 - Time2) * 0.5 * (LossRateT2 + LossRateT3)) * (2
/ 3))
        FinishTime3 = (LossRateT3 / (3 * MitigationRate)) + Time3
        TotalLoss = TotalLossT3 + (((FinishTime3 - Time3) * 0.5 * LossRateT3) * (2 / 3))
        finish = FinishTime3
        Debug.Print "Finish Time with 3 truck at", finish
        'Debug.Print "At time", t, "Total Loss is", TotalLossT
        Debug.Print "At time", finish, "Total Loss is", TotalLoss

    Else
        finish = FinishTime2
        Debug.Print "Finish Time with 2 truck at", finish
        'Debug.Print "At time", t, "Total Loss is", TotalLossT
        Debug.Print "At time", finish, "Total Loss is", TotalLoss

    End If

End If


'*************************************************
'check simultaneous arrivals of trucks in the code
'*************************************************
If TotalLoss > Damage Then      'if the mitigation line went over the loss
    TotalLoss = Damage          'put the total intended loss (damage) as the total loss
End If

'Debug.Print "At time", t, "Loss rate is", LossRateT
'Debug.Print "At time", t, "Total Loss is", TotalLossT
'Debug.Print Finish
End Sub
```

FireStation

```
Option Explicit
'This module stores information regarding fire stations and their trucks.  It keeps track of where each truck
'is at all times.

Private Type trucks
   status As String
   Location As Long
   return As Single
End Type

Private crew As Integer
Private MaxHoursOn As Single
Private name As String
Public StationType As String
Private PumperTruck() As trucks
Private ChiefTruck() As trucks
Private HazMatTruck() As trucks
Private EMS() As trucks
Private TankerTruck() As trucks
Private probMaint As Single
Private MaintTime As Integer

Public Sub MaintInfo(prMaint As Single, MaintTm As Integer)
'Trucks are unavailable some percentage of the time for maintenance.  This is controlled by the parameters on
'the Header tab.

probMaint = prMaint
MaintTime = MaintTm

End Sub

Public Sub DefineStation(crew As Integer, MilCiv As String, nm As String)

crew = crew
StationType = MilCiv
name = nm

End Sub

Public Sub AddTruck(truck As String)
'When reading in the input data, each type of truck is placed in its own array to be managed and its status
'is set to rest, meaning that it is in the station and ready to go.
```

```
Select Case truck
   Case "Pumper"

      ReDim Preserve PumperTruck(0 To UBound(PumperTruck, 1) + 1)
      PumperTruck(UBound(PumperTruck, 1)).status = "Rest"

   Case "Command"

      ReDim Preserve ChiefTruck(0 To UBound(ChiefTruck, 1) + 1)
      ChiefTruck(UBound(ChiefTruck, 1)).status = "Rest"

   Case "HazMat"

      ReDim Preserve HazMatTruck(0 To UBound(HazMatTruck, 1) + 1)
      HazMatTruck(UBound(HazMatTruck, 1)).status = "Rest"

   Case "EMS"

      ReDim Preserve EMS(0 To UBound(EMS, 1) + 1)
      EMS(UBound(EMS, 1)).status = "Rest"

   Case "Tanker"

      ReDim Preserve TankerTruck(0 To UBound(TankerTruck, 1) + 1)
      TankerTruck(UBound(TankerTruck, 1)).status = "Rest"

End Select


End Sub

Public Function SendTruck(loc As Integer, truck As String, Time As Single, RV As RVGen) As
Integer
'This function is the interface for the simulation to get a truck for an incident.

Select Case truck
   Case "Pumper"

      SendTruck = PickTruck(PumperTruck, loc, Time, RV)

   Case "Command"

      SendTruck = PickTruck(ChiefTruck, loc, Time, RV)

   Case "HazMat"
```

```
        SendTruck = PickTruck(HazMatTruck, loc, Time, RV)

    Case "EMS"

        SendTruck = PickTruck(EMS, loc, Time, RV)

    Case "Tanker"

        SendTruck = PickTruck(TankerTruck, loc, Time, RV)

End Select

End Function

Private Function PickTruck(tManager() As trucks, loc As Integer, Time As Single, RV As
RVGen) As Integer
'This function sorts through a truck array to find the best truck to send to an incident.

