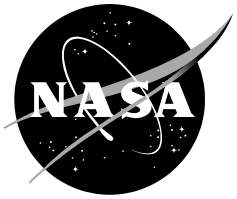
NASA/CR—2011–000000



Disaster Response Effectiveness of a Fleet of Civil Tiltrotor Aircraft Operating in the Next Generation Air Transportation System (NextGen)

Terry L. Gibson
Bell Helicopter Textron Inc., Hurst, Texas

Michael J. Jagielski
Bell Helicopter Textron Inc., Hurst, Texas

John R. Barber
Bell Helicopter Textron Inc., Hurst, Texas

December 2011

NASA STI Program ... in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Report Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing help desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at *http://www.sti.nasa.gov*

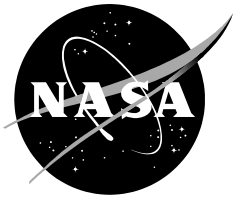
- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA STI Help Desk at 443-757-5803

- Phone the NASA STI Help Desk at 443-757-5802

- Write to:
  NASA STI Help Desk
  NASA Center for AeroSpace Information
  7115 Standard Drive
  Hanover, MD 21076-1320

NASA/CR—2011–000000



Modeling High-Speed Civil Tiltrotor Transports in the Next Generation Airspace

Task 7 - Disaster Response Effectiveness Analysis

Terry L. Gibson
Bell Helicopter Textron Inc., Hurst, Texas

Michael J. Jagielski
Bell Helicopter Textron Inc., Hurst, Texas

John R. Barber
Bell Helicopter Textron Inc., Hurst, Texas

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, CA 94035-1000

December 2011

Acknowledgments

The authors would like to acknowledge that this research and development was accomplished with the support and guidance of Mr. Larry Young of the National Aeronautics and Space Administration (NASA) Ames Research Center and Mr. William Chung of the Science Applications International Corporation (SAIC).

Available from:

NASA Center for AeroSpace Information
7115 Standard Drive
Hanover, MD 21076-1320
443-757-5802

Table of Contents

List of Figures

List of Tables

Introduction

Government and civilian agencies rely on vertical take-off and landing (VTOL) aircraft to perform a variety of missions to assist and protect the public during emergencies resulting from man-made or natural disasters. Helicopters have proven to be critical to the success of post-disaster operations due to their ability to hover, land in physically restricted areas, and enter those areas inaccessible to land or water vehicles. Understanding how advances in rotorcraft technology can be applied to these operations will enhance the effectiveness of emergency services.

This report presents the results of analyses investigating the ability of a fleet of civil tiltrotor (CTR) aircraft, operating in the Next-Generation Air Transportation System (NextGen) (Refs. 1-4) and augmenting a conventional rotorcraft (CRC) fleet, to significantly improve vertical lift effectiveness in post-disaster operations such as those performed in the aftermath of hurricane Katrina. Using a stochastic queuing model to simulate the emergency response events associated with a “Katrina-like” disaster, the effectiveness of rotorcraft fleets (consisting of a mix of CRC and conceptual CTR aircraft) was analyzed performing transport, search and rescue (SAR) and evacuation missions. The study used historical data from the Katrina disaster to provide a baseline for simulated aircraft and mission event characteristics, types and quantity.

The study focused on three CTR aircraft sizes consisting of 10-, 30- and 120- passenger capacities (or equivalent cargo capacity) and an operational strategy afforded by the CTR’s increased speed and range relative to CRC. Conventional rotorcraft, due to their limited range, typically transport evacuees to a local site for staging and eventual evacuation to a remote, safe location by fixed-wing aircraft. With a fleet augmented by CTR aircraft, ambulatory evacuees can be transported immediately to a safe, remote evacuation destination eliminating the intermediate stop at a local staging site. The study showed that this strategy and CTR aircraft performance, afforded by the use of tiltrotor technology, increases the pace at which evacuation and rescue are performed saving lives and property, while enhancing post-disaster mission organization.

Analysis Details

A comparative analysis was performed using a software tool to model post-disaster operations of rotorcraft and capturing specific measures of effectiveness for evaluation. The tool is capable of evaluating many man-made or natural disaster scenarios; however, given the availability of reference data on the Hurricane Katrina disaster, the analysis focused on a hurricane’s post-disaster operations employing helicopters. This study did not attempt to model the detailed events of these operations, but rather used the data to provide a basis for analyzing rotorcraft fleet configurations and potential operational strategies afforded by the fleet within in a similar disaster scenario.

Disaster Scenario

In August 2005, Hurricane Katrina made landfall in southeastern Louisiana with devastating consequences (Refs. 6-20). The ensuing destruction of the region was due primarily to the storm surge. The hardest hit area was the city of New Orleans. The storm surge caused the catastrophic failure of the city’s levee system and flooding of eighty percent of the city. Those of the city population that did not heed the mandatory, pre-storm evacuation instructions died in the floodwaters or were stranded in their homes in need of immediate rescue due to flooding, fires and uninhabitable conditions.

Local, state and federal authorities converged on the area with food, water, and equipment. Nearly 400 helicopters were called upon to transport people and supplies, evacuate the injured and ambulatory persons, perform utility operations, and rescue those stranded by flooding. As shown in Table 1, these helicopters consisted of many types and came from various organizations.

Table 1. Helicopters Used During Post-Katrina Operations

Organization	Class	Quantity	Type
Coast Guard			
	Light	30	HH-65
	Medium	19	HH-60
	Heavy	0	
National Guard			
	Light	9	OH-58
	Medium	80	UN-1N, UH-60
	Heavy	40	CH-47
Army			
	Light	6	OH-58
	Medium	35	UN-1N, UH-60
	Heavy	11	CH-47
Air Force			
	Light	0	
	Medium	34	UN-1N, HH-60
	Heavy	5	MH-53
Navy/Marines			
	Light	0	
	Medium	65	MH-60, SH-3, CH-53, CH-46
	Heavy	12	MH-53E
Other			
	Light	50	S-76B, S-76C, S-70, Bell 206L4, Bell 212, Bell 407, Bell 430, EC 120, AS 350B3
	Medium	0	
	Heavy	0	
Totals			
	Light	***95***	
	Medium	***233***	
	Heavy	***68***	
	Total	***396***	

Over eleven days, helicopter operators performed missions continuously, missions attributed with saving numerous lives and preventing the further loss of property. Helicopters transported 3,790 tons of cargo (consisting of food, water, sandbags and other equipment) and over 53,000 people. Table 2 presents the number of people transported for each type of evacuation and rescue mission.

Table 2. Number of People Evacuated Per Mission Type

Mission	Number of People
Ambulatory Evacuation	18,078
Non-Ambulatory Evacuation (MedEvac)	10,626
Search and Rescue	24,367
Total	***53,071***

Operational Strategy

An operational strategy defines the proposed activities of rotorcraft performing missions during post-disaster operations. This strategy defines, for each rotorcraft asset, the type of aircraft and how it is to be used during the execution of each mission type. Five basic scenario locations were identified. These locations define origination, intermediary, and destination points for aircraft during the course of a mission and are defined as follows:

- **Remote Base** – the origination site of rotorcraft outside the disaster area.
- **Remote Site** – the final destination for evacuees outside the disaster area. Refueling is available at this location.
- **Local Base** – the rotorcraft base within the disaster area. Refueling is available at this location.
- **Local Site** – the intermediate staging location for evacuees or cargo drop-off site within the disaster area. Fixed-wing aircraft base.
- **Event Site** – the location of an event within the disaster area (e.g. evacuation pick-up, rescue, cargo pick-up).

One common component of this strategy for all missions is to deploy rotorcraft assets to the disaster area from remote bases. This asset deployment to the mission area occurs once air traffic is allowed within the disaster area and post-disaster operations may begin. These rotorcrafts include those that may have been pre-positioned to safe locations outside the disaster area and other rotorcraft used by participating organizations and deployed from distant locations. After asset arrival at the local base within the disaster area, these rotorcrafts are assigned missions from a command and control center. The missions assigned include the following:

- Evacuation (Ambulatory)
- Medical Evacuation (Non-Ambulatory)
- Search and Rescue
- Cargo Transport

The operational strategy for each mission is defined in the following sections.

Evacuation

The objective of the evacuation mission is to transport evacuees (non-MedEvac) within the disaster area to a safe, remote site outside the disaster area. The command and control center assigns a rotorcraft to the mission that meets the mission requirements (e.g. passenger capacity, range, etc.). As shown in Figure 1, the assigned rotorcraft is launched from a local base to the location of the evacuation event or request. Upon arrival, evacuees are loaded on the rotorcraft for evacuation. Once loaded, the rotorcraft departs to an evacuee drop-off site which may be local or remote.

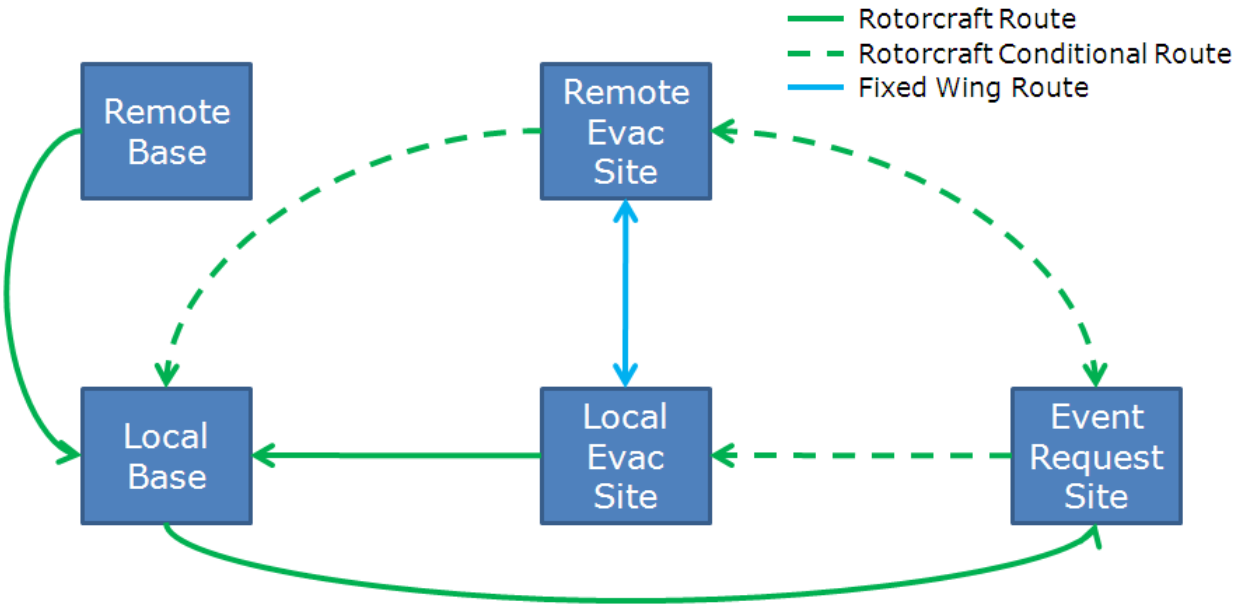


Figure 1. Evacuation Strategy

The local site is selected as the mission termination point if the rotorcraft does not meet the minimum passenger occupancy requirements or the rotorcraft cannot meet the range requirement for travel to the remote site. At the local site, fixed-wing aircraft are used to transport awaiting evacuees to the remote evacuation destination outside the disaster area. Once the rotorcraft has offloaded evacuees, the rotorcraft returns to the local base for refueling and the next mission assignment.

The remote site is selected if the rotorcraft meets its minimum passenger occupancy requirements and is capable of making the distance to the remote site. This evacuation strategy simplifies and speeds the evacuation process by eliminating the need to stop at the local site. Upon arrival at the remote site, the evacuees are offloaded and the rotorcraft awaits the next mission assignment. If no assignment is provided within a predefined time requirement, the rotorcraft returns to the local base, refueled, and awaits further mission instruction.

Medical Evacuation

The objective of the medical evacuation (MedEvac) mission is to transport non-ambulatory evacuees to a site where medical care is provided. This mission is assigned by the command and control center to any rotorcraft that meets the minimum mission requirements (e.g. litter capacity, range, etc.). As shown in Figure 2, the MedEvac aircraft is launched from a local base to the event request site (pick-up site) where evacuees are loaded. In order to provide medical attention to the evacuees as quickly as possible, evacuees are transported to the local site offering medical care. At this site, the evacuees are treated and stabilized prior to transfer by fixed-wing aircraft to a safe, remote site outside the disaster area. The rotorcraft returns to the local base, refueled, and awaits the next mission instruction.

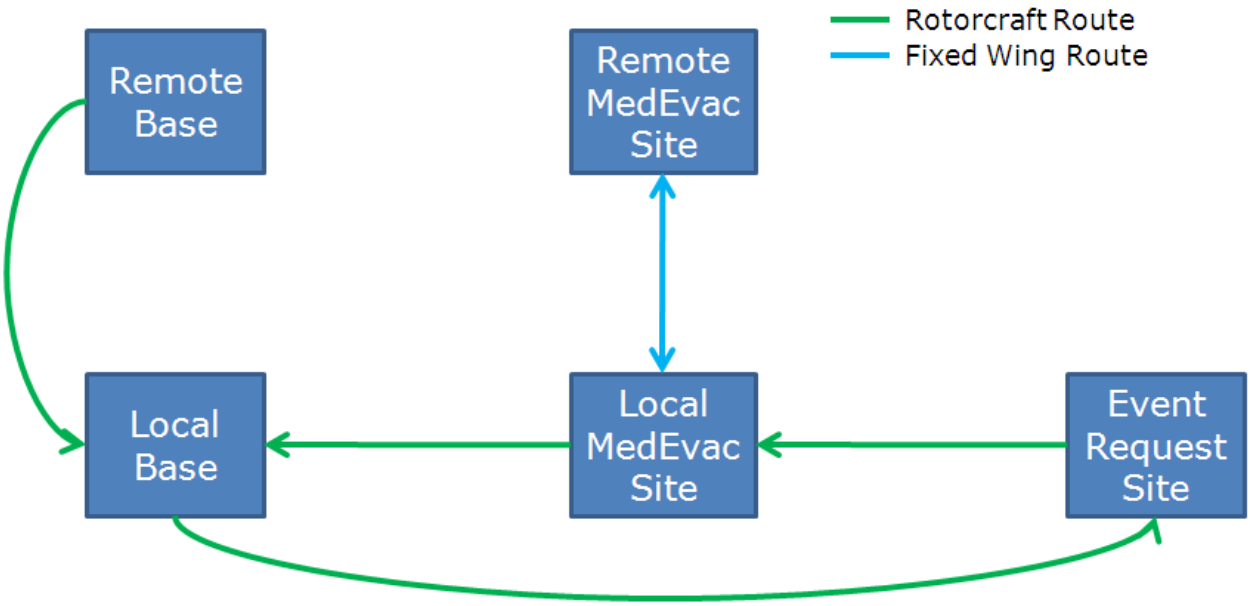


Figure 2. MedEvac Strategy

Search and Rescue

The objective of the search and rescue (SAR) mission is to search a predefined area within the disaster area for people who are stranded and in need of immediate rescue and evacuation. The command and control center assigns a rotorcraft to the mission if it meets the minimum mission requirements. The requirements include range to the search area and search area coverage. As shown in Figure 3, the rotorcraft is launched from a local base to the search area. Upon arrival at the search area, the rotorcraft commences searching the area for persons requiring rescue. If rescuees are located, they are picked up and transported to either a local or remote evacuation site. The site is selected based upon the rotorcraft's remaining fuel and range capability.

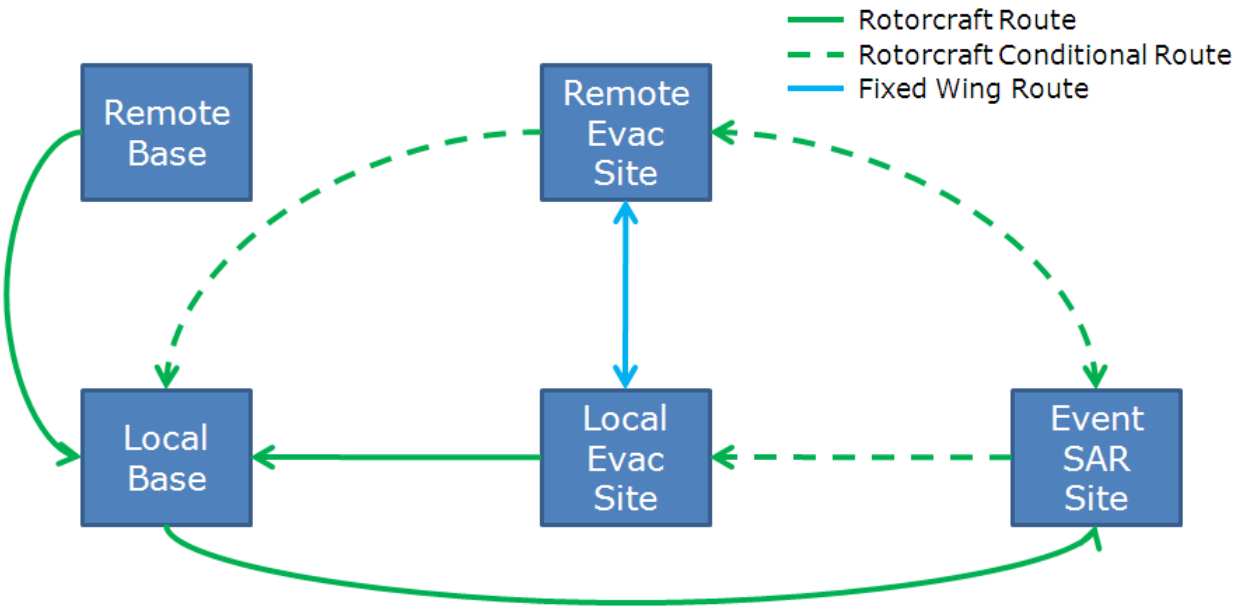


Figure 3. SAR Strategy

A local site is selected (default) if the range and fuel requirements for remote evacuation are not met. At the local site, rescuees are off-loaded and await transport to a safe, remote evacuation site by fixed-wing aircraft. The rotorcraft returns to the local base, refueled and is placed in a queue for the next mission assignment.

A remote site is selected if the range and fuel requirements are met. Once rescuees are transported to the remote site, they are offloaded and the rotorcraft is refueled. The rotorcraft remains at the remote site until the next assignment within a predefined assignment time requirement. If no assignment is made within the time requirement, the rotorcraft returns to the local base, is refueled and awaits the next assignment.

Cargo Transport

A cargo transport mission is performed to move supplies and equipment from a local pick-up site to a local drop-off site. The command and control center assigns a rotorcraft to the mission if it meets the minimum requirements of the mission that include mission range and cargo capacity (pounds). As shown in Figure 4, the rotorcraft is launched from a local base to the cargo pick-up site. At this site, the cargo is loaded and secured for transport to the local drop-off site. Once the cargo is off-loaded at the drop-off site, the rotorcraft returns to the local base for refueling and next mission assignment.

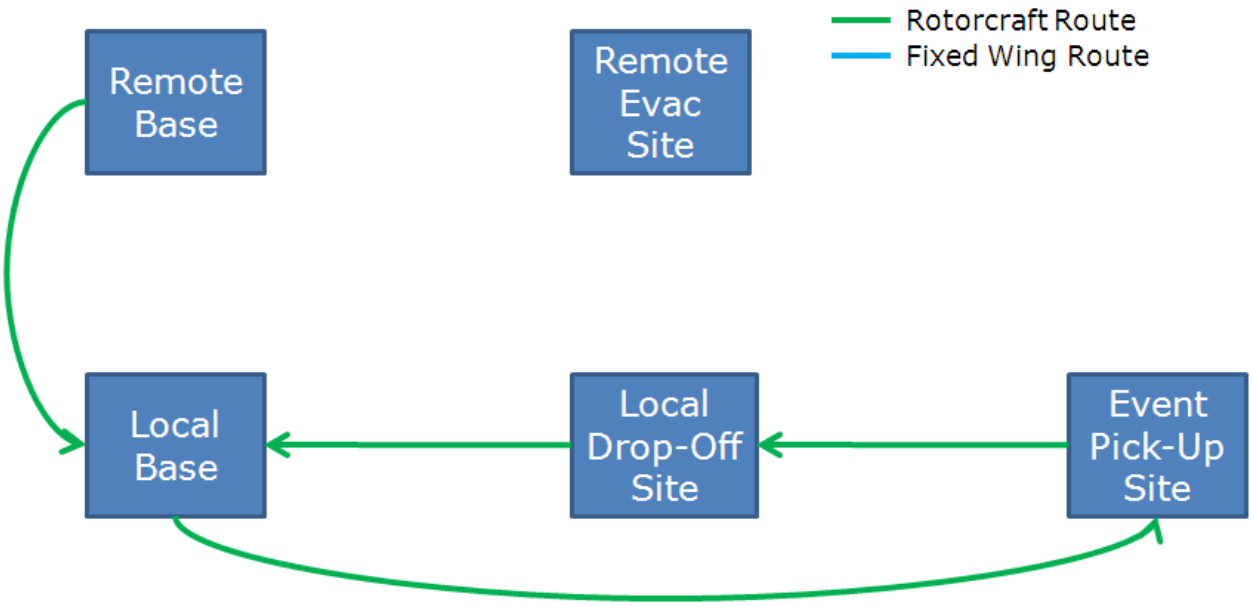


Figure 4. Cargo Transport Strategy

Modeling

The top-level view of the simulation model, shown in Figure 5, creates both vehicle and event objects which enter queues in the command and control function. All vehicles originate at a remote base outside the disaster area or “area of operation”. The area of operation will be defined as the area that contains local bases, the local site, and location of events. The command and control function, a specialized super-node, then merges vehicles and events into a single “mission” object to be processed. A determination will be made whether the mission continues on to the remote site or the local site. If the local site is the destination, evacuees enter a queue waiting to be transferred by fixed-wing aircraft to the remote site. Once the

evacuees arrive at the final destination, statistics are collected and measures of effectiveness are computed. A complete, detailed description of the simulation design and data requirements is provided in appendices at the end of this report. These appendices include:

- Appendix A - simulation design.
- Appendix B - global attributes defining simulation data.
- Appendix C – M^3S input file representing the simulation network in XML format.
- Appendix D - Java™ software plug-ins developed for modeling specific simulation logic.