Dim i As Integer, U As Single

PickTruck = 0

For i = 1 To UBound(tManager, 1)
    If tManager(i).status = "Rest" Then 'We only want to use trucks that are in the station.
        U = RV.RV("uniform", 0, 1) 'At times, a truck in the station is being maintenanced and is
unavailable.
        If U < probMaint Then
            tManager(i).status = "Busy"
            tManager(i).return = Time + MaintTime
        Else: 'If the truck is ready to go, change its status and location to reflect that it has left the
station.
            PickTruck = i
            tManager(i).status = "Busy"
            tManager(i).Location = loc
            Exit For
        End If
    End If
Next i

'Truck statuses are not changed at any particular time, but if the return time is less than the
current time, the
'truck is at the station and can be used.  There is no need to change the status, because it is busy
again.
If PickTruck = 0 Then
    For i = 1 To UBound(tManager, 1)
```

```
      If tManager(i).return < Time Then
         U = RV.RV("uniform", 0, 1) 'Again, the truck may be under maintenance and
unavailable.
         If U < probMaint Then
            tManager(i).return = Time + MaintTime
         Else: 'If the truck is ready to go, change its status and location to reflect that it has left the
station.
            PickTruck = i
            tManager(i).Location = loc
            Exit For
         End If
      End If
   Next i
End If

End Function

Public Sub ReturnTime(ID As Integer, truck As String, Time As Single)
'Here the simulation can tell the station when their trucks will return.  The station also notes that
the truck is busy.

Select Case truck
   Case "Pumper"

      PumperTruck(ID).return = Time
      PumperTruck(ID).status = "Busy"

   Case "Command"

      ChiefTruck(ID).return = Time
      ChiefTruck(ID).status = "Busy"

   Case "HazMat"

      HazMatTruck(ID).return = Time
      HazMatTruck(ID).status = "Busy"

   Case "EMS"

      EMS(ID).return = Time
      EMS(ID).status = "Busy"

   Case "Tanker"

      TankerTruck(ID).return = Time
      TankerTruck(ID).status = "Busy"
```

```
End Select

End Sub

Public Function BestResponse(truck As String, ID As Integer) As Single
'If no trucks are in the station, the station will look for the truck with the earliest return time.

Dim i As Integer

ID = 1

Select Case truck
   Case "Pumper"

      For i = 1 To UBound(PumperTruck, 1)
         If PumperTruck(i).return < PumperTruck(ID).return Then
            ID = i
            BestResponse = PumperTruck(ID).return
         End If
      Next i

   Case "Command"

      For i = 1 To UBound(ChiefTruck, 1)
         If ChiefTruck(i).return < ChiefTruck(ID).return Then
            ID = i
            BestResponse = ChiefTruck(ID).return
         End If
      Next i

   Case "HazMat"

      For i = 1 To UBound(HazMatTruck, 1)
         If HazMatTruck(i).return < HazMatTruck(ID).return Then
            ID = i
            BestResponse = HazMatTruck(ID).return
         End If
      Next i

   Case "EMS"

      For i = 1 To UBound(EMS, 1)
         If EMS(i).return < EMS(ID).return Then
            ID = i
            BestResponse = EMS(ID).return
```

```
            End If
        Next i

    Case "Tanker"

        For i = 1 To UBound(TankerTruck, 1)
            If TankerTruck(i).return < TankerTruck(ID).return Then
                ID = i
                BestResponse = TankerTruck(ID).return
            End If
        Next i

End Select

End Function

Public Sub ResetTrucks()
'At the beginning of a new replication, each truck will be set back in its station and return to rest
status.

Dim i As Integer

For i = 1 To UBound(PumperTruck, 1)
    PumperTruck(i).return = 0
    PumperTruck(i).status = "Rest"
Next i

For i = 1 To UBound(ChiefTruck, 1)
    ChiefTruck(i).return = 0
    ChiefTruck(i).status = "Rest"
Next i

For i = 1 To UBound(HazMatTruck, 1)
    HazMatTruck(i).return = 0
    HazMatTruck(i).status = "Rest"
Next i

For i = 1 To UBound(EMS, 1)
    EMS(i).return = 0
    EMS(i).status = "Rest"
Next i

For i = 1 To UBound(TankerTruck, 1)
    TankerTruck(i).return = 0
    TankerTruck(i).status = "Rest"
Next i
```

```
End Sub

Public Sub AddHours(hours As Single)
'This sub routine is not currently being used, but could be used to track the utilization of the
station.

MaxHoursOn = MaxHoursOn + hours

End Sub

Public Function Utilization(maxTime As Single) As Single
'This sub routine is not currently being used, but could be used to track the utilization of the
station.