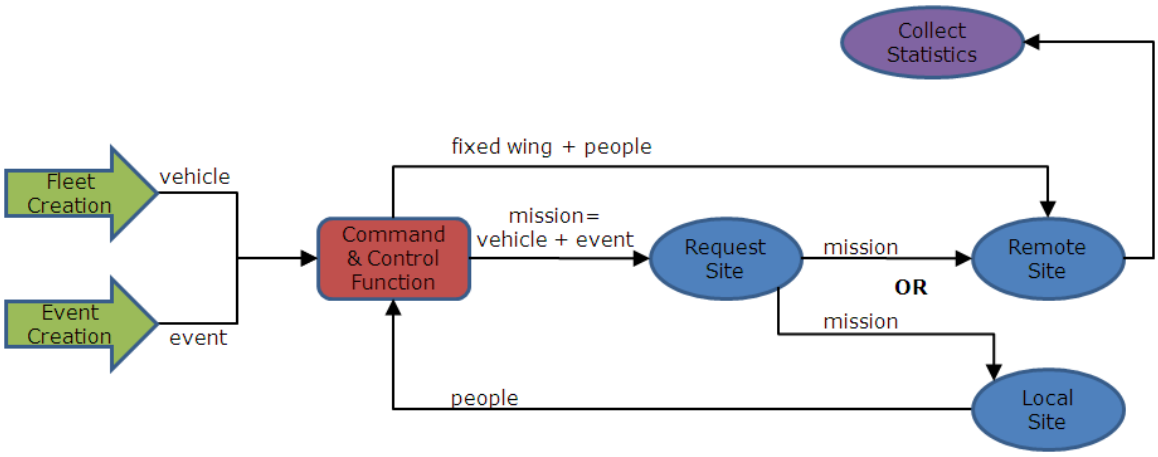


Figure 5. Top-Level Simulation Model

Analysis Software

The "Katrina-like" post-disaster events and helicopter operations were modeled using the Bell Helicopter proprietary model, Mathematical Monte-Carlo Modeling Simulation (M^3S), an implementation of the Simulation Language for Alternative Modeling (SLAM®) (Ref. 5). The probabilistic model provides an interactive, graphical means for modeling discrete and continuous processes through the use of nodes and activities. Entities, nodes and activities are the basic elements used to describe a process within the model. Examples of network components are shown in Figure 6. This figure presents nodes and activities required to create the evacuation component of the simulation network.

Entities represent units of traffic flowing through the process. Entities are aircraft and events. Attributes are assigned to each entity in the process. These attributes are numerical values carried along with each entity as it travels through the process and specify its characteristics. For example, each aircraft in the model is assigned capacity and performance characteristics. These characteristics influence how well each aircraft performs within the simulation.

Activities represent time delays associated with entity movement. For example, in the simulation a branch may represent the movement of a helicopter from one location to another. Also, activities may be conditional and require conditions to be met to allow entities to flow through the activity thus directing the flow of entities through the process. For example, the destination site of evacuees is selected based upon the aircraft's remaining fuel or range limitations.

The M^3S software provides an accurate means to simulate the aircraft, events and operational strategies used during disaster scenarios.

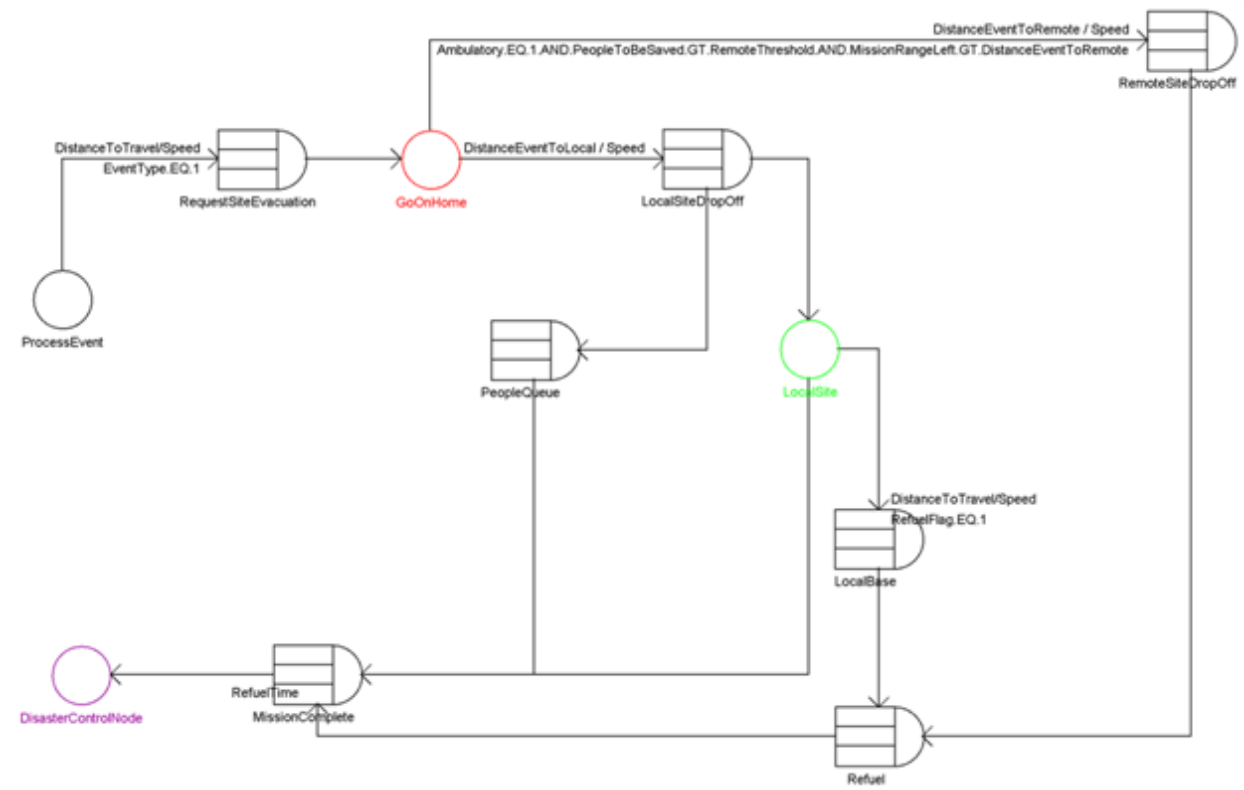


Figure 6. M^{3}S Evacuation Network

Simulation Probability Distributions

Probability distribution functions are utilized to model major events in the simulation. These functions provide a means to model the probability and density of the occurrence of a random variable. Within this analysis, the uniform probability distribution function was selected to describe random variables with equally spaced outcomes such as mission ranges and quantities per mission event. These random variables are listed in Table 3 along with their predefined minimum and maximum values. Because detailed helicopter sortie data was not available for sorties flown during Katrina post-disaster operations, the uniform distribution was considered the proper choice for describing the occurrence of these random variables. Other probability distribution functions describing these random variables could be substituted if data becomes available. Additionally, mission event creation utilized the uniform probability distribution function and is described later in this report.

Table 3. Uniform Probability Distribution Parameters

Parameter	Minimum	Maximum
Weight of Supplies to Transport	0.25 tons	10 tons
Number of People to Evacuate/Event	1 person	500 people
Number of People to Rescue/Event	1 person	10 people
Number of People to MedEvac/Event	1 person	100 people
Distance Between Remote Base and Local Base	300 nmi	1000 nmi
Distance Between Local (Base or Site) and Event	0 nmi	150 nmi
Distance Between Remote Site and Event	300 nmi	500 nmi
Distance Between Local Base and Local Site	1 nmi	50 nmi
Search and Rescue Range	0 nmi	50 nmi

Due to the use of random variables within the model, a number of model iterations or executions were performed for each run configuration to assure a stable, conclusive outcome convergence. Tests were performed to determine number of iterations required for solution convergence. It was determined that averaging each run's output over one-hundred (100) iterations satisfied the convergence criteria. It should also be noted that the same random number generator seed was used for each run configuration evaluation to promote replication.

Aircraft

The analysis used performance and capacity characteristics for seven aircraft types. These aircraft types include six rotorcraft types and one fixed-wing type. These aircraft types include:

- Light Conventional Rotorcraft (CRC-L)
- Medium Conventional Rotorcraft (CRC-M)
- Heavy Conventional Rotorcraft (CRC-H)
- Light Civil Tiltrotor (CTR-L)
- Medium Civil Tiltrotor (CTR-M)
- Heavy Civil Tiltrotor (CTR-H)
- Fixed-Wing (FW)

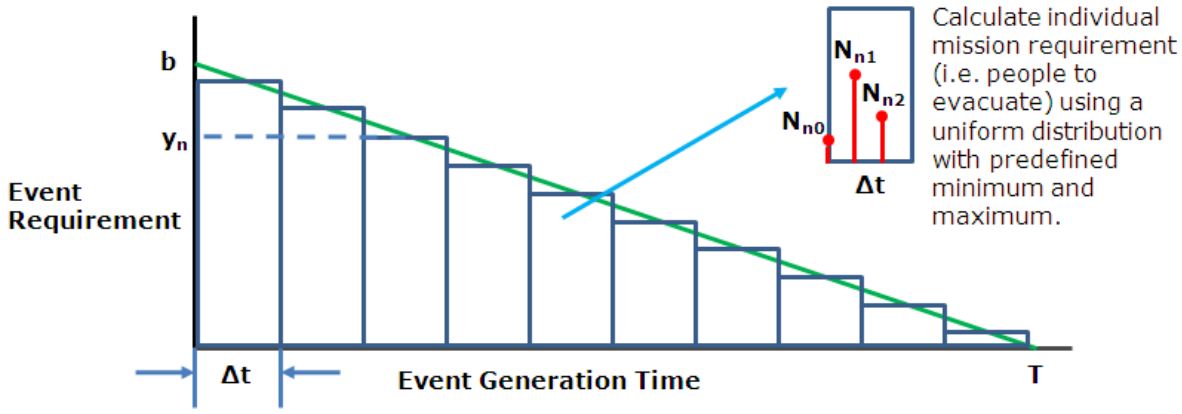
The CRC and CTR characteristics were used by the command and control function to assign available rotorcraft to missions that best met mission requirements. FW aircraft, of generic type, were only used to transport evacuees from the local evacuation site to a remote evacuation site. The performance and capacity characteristics of each aircraft type are provided in Table 4. The characteristics for each CRC modeled were obtained by averaging the data for each helicopter per class used during Katrina operations. CTR performance data was generated during a task performed earlier in the overarching NextGen study. A generic, commercial FW aircraft was modeled in this study. The FW aircraft characteristics missing in the table were not required for modeling.

Table 4. Rotorcraft Characteristics

Aircraft	Crew	Pax	Litters	Empty Weight (lbs)	MTOW (lbs)	Cruise Speed (kts)	Range (nmi)	Fuel (lbs)	Payload (lbs)
CRC-L	2	8	3	4,538	7,858	129	330	1,254	1,669
CRC-M	3	18	10	14,733	25,391	138	388	3,761	6,254
CRC-H	5	43	21	30,440	57,917	143	513	6,945	19,462
CTR-L	2	10	5	11,022	16,192	280	821	2,500	2,200
CTR-M	3	30	15	32,160	46,430	340	804	7,000	6,600
CTR-H	5	120	60	98,737	147,647	345	1,331	21,441	26,399
FW	4	175	-	-	-	400	2,000	-	-

Event Creation and Mission Requirements

An event creation function was used to determine the mission requirements for each mission type during the simulation. This analysis assumed that event creation would be the greatest at the beginning of the simulation, the time at the start of post-disaster operations, and would degrade over time until all evacuation, SAR and cargo transport operations were completed. Due to the scarcity of data to define the actual event distribution that occurred during Katrina post-disaster operations, the analysis assumed a linear degradation of event requirements as depicted in Figure 7. However, the event distribution can easily be replaced with another once a more descriptive representation is determined.



$A = (b \times T)/2; A$: Area (i.e. total number of people, total tons of cargo)

$b = (2 \times A)/T; b$: y intercept; T: event generation end time

$m = -b/T; m$: slope

$y_n = m \times [n \times \Delta t + \Delta t/2] + b; n = 0, T/\Delta t - 1; y$: event requirement at time $t; \Delta t$: event requirement time duration

$N_n = y_n \times \Delta t; N$: individual mission requirement for Δt

Figure 7. Event Creation

Once the total evacuation requirement that occurred over a simulation time duration is known, the number of people to evacuate during a specific simulation time interval can be calculated. For this analysis, a 6 hour time interval was used. Once the evacuation requirement for this time period is calculated, the individual mission requirements are determined for discrete times using a uniform probability distribution function to describe the random variable, number of people requiring evacuation per event. This function also requires that a minimum and maximum limit be defined per mission evacuation. A random draw of the number of evacuations per event is continued until the individual evacuation requirement is met for this time period with the requirements equally spaced over this period.

As individual event requirements are created, it is possible that the requirements may exceed the mission capability of a single rotorcraft thus requiring the creation and assignment of additional rotorcraft missions by the command and control function. In this case, the command and control function will divide the individual event requirement into events that can be serviced with multiple, available rotorcrafts. For example, if an individual mission requirement is created that requires the evacuation of 20 people and only 10 passenger rotorcrafts are available, the command and control function divides the event requirement into 2 missions with each requiring 1 rotorcraft with a 10 passenger capacity. This approach optimizes resource utilization to promote simulation efficiency.

In Figure 8 and Figure 9, the event creation requirements for the baseline scenario are presented using the event creation methodology defined previously. These requirements define the total number of missions, mission's rate of occurrence, and the total length of time in which missions are performed. With time as the enemy of human survival in disastrous conditions, the bulk of rotorcraft missions executed at the beginning of post-disaster operations are focused more heavily on SAR and MedEvac missions. This analysis assumes that requests for these services continue for 5.5 days at a very high rate of occurrence, while cargo transport missions

continuing for 8.25 days and general evacuation for 11 days at a lower rate of occurrence. These requirements are logical and based upon narrative describing Katrina events.

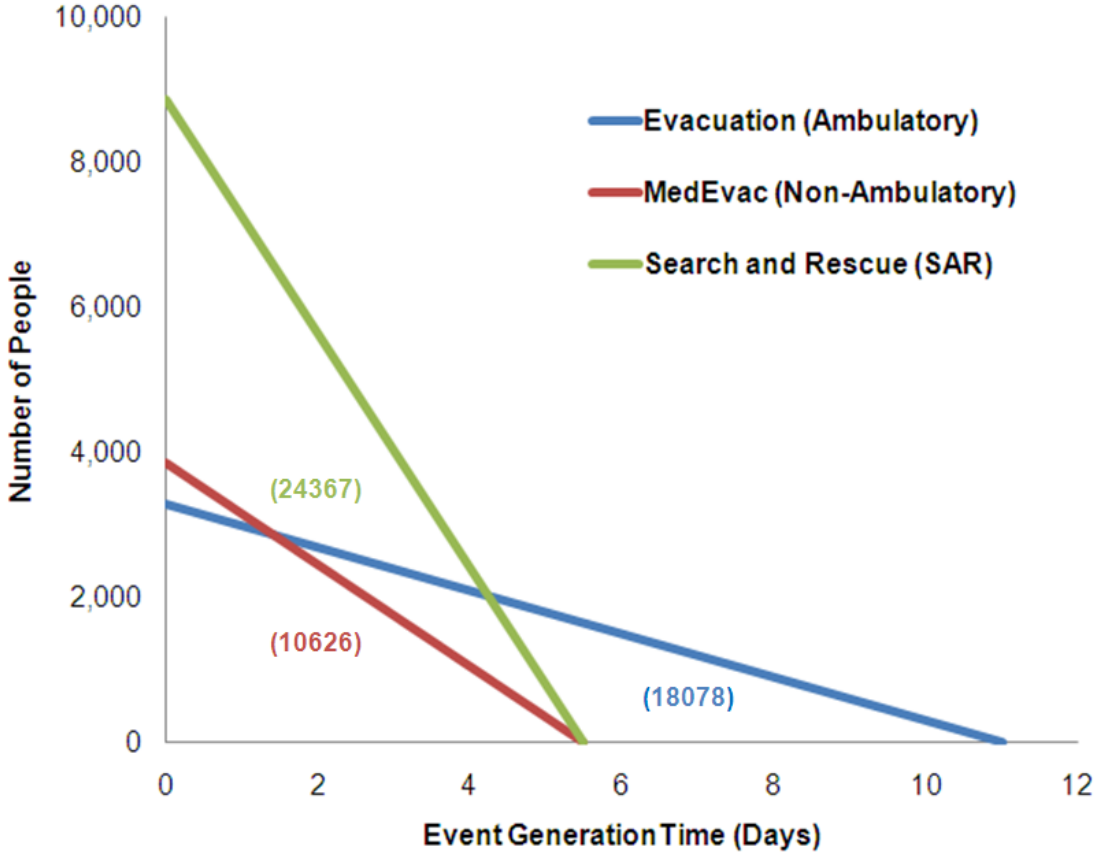


Figure 8. Evac, MedEvac, and SAR Event Creation

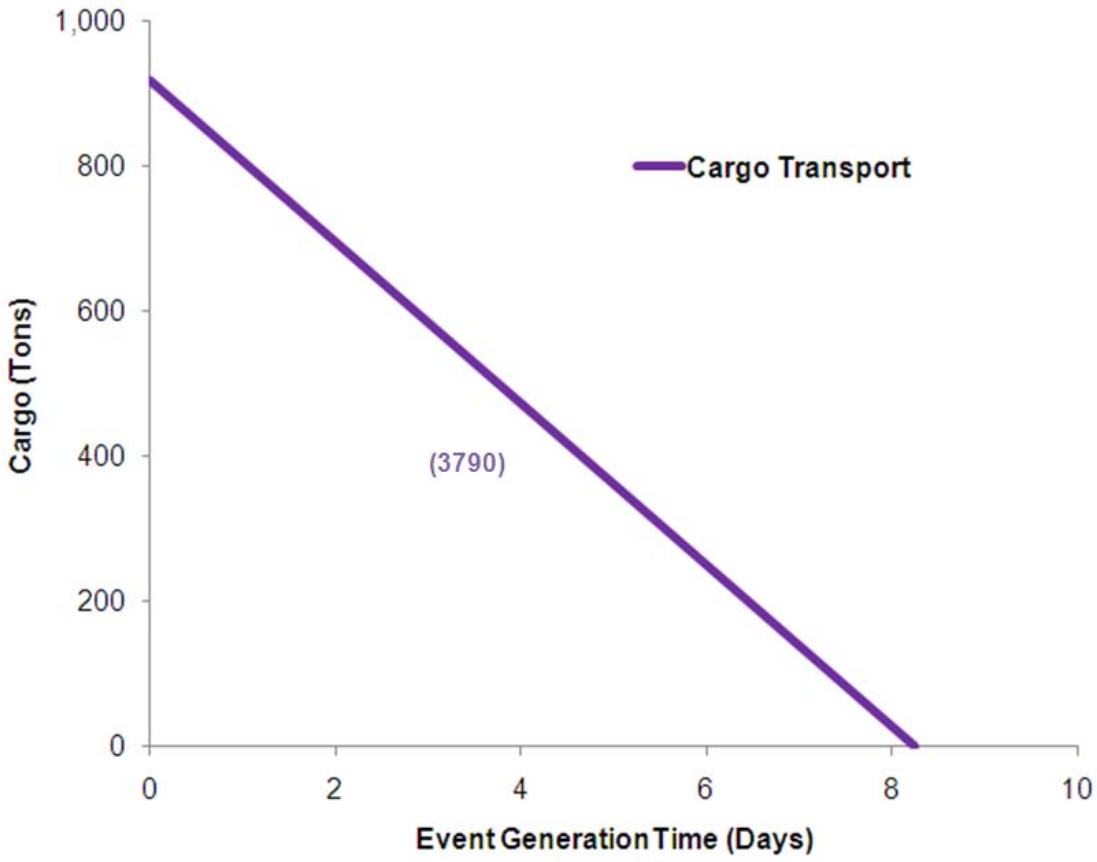


Figure 9. Cargo Transport Event Creation

Asset Mission Assignment

The command and control function assigns rotorcraft to specific missions that are generated from scenario events. The command and control function consists of the following lists and queues (Figure 10):

- **Vehicle List**: Contains the fleet mix of rotorcrafts.
- **Event Queue**: Contains the list of events (Evacuation, MedEvac, SAR, and Transport). These events are ordered first by event priority and next by time.
- **Fixed-Wing List**: Contains the list of fixed-wing air vehicles. Fixed-wing vehicles are only used to move/transfer people from a local site to a remote site.
- **Local People Queue**: Contains a list of people that were transported from an evacuation, MedEvac or SAR event but were not taken to a remote site. The order of the queue is FIFO.