Utilization = MaxHoursOn / maxTime

End Function

Private Sub Class_Initialize()

ReDim PumperTruck(0 To 0)
ReDim ChiefTruck(0 To 0)
ReDim HazMatTruck(0 To 0)
ReDim EMS(0 To 0)
ReDim TankerTruck(0 To 0)

End Sub
```

GoodIntent

```
Option Explicit

Public nTrucks As Integer
Private hTrucks As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0
```

```
End Sub
```

HazCond

```
Option Explicit

Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 2
nEMS = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

MalFalseCall

```
Option Explicit

Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
nEMS = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

Med_Treat

```
Option Explicit

Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
nEMS = 1
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

OREO

```
Option Explicit

Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 2
nEMS = 1
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

OtherFalseCall

```
Option Explicit
```

```
Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
nEMS = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

OtherFires

```
Option Explicit

Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 2
nEMS = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

OtherRescue

```
Option Explicit
```

```
Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
nEMS = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

RVGen

```
Option Explicit
'This class module is a compliation of Random Number generators created for several projects in the SEOR Masters
'Program by one of the members of the Fall 2012 team.  While not all distributions were used in the Fall 2012
'simulation, they are available for other incident models.  Each RV generator is coded from Law's Simulation
'Modeling & Analysis textbook.

Public Function RV(dist As String, p1 As Single, Optional p2 As Single, Optional p3 As Single) As Single
'This function will identify the RV generator to use and send the appropriate parameters to get the next
'random variable.

Select Case LCase(dist)
Case "normal"
   RV = NormalRV(p1, p2)
Case "beta"
   RV = BetaRV(p1, p2)
Case "triangular"
   RV = TriangularRV(p1, p2, p3)
Case "uniform"
   RV = UniformRV(p1, p2)
Case "lognormal"
   RV = LogNormalRV(p1, p2)
```

```
Case "exponential"
   RV = ExponentialRV(p1)
Case "gamma"
   RV = GammaRV(p1, p2)
End Select

End Function

Private Function NormalRV(m As Single, s As Single) As Double
'This formulation comes from Law 8.3.6 on pp. 453-454

Dim U1 As Single, U2 As Single
Dim V1 As Single, V2 As Single
Dim W As Single, y As Single
Dim x As Single

W = 2
Do While W > 1

   U1 = Rnd()
   U2 = Rnd()
   V1 = 2 * U1 - 1
   V2 = 2 * U2 - 1
   W = (V1 ^ 2) + (V2 ^ 2)

   If W <= 1 Then x = V1 * ((-2 * Log(W)) / W) ^ 0.5

Loop

NormalRV = m + s * x

End Function

Private Function GammaRV(Alpha As Single, Beta As Single) As Double
'This formulation comes from Law 8.3.4 on pp. 449-452

Dim U1 As Single, U2 As Single
Dim b As Single, x As Single
Dim y As Single

If Alpha < 1 Then

   Dim P As Single

   b = (Exp(1) + Alpha) / Exp(1)
   x = 100
```

```
    Do While x = 100
       U1 = Rnd()
       P = b * U1
       If P > 1 Then
          y = -Log((b - P) / Alpha)
          U2 = Rnd
          If U2 <= y ^ (Alpha - 1) Then x = y
       Else:
          y = P ^ (1 / Alpha)
          U2 = Rnd()
          If U2 <= Exp(-y) Then x = y
       End If
    Loop

Else:

    Dim a As Single, z As Single
    Dim v As Single, W As Single
    Dim q As Single, Th As Single
    Dim d As Single

    a = 1 / (2 * Alpha - 1) ^ 0.5
    b = Alpha - Log(4)
    q = Alpha + 1 / a
    Th = 4.5
    d = 1 + Log(Th)

    x = 100
    Do While x = 100
       U1 = Rnd()
       U2 = Rnd()
       v = a * Log(U1 / (1 - U1))
       y = Alpha * Exp(v)
       z = (U1 ^ 2) * U2
       W = b + q * v - y
       If W + d - Th * z >= 0 Then
          x = y
       Else:
          If W >= Log(z) Then x = y
       End If
    Loop
End If

    GammaRV = Beta * x
```

```
End Function

Private Function BetaRV(a1 As Single, a2 As Single) As Single
'This formulation comes from Law 8.3.8 on p. 455

Dim Y1 As Single, Y2 As Single

Y1 = GammaRV(a1, 1)
Y2 = GammaRV(a2, 1)

BetaRV = Y1 / (Y1 + Y2)

End Function

Private Function TriangularRV(a As Single, m As Single, b As Single) As Single
'This formulation comes from Law 8.3.15 on pp. 457-458

Dim U As Single, x As Single

U = Rnd()

If U <= (m - a) / (b - a) Then
   x = a + (U * (b - a) * (m - a)) ^ 0.5
Else:
   x = b - ((1 - U) * (b - a) * (b - m)) ^ 0.5
End If

TriangularRV = x

End Function

Private Function UniformRV(a As Single, b As Single) As Single
'This formulation comes from Law 8.3.1 on p. 448

Dim U As Single

U = Rnd()
UniformRV = a + (b - a) * U

End Function

Private Function LogNormalRV(m As Single, v As Single) As Single
'This formulation comes from Law 8.3.7 on p. 454

Dim x As Single
Dim mm As Single, s As Single
```

```
mm = Log((m ^ 2 / (m ^ 2 + v) ^ 0.5))
s = (Log(1 + (v / m ^ 2))) ^ 0.5

x = Exp(NormalRV(mm, s))

LogNormalRV = x

End Function

Private Function ExponentialRV(l As Single) As Single
'This formulation comes from Law 8.3.2 on p. 448

Dim U As Single, x As Single

U = Rnd()
x = -l * Log(U)

ExponentialRV = x

End Function
```