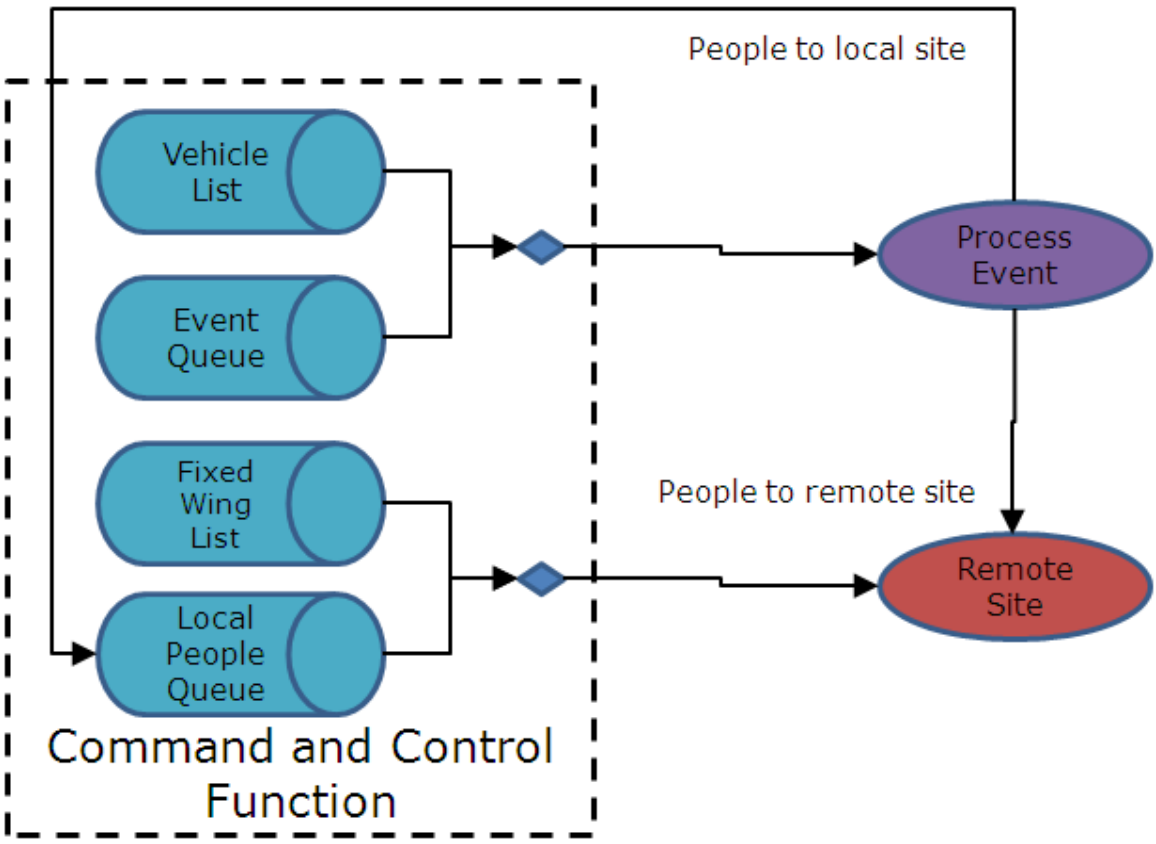


Figure 10. Command and Control Function

The command and control function performs two major tasks by merging a vehicle list and queue:

- **Missions**: The vehicle list and event queue are used to generate missions. As events and/or vehicles enter the command and control function, the function determines if any vehicles, currently in the list, satisfy the requirements for any of the events. If the event involves people (Evacuation or SAR), a determination is made on whether the people are transported to a local site or to a remote site. If the people are transported to a local site, then the people enter the local people queue.
- **People Transfer Event**: The fixed-wing list and local people queue are used to generate people transfer events. If the number of people in the queue reaches a certain threshold and at least one fixed-wing vehicle is available, then a fixed-wing vehicle is loaded to transfer the people to a remote site.

Each event generated (Evacuation, MedEvac, SAR, and Transport) may be too large for one air vehicle (e.g. evacuate 1,000 people). In general, the command and control function initially sorts all the currently available air vehicles by the amount of mission range (or fuel) remaining. When more than one rotorcraft is available to transport the remaining people (or supplies), the controller chooses the rotorcraft with the closest capacity (passenger or useful load) to perform the mission.

The remainder of this section discusses steps the command and control function takes when processing the various event types.

Search and Rescue (SAR)

1. The available rotorcraft list is reduced to a list of rotorcraft that satisfy the SAR threshold capacity, rotorcrafts with passenger capacity greater than this threshold are not considered for SAR missions. *Note: It is assumed that the distribution used to generate the event will not generate a number greater than this threshold.*
2. This reduced list is then sorted by time on station (TOS) remaining (or mission range left).
3. The rotorcraft with the most TOS remaining (or mission range left in our case) must have a mission range left greater than the SAR range threshold; otherwise no rotorcraft can currently perform this mission.
4. A rotorcraft attribute is set to the rotorcrafts remaining TOS (in range). This attribute is compared to the randomly created search range for the event to determine whether the rotorcraft finds the people or not. However, in this analysis, the range was set to a value that allowed for people to always be found.

Evacuation

1. Initially, it is determined if any rotorcraft can take the evacuees straight to the remote. If that is the case, rotorcrafts that meet this requirement are then sorted by passenger capacity.
2. The command and control function makes assignments until the event requirement (total evacuees) is met or it runs out of rotorcraft.
3. Of the remaining rotorcrafts, it is determined if any can make it to the event, back to the local site and refuel. The rotorcrafts that satisfy this requirement are then sorted by passenger capacity.
4. Step 2 is then repeated for these vehicles.

MedEvac

1. The command and control logic for MedEvac is similar to that for Evacuation, except these events always require transporting people from the event site to the local site. Initially, the rotorcraft list is reduced to those rotorcrafts that can perform the mission.
2. The reduced list is sorted by litter capacity.
3. The command and control function makes assignments until the event requirement is met or it runs out of rotorcrafts.

Cargo Transport

1. The rotorcraft list is reduced to those rotorcrafts which can perform the mission.
2. The reduced list is sorted by payload size.
3. The command and control function makes assignments until the event requirement (total payload) is met or it runs out of rotorcrafts.

Note: If the command and control function exhausts the supply of available rotorcrafts before the event requirement is met, then the event is placed back on the event queue with the reduced event requirement. For example, it there was an evacuation event for 450 people and after assignments were made there were still 65 people left from this event, then the event would still exist in the queue but with only a 65-person requirement.

Mission Priority

In this analysis, a simple ordering scheme was investigated to prioritize mission assignments. The scheme allowed ranking of missions in order of execution importance where a ranking of 1 is of high importance and 10 is of low importance. The missions were then ordered in the mission assignment queue based on this ranking. A mission priority was initially defined as follows:

- MedEvac (1)
- Evacuation (2)
- SAR (3)
- Cargo Transport (4)

However, during tests, it was discovered that this simple prioritization scheme needed additional mission logic in order to be useful. The high-priority missions dominated the mission assignments until all of those missions were completely exhausted. Therefore, for this analysis, all missions were given the same priority and assigned in the order in which they were received or first-in–first-out (FIFO).

Verification

Verification was performed on simulation software algorithms to confirm accuracy of the results.

Assumptions

Analysis assumptions were made to bound and provide definition of the modeling detail required of the analysis. Assumptions, if not found appropriate, can affect the behavior of the analysis model and, therefore, affect the results of the analysis. The assumptions made during the performance of this comparative study are considered appropriate when considering the capabilities of a conceptual rotorcraft product performing post-disaster mission tasks. The following analysis assumptions were agreed upon by the project managers at the beginning of the project:

- Aircraft are configured to carry the required mission payloads (no time lost for altering configuration).
- Unlimited supply of fuel at bases.
- 100% aircraft availability (meaning that a given asset is always available if it is not currently assigned to a mission).
- No distinction between day and night operations.
- Adequate landing zones available at all sites.
- Among the CRCs, only the CRC-L and CRC-M are allowed to perform SAR missions.
- Among the CTRs, only the CTR-L is allowed to perform SAR missions.
- 75% minimum rotorcraft passenger occupancy requirement for event-to-remote site flight.
- 86% (150 passengers of the175 maximum occupancy) minimum FW passenger occupancy requirement to depart from local site.
- Remote site distance for FW aircraft is 400 nmi.
- Unlimited number of FW assets available.
- 45-minute load/unload time for FW assets (Local Site and Remote Base).
- No distinction between internal and external cargo loads.
- No time delay for treating/stabilizing non-ambulatory evacuees.
- No distinction between ambulatory and non-ambulatory evacuees transported by FW.
- Rotorcrafts are refueled after each mission.

- Evacuees always located during SAR missions.

Analysis Results and Observations

Various fleet configurations and mission demands were formulated into run cases and executed within the modeled scenario. The results of these run cases, as well as analyst observations, are presented in the following sections. Also, additional run case excursions were spawned, executed, and documented to support analyst observations.

Analysis Run Cases

An analysis run matrix was defined to compare various configurations of a rotorcraft fleet under various mission demands. The following configurations and mission demands were considered:

- Baseline CRC fleet with baseline mission demand
- Reduced CRC fleet augmented with CTRs and baseline mission demand
- Baseline CRC fleet augmented with CRCs and baseline mission demand
- Baseline CRC fleet augmented with CTRs and baseline mission demand
- Baseline CRC fleet with increased mission demand
- Baseline CRC fleet augmented with CRCs and increased mission demand
- Baseline CRC fleet augmented with CTRs and increased mission demand

The run matrix was revised, after initial run executions and sensitivity analysis and ultimately agreed upon by project management. These run cases are provided in Table 5. Figure 11 provides a charted and annotated description of the run cases.

Run ID 0 (Baseline)

A baseline case was executed to provide an initial basis of reference for the analysis. The baseline case consisted of a multi-class CRC fleet responding to the baseline event types and quantity.

Run IDs 1-6 (CRC Replacement & Reduction)

These cases consist of a 25% reduction in each class of CRC and replaced by a progressively reduced number of CTR aircraft. The fleet configuration for Run ID 1 is a replacement of 25% of the CRC aircraft in each class of the baseline fleet with CTR assets. Fleet configurations for Run ID 2-4 consists of a progressive reduction in all classes of CTR aircraft by 25%. Run IDs 5 and 6 were run case excursions added late in the analysis to help determine the point in which the baseline fleet's MOEs equaled those provided by a mix of CRC and CTR aircraft.

Run IDs 7-12 (Fleet Augmentation)

These cases consist of 10, 20 and 30% increases to the quantity of rotorcraft in the baseline fleet. Run IDs 7, 9, and 11 augment the baseline fleet with CRCs in all classes and Run IDs 8, 10, 12 augment the baseline fleet with CTR aircraft increases in all classes.

Run IDs 13-15 (25% Demand Increase)

These cases provide an evaluation of 3 fleet configurations under increased rotorcraft demand (increased number of events of all types) of 25%. These fleets evaluated consist of the baseline fleet, the baseline fleet augmented with 25% additional CRC assets in all classes, and baseline fleet augmented by 25% additional CTR assets in all classes.

Run IDs 16-18 (50% Demand Increase)

These cases provide an evaluation of 3 fleet configurations under increased rotorcraft demand (increased number of events of all types) of 50%. These fleets evaluated consist of the baseline fleet, the baseline fleet augmented with 25% additional CRC assets in all classes, and the baseline fleet augmented by with 25% additional CTR assets in all classes.

Run IDs 19-21 (75% Demand Increase)

These cases provide an evaluation of 3 fleet configurations under increased rotorcraft demand (increased number of events of all types) of 75%. These fleets evaluated consist of the baseline fleet, the baseline fleet augmented with 25% additional CRC assets in all classes, and the baseline fleet augmented by with 25% additional CTR assets in all classes.

Run IDs 22-24 (100% Demand Increase)

These cases provide an evaluation of 3 fleet configurations under increased rotorcraft demand (increased number of events of all types) of 100%. These fleets evaluated consist of the baseline fleet, the baseline fleet augmented with 25% additional CRC assets in all classes, and the baseline fleet augmented by with 25% additional CTR assets in all classes.

Table 5. Analysis Run Cases

	Light		Medium		Heavy			
Run ID	CRC	CTR	CRC	CTR	CRC	CTR	Total R/C	Number of Events
0	95	0	233	0	68	0	396	Baseline
1	71	24	175	58	51	17	396	Baseline
2	71	18	175	44	51	13	372	Baseline
3	71	12	175	29	51	9	347	Baseline
4	71	6	175	15	51	4	322	Baseline
5	71	5	175	12	51	3	317	Baseline
6	71	2	175	6	51	2	307	Baseline
7	105	0	256	0	75	0	436	Baseline
8	95	10	233	23	68	7	436	Baseline
9	114	0	280	0	82	0	476	Baseline
10	95	19	233	47	68	14	476	Baseline
11	124	0	303	0	89	0	516	Baseline
12	95	29	233	70	68	21	516	Baseline
13	95	0	233	0	68	0	396	Baseline + 25%
14	119	0	291	0	85	0	495	Baseline + 25%
15	95	24	233	58	68	17	495	Baseline + 25%
16	95	0	233	0	68	0	396	Baseline + 50%
17	119	0	291	0	85	0	495	Baseline + 50%
18	95	24	233	58	68	17	495	Baseline + 50%
19	95	0	233	0	68	0	396	Baseline + 75%
20	119	0	291	0	85	0	495	Baseline + 75%
21	95	24	233	58	68	17	495	Baseline + 75%
22	95	0	233	0	68	0	396	Baseline + 100%
23	119	0	291	0	85	0	495	Baseline + 100%
24	95	24	233	58	68	17	495	Baseline + 100%

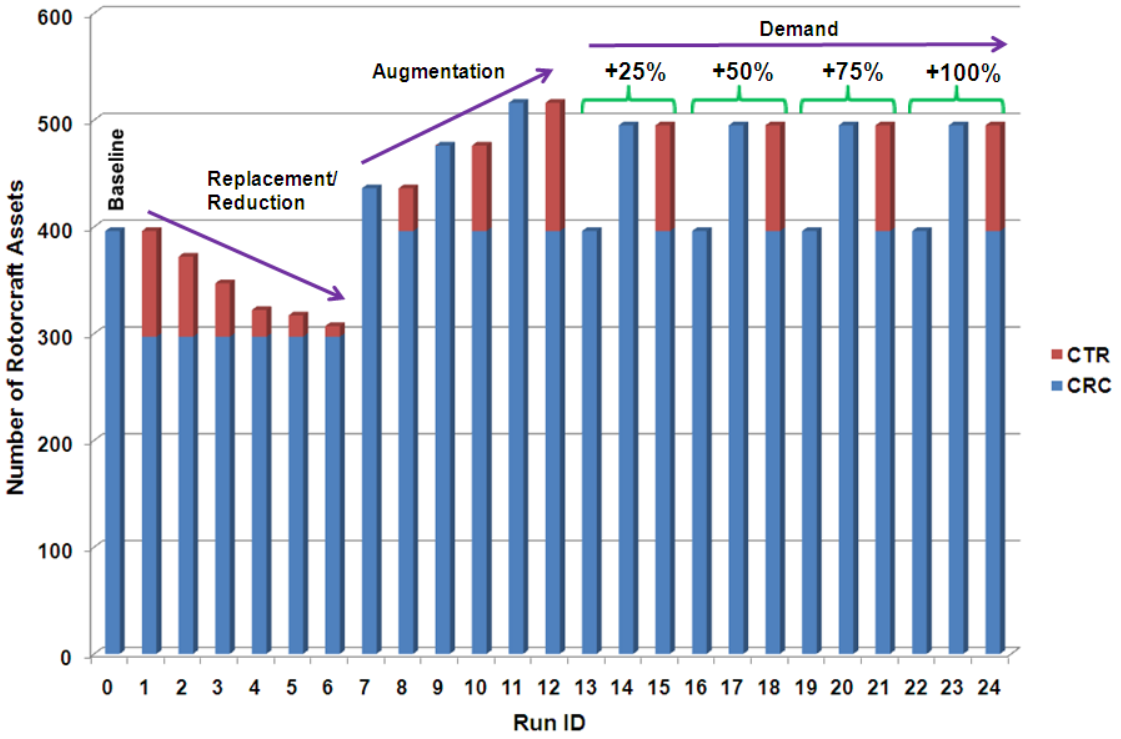


Figure 11. Charted Run Matrix

Measures of Effectiveness

To provide a means of comparison, measures of effectiveness (MOE) where captured for each run case. MOEs provide a system-level metric for quantifying the performance of a solution in meeting requirements and provide decision makers a basis for selecting a particular solution. The following MOEs were evaluated:

- Fleet Build-Up
- Average Response Time for Evacuation
- Average Response Time for Cargo Transport
- Evacuation Productivity
- Cargo Transport Productivity

Individual mission requirements are a function of event creation parameters described in previous sections of this report. The requirement determines the type and quantity of rotorcraft deployed on a mission. For the baseline run (Run ID 0), 6,800 missions were executed during the scenario simulation. The quantity and percentage of missions required by mission type is presented in Figure 12. As shown in the figure, SAR missions occurred most often. This can be attributed to the following:

- The total number of people requiring rescue during the scenario (24,367)
- The number of people requiring rescue per mission (1-10)
- The passenger capacity of rotorcraft allowed to perform the SAR mission
    - CRC-L (8 Pax)
    - CRC-M (18 Pax)
    - CTR-L (10 Pax)

Given that over 24,000 people required rescue during the scenario and the maximum number of people requiring rescue per mission is 10, the large number of SAR missions is explained. This same logic can be used to explain other mission type totals for the baseline run case. It can also be deduced that, while holding the mission requirements constant and increasing rotorcraft performance, the number of missions required for certain mission types may decrease and, as a result, provide a reduction in fleet operations and support cost incurred.

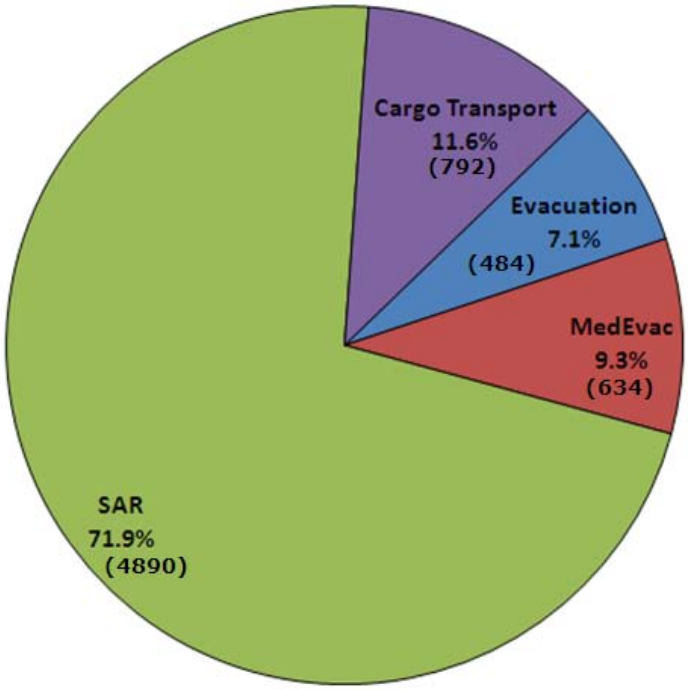


Figure 12. Missions by Type for Baseline (Run ID 0) Case

Fleet Build-Up

This MOE evaluates the time required to move assets from remote bases to the local base within the disaster area at the beginning of the simulated scenario.

At the beginning of the simulation, rotorcraft assets are located at remote bases at random ranges from the center of the disaster area. These assets are relocated from these remote bases to a local base within the disaster area. During travel, some assets may require refueling. If so, a refueling time will be added to the total travel time. The total travel time of the assets is used to determine fleet build-up statistics. These statistics, provided in Table 6, were calculated for the baseline CRC fleet (Run ID 0) and a rotorcraft fleet containing a mix of CRC and CTR aircraft (Run ID 1).

The fleet evaluated in Run ID 1 contains the same number of rotorcraft assets as the baseline; however, 25% of the CRC assets are replaced with CTR assets in all asset classes. Run ID 1 realized a 15% reduction in average asset travel time to the local base. This provided assets within the disaster area more quickly than the baseline fleet and allowed assets to begin operations sooner.

Table 6. Fleet Build-Up

Run ID	Mean (hours)	Min (hours)	Max (hours)	Std. Dev. (hours)
0	5.35	2.13	9.26	1.78
1	4.56	0.93	9.26	2.10

Average Response Time for Evacuation

This MOE evaluates the average time between an evacuation, MedEvac, or rescue request event, and the time the service is rendered (evacuation to a remote site).

In Run IDs 1-6, the effect of reducing the number of CRCs in the baseline fleet and replacing those CRC with CTR aircraft is investigated by calculating the fleet's average response time for evacuation. As shown in Figure 13, the replacement of CRCs with CTRs reduced the average response time as compared to the baseline fleet.

Run ID 1 (25% replacement) with 396 rotorcraft provided the greatest reduction in average response time of 22%. The smallest fleet evaluated was in Run ID 6 with only 307 rotorcrafts. Run ID 6, a 25% reduction in CRCs and replacement by a small quantity of CTRs, performed better than the baseline by reducing the average response time by 7%. The results show that all fleets evaluated consisted of a sufficient quantity of rotorcraft to meet the event demands at a pace better than the baseline fleet's capability.

Given the assumptions and stochastic parameters defined in this analysis, the reduction in response time can be attributed to the following:

- Greater speed offered by all CTR aircraft in each class
  - Faster initial arrival at local base
  - Faster execution of missions
- Greater range offered by the CTR in each class (specifically the CTR-L and CTR-M)
  - Remote site is accessible by all CTR aircraft
  - Very low probability that the remote site is accessible by CRC-L and CRC-M

Figure 13. Evac, SAR, & MedEvac Response Time - CRC Replacement & Reduction

The greater speed of the CTR aircraft allowed some fleet assets to arrive at the local base and begin evacuation operations much sooner than CRCs in the baseline fleet. Once at the local base, the greater speed and range offered by the fleet augmented by CTR aircraft reduced the response time across all mission types. The greater range reduced the response time for general evacuation and SAR missions by providing the additional capability of evacuating people directly to the remote site, bypassing the intermediate, local site and eliminating the associated delay. Evidence of this is shown in Figure 14. The baseline fleet evacuated 27% of the people directly to the remote site while Run ID 1 evacuated 43%. As a result, fewer FW flights were required when CTR aircraft were employed in the fleet as shown in Figure 15. The smaller quantity of rotorcraft and FW aircraft in the fleet outperformed the baseline fleet and may offer a solution that provides increased safety and lower operating costs.