ServiceCall

```
Option Explicit

Public nTrucks As Integer
Private hTrucks As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

SpecialIncident

```
Option Explicit
```

```
Public nTrucks As Integer
Private hTrucks As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

SW_ND

```
Option Explicit

Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks As Integer
Private hEMS As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 2
nEMS = 2
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

Timeline

```
Option Explicit
'This module manages the Event List for the simulation.  The Fall 2012 team built the event list at the
'beginning of each replication, but there should be no difficulty with adding additional events once the
'replication has begun.
```

```
Private Timeline() As Variant
Private current As Single 'This is the index for Timeline, not a time.

Private Sub Class_Initialize()

ReDim Timeline(2, 1)
current = 0

End Sub

Public Sub AddEvent(Time As Single, eventName As String)
'This sub routine is used to add an event given a clock time and event name.  The event is hung on the timeline
'using an incertion method.

Dim i As Integer

ReDim Preserve Timeline(2, UBound(Timeline, 2) + 1) 'Add one more space to the timeline.

For i = UBound(Timeline, 2) - 1 To 1 Step -1 'Start at the latest time.
   If Timeline(1, i) > Time Then 'If the new time is earlier, move the old time towards the end of the timeline.
      Timeline(1, i + 1) = Timeline(1, i)
      Timeline(2, i + 1) = Timeline(2, i)
   Else: 'If the new time is later then, place it in the index behind the old time it is compared to.
      Timeline(1, i + 1) = Time
      Timeline(2, i + 1) = eventName
      Exit For
   End If
Next i

End Sub

Public Function NextEvent() As String
'This function moves to the next event and tells the simulation the name of the event.

current = current + 1
NextEvent = Timeline(2, current)

End Function

Public Function Time() As Single
'This function tells the simulation the clock time of the current event.

Time = Timeline(1, current)
```

```
End Function

Public Function NumOfEvents() As Integer
'This function counts how many events are on the whole timeline.  Because of the way it is coded, there will be two
'blank entries at the beginning of the timeline.  Therefore, the number of events is actually two less than what
'this function will say.

NumOfEvents = UBound(Timeline, 2)

End Function
```

UnkIncident

```
Option Explicit

Public nTrucks As Integer
Private hTrucks As Integer
Public PropLoss As Single
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()

nTrucks = 1
PropLoss = 0
CivHumLoss = 0
MilHumLoss = 0

End Sub
```

VehicleFire

```
Option Explicit

Public nTrucks As Integer
Public nEMS As Integer
Private hTrucks() As Integer
Private hEMS() As Integer
Public CivHumLoss As Single
Public MilHumLoss As Single

Private Sub Class_Initialize()
```

```
ReDim hTrucks(1 To 2, 1 To 3)
ReDim hEMS(1 To 2)
nTrucks = 2
nEMS = 2
CivHumLoss = 0
MilHumLoss = 0

End Sub

Public Sub AddTruck(truckID As Integer, responseTime As Integer, stationID As Integer)
'Each truck is added individually as the main simulation determines the next best truck to send.

Dim i As Integer

For i = 1 To UBound(hTrucks, 1)
   If hTrucks(i, 1) = 0 Then
      hTrucks(i, 1) = truckID
      hTrucks(i, 2) = responseTime
      hTrucks(i, 3) = stationID
      Exit For
   End If
Next i

nTrucks = nTrucks - 1

End Sub

Public Function StartEvent(pTotLoss As Single) As Integer
'This is the main function of the module.  Since no team has built a model for this yet, the event
is hard coded.

pTotLoss = 0.75
StartEvent = 90

End Function
```