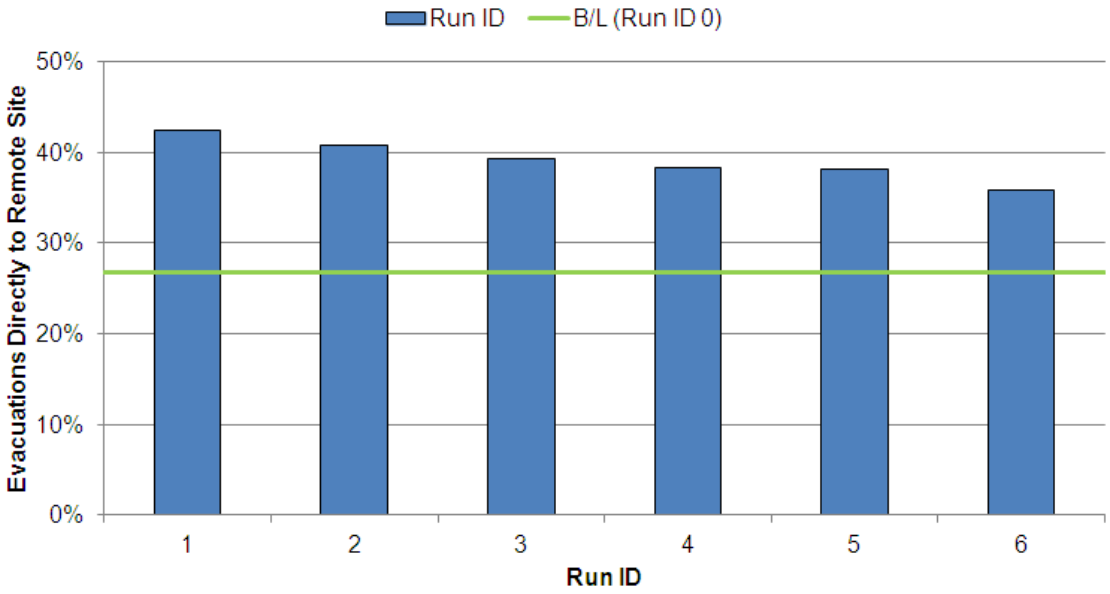


Figure 14. Evacuations Directly to Remote Site - CRC Replacement & Reduction

The reduction and replacement of CRC with CTR aircraft produced a measurable reduction in the average response time required to complete the simulation's evacuation events within the 11 day simulation time frame. However, it is recognized that a rotorcraft fleet that is capable of meeting the same mission requirements within a shorter timeframe is desirable. That analysis will be presented later in this report.

In Run IDs 7-12, the effect on evacuation response time due to the augmentation of the baseline rotorcraft fleet with CRC and CTR aircraft is investigated. As shown in Figure 16, the augmentation of the CRC fleet with additional CRCs (Run IDs 7, 9, and 11) provided a modest reduction in response time reduction when compared to the baseline fleet. As shown in Figure 17, the augmentation of the baseline fleet with additional CRCs did not increase the number of people evacuated directly to the remote site even with the addition of rotorcraft capable of meeting the range requirements. This is due to the baseline scenario not stressing the capabilities of the baseline fleet. Additionally, Figure 18 shows no reduction in the number of fixed-wing aircraft flights. This suggests that the response time reduction was due to additional usage of CRC assets with larger passenger capacity and speed which provided a more efficient and speedy transfer of people from the event location to the local base.

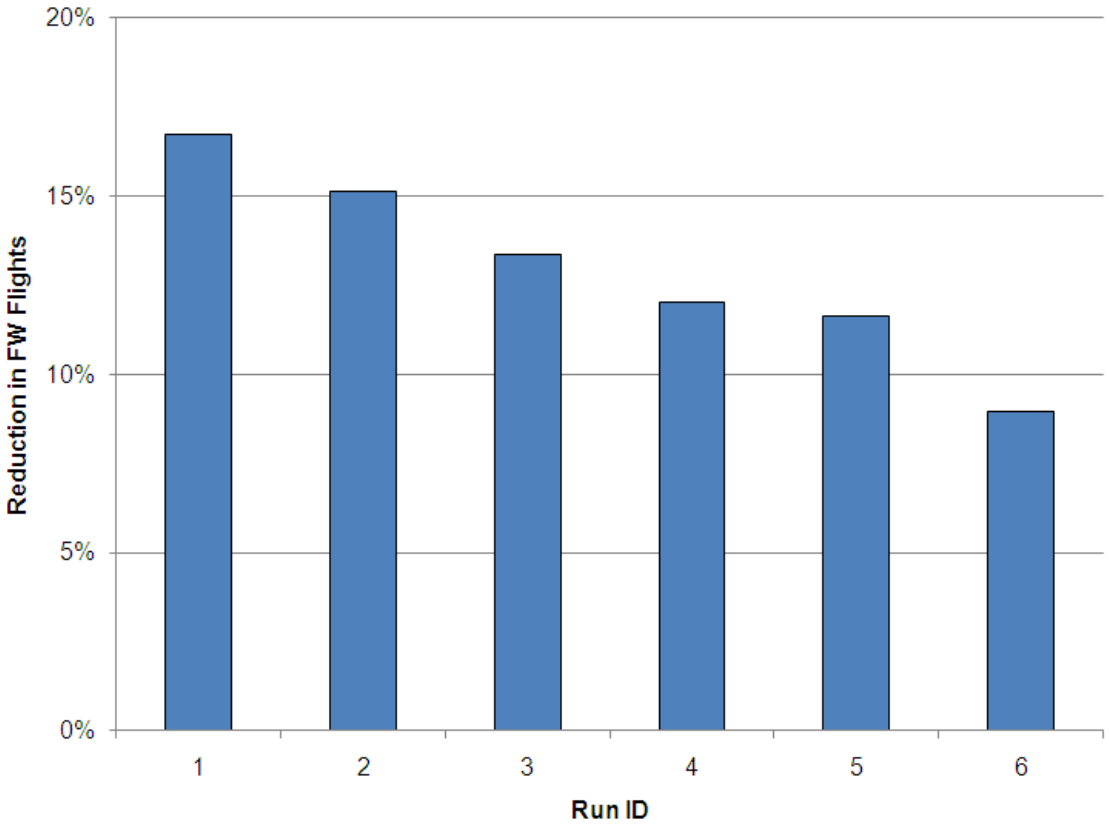


Figure 15. Reductions in FW Flights - CRC Replacement & Reduction

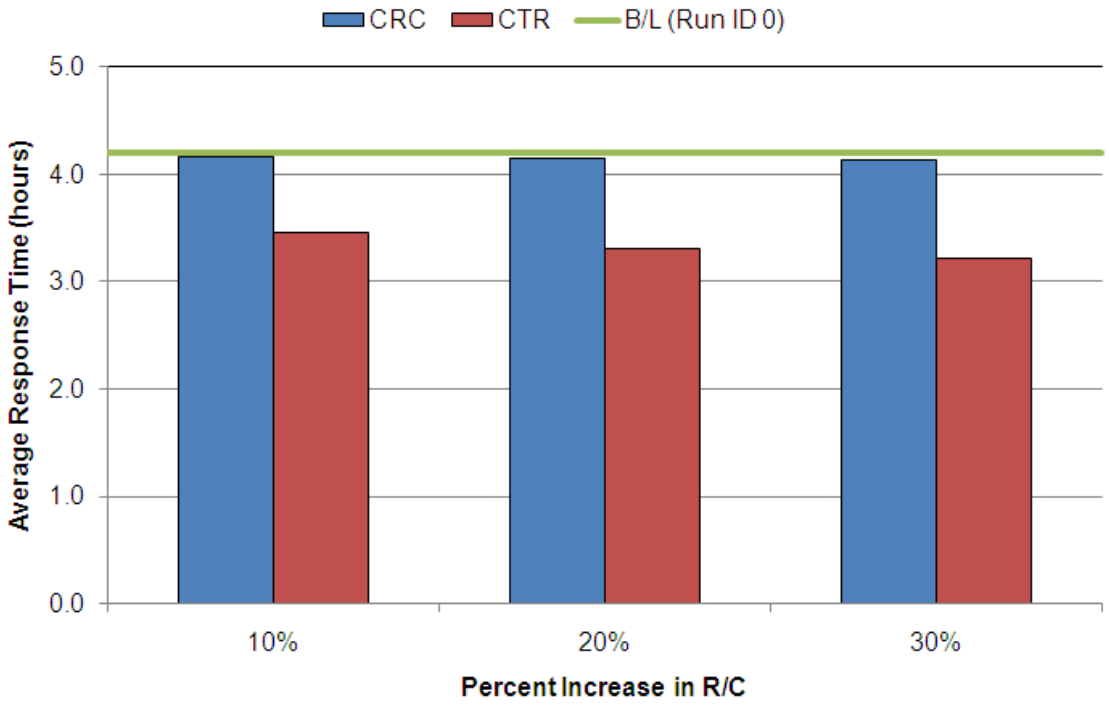


Figure 16. Evac, SAR, & MedEvac Response Time - Fleet Augmentation

Figure 17. Evacuations Directly to Remote Site - Fleet Augmentation

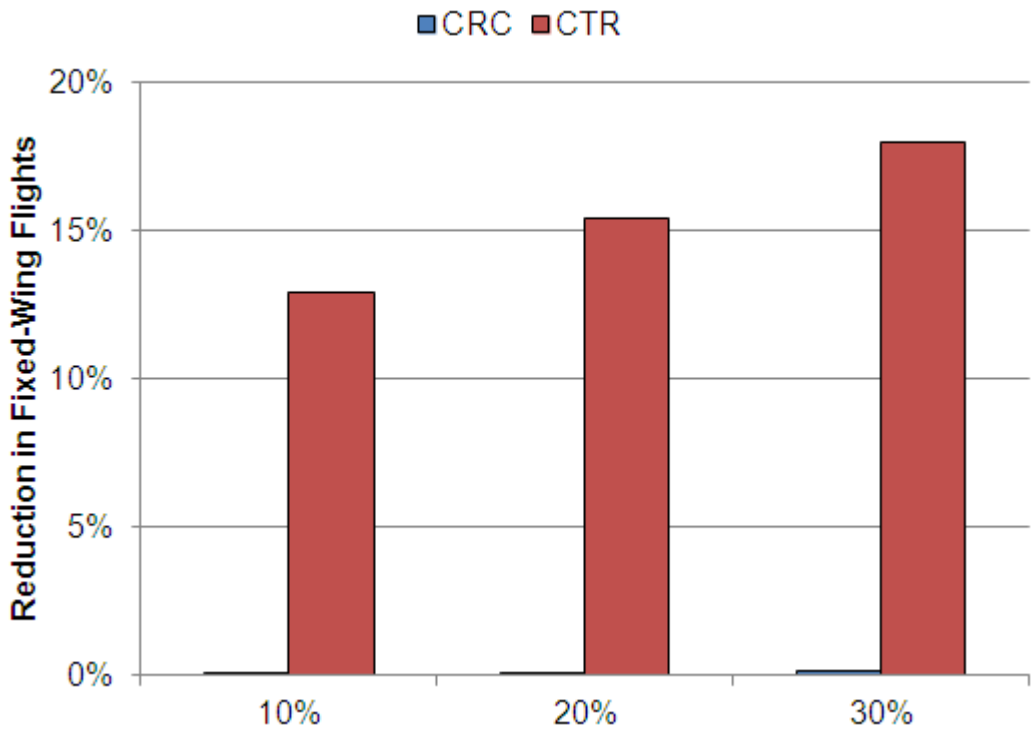


Figure 18. Reduction in Fixed-Wing Flights - Fleet Augmentation

The augmentation of the baseline fleet with CTR aircraft (Run IDs 8, 10, and 12) reduced the response time significantly as compared to the baseline CRC fleet (Figure 16). Due to the range limitations of the CRC-L and CRC-M, these rotorcrafts required offloading evacuees at the local site for remote evacuation by FW aircraft. However, the baseline CRC fleets

augmented with CTR aircraft provided a reduced evacuation response time (Figure 16) by providing the capability to evacuate people directly to the remote site (Figure 17) and therefore reducing the number of evacuations by fixed-wing aircraft (Figure 18). As the percentage of CTR aircraft is increased within the fleet, more SAR missions were assigned to the CTR-L. However, the greater assignment of CTR-L aircraft to SAR missions did not translate to an equal reduction in response time due to the passenger occupancy requirement of 75% for remote site evacuation. Additionally, the increased number of CTR-M and CTR-H aircraft did not significantly improve the fleet's ability to evacuate people directly to the remote site. This is a result of adequate resources already being available for evacuation missions and the lower probability of not meeting the passenger occupancy requirements for remote evacuation.

The effectiveness of the baseline fleet and two augmented baseline fleets was investigated while increasing the fleet demand by increasing the number or scenario events (i.e. number of people evacuated, tons of cargo transported). The augmented fleet configurations and demands are identified in Run IDs 13-24. The two fleet configurations consisted of augmenting the baseline CRC fleet with the following:

- Additional 25% increase in CRCs in all classes.
- Additional 25% increase in rotorcraft of CTR aircraft type in all classes.

The increased fleet demand was provided by increasing all event types by 25%, 50%, 75% and 100%. As shown in Figure 19, the baseline CRC fleet is the most susceptible to increased mission demand and the baseline fleet augmented with CTR aircraft is the least susceptible. As the demand increases, the response time increases. The response time increase over the range of demand increase is as follows:

- Baseline fleet response time increased 40%.
- CRC augmented fleet response time increased 17%.
- CTR augmented fleet response time increased 1%.

As events increased, it became increasingly harder for the baseline CRC fleet to meet the demand requirements. Increasing the baseline fleet with additional CRC assets made it easier for the fleet to meet demand. The additional assets provided more assets that could be used for local evacuation events, specifically CRC-L and CRC-H, and more assets that could be used to evacuate people directly to the remote evacuation site, specifically CRC-H.

The addition of CTR aircraft to the baseline fleet provided rotorcraft in the light and medium classes with range capability to evacuate people directly to the remote site. This impact is most noticed in evacuation missions. With the CTR aircraft available to the command and control function for mission assignments, the CTR would be the aircraft of choice because of its range, speed and passenger capacity. Additionally, the CTR aircraft were able to meet each demand increase with very little effect on response time. Only a 1% increase in response time was observed over the range of event demands.

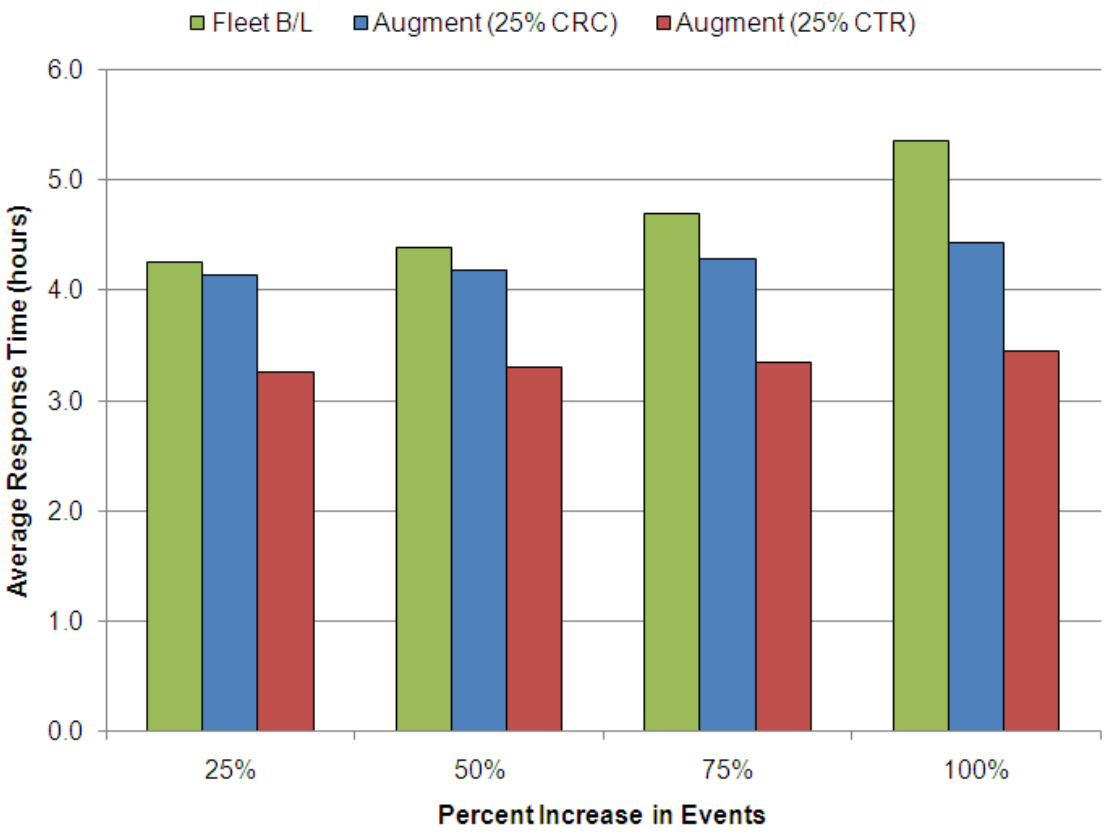


Figure 19. Evac, SAR, & MedEvac Response Time - Demand Increase

Average Response Time for Cargo Transport

This MOE evaluates the average time between a transport request and the time the service is rendered (delivered to destination). The pick-up and drop-off sites are both local to the disaster area of operation.

The average response time for cargo transport is evaluated in Run IDs 1-6. As shown in Figure 20, the replacement of CRCs with CTRs reduced the average response time as compared to the baseline CRC fleet. Run ID 6 matched the baseline response time with significantly fewer rotorcraft assets in the fleet. Specifically, the improvement in response time can be attributed to the availability of CTR assets and the speed and range provided by the CTRs augmenting the baseline fleet. The greater range of the CTR only affected this MOE at the beginning of the scenario since this mission is performed in the local area. The greater range allowed the CTR to arrive at the disaster area more quickly and begin transport operations requests that were waiting in the rotorcraft assignment queue within the command and control function. When missions were assigned to the CTR assets, missions were performed more quickly.

In Figure 21, the response time is presented for increases of rotorcraft in the baseline fleet of 10, 20, and 30% as described in Run IDs 7-12. Given the CTR speed advantage, a more pronounced reduction in response time would seem reasonable. However, this did not occur due to the rotorcraft payload characteristics and the command and control function's mission assignment methodology.

Figure 20. Cargo Transport Response Time - CRC Replacement & Reduction

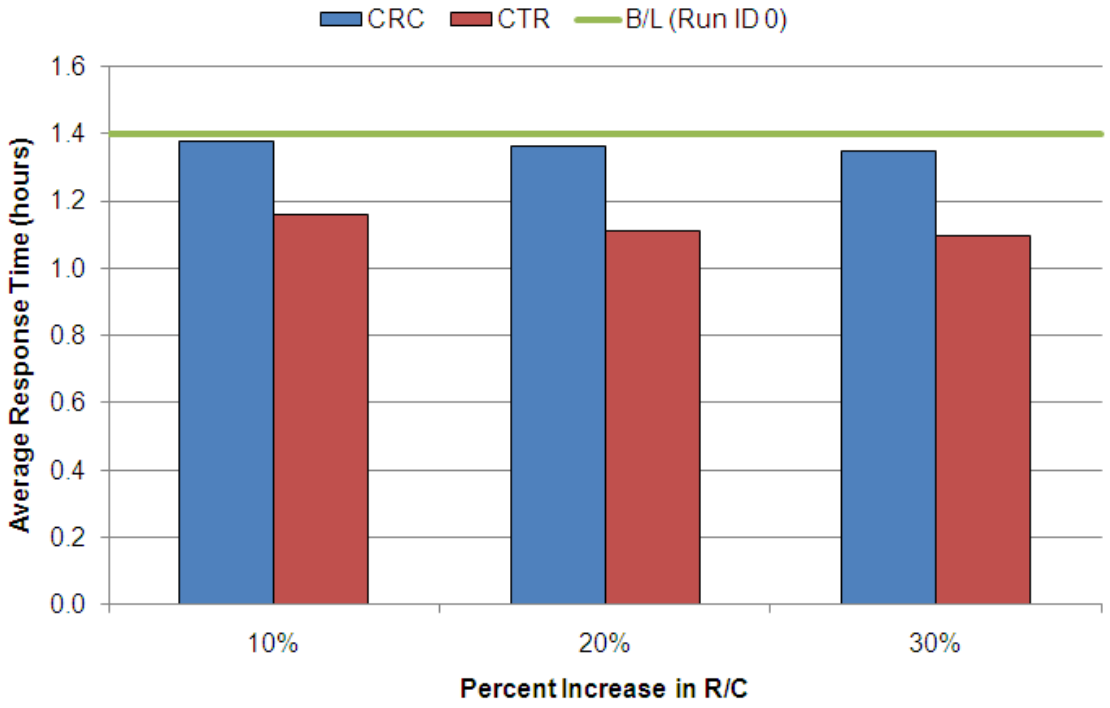


Figure 21. Cargo Transport Response Time - Fleet Augmentation

The command and control function's assignment methodology (as described in a previous section of this report) initially eliminates those rotorcrafts from the rotorcraft assignment queue that can't perform the mission due to the mission's range requirement. Next, the command and control function sorts the remaining rotorcraft in the queue by payload capacity with rotorcraft having the greatest payload capacities at the top of the queue. The command and control

function will then scan down the queue until it finds the rotorcraft that most closely matches the payload requirement. If a single rotorcraft cannot be found to meet the mission's payload requirement, the requirement may be broken up into smaller payloads and assigned to multiple rotorcrafts.

Given the command and control's assignment methodology described in the previous paragraph, the majority of the cargo transport missions are assigned to CRC assets. This is confirmed when comparing Run ID 11 (30% increase in CRC in all classes) and 12 (30% augmentation of CTR in all classes) presented in Table 7. Providing that rotorcrafts are available, the following assignments will occur:

CRC-L: 500 lbs (minimum requirement) ≤ load requirement ≤ 1,669 lbs.
CTR-L: 1,669 lbs < load requirement ≤ 2,220 lbs.
CRC-M: 2,200 lbs < load requirement ≤ 6,254 lbs.
CTR-M: 6,254 lbs < load requirement ≤ 6,600 lbs.
CRC-H: 6,600 lbs < load requirement ≤ 19,462 lbs.
CTR-H: 19,462 lbs < load requirement ≤ 20,000 lbs (maximum requirement).

Table 7. Cargo Transport Missions for Run ID 11 and 12

R/C	Payload (lbs)	Number of Missions (Run ID 11)	Number of Missions (Run ID 12)
CRC-L	1,669	67	48
CRC-M	6,254	182	166
CRC-H	19,462	538	476
CTR-L	2,200	N/A	11
CTR-M	6,600	N/A	51
CTR-H	26,399	N/A	27

Using Run IDs 13-24, the effect of increasing mission events of all types on cargo response time was evaluated for three fleet configurations. The configurations consist of the baseline CRC fleet, the baseline fleet augmented with a 25% increase in all class types, and a 25% increases in rotorcraft of CTR type in all classes.

As was discussed in the previous paragraph, assignment methodology, speed and rotorcraft availability contributed to the difference in response time between the CRC augmented fleet and the CTR augmented fleet. As shown in Figure 22, the increased mission demands has a more of an effect on the baseline fleet and augmented CRC fleet rather the CTR fleet. As demand increases the availability of rotorcraft is reduced. When CRC availability becomes more limited, more CTR assignments are made and the greater CTR speed and payload capacity benefit is revealed.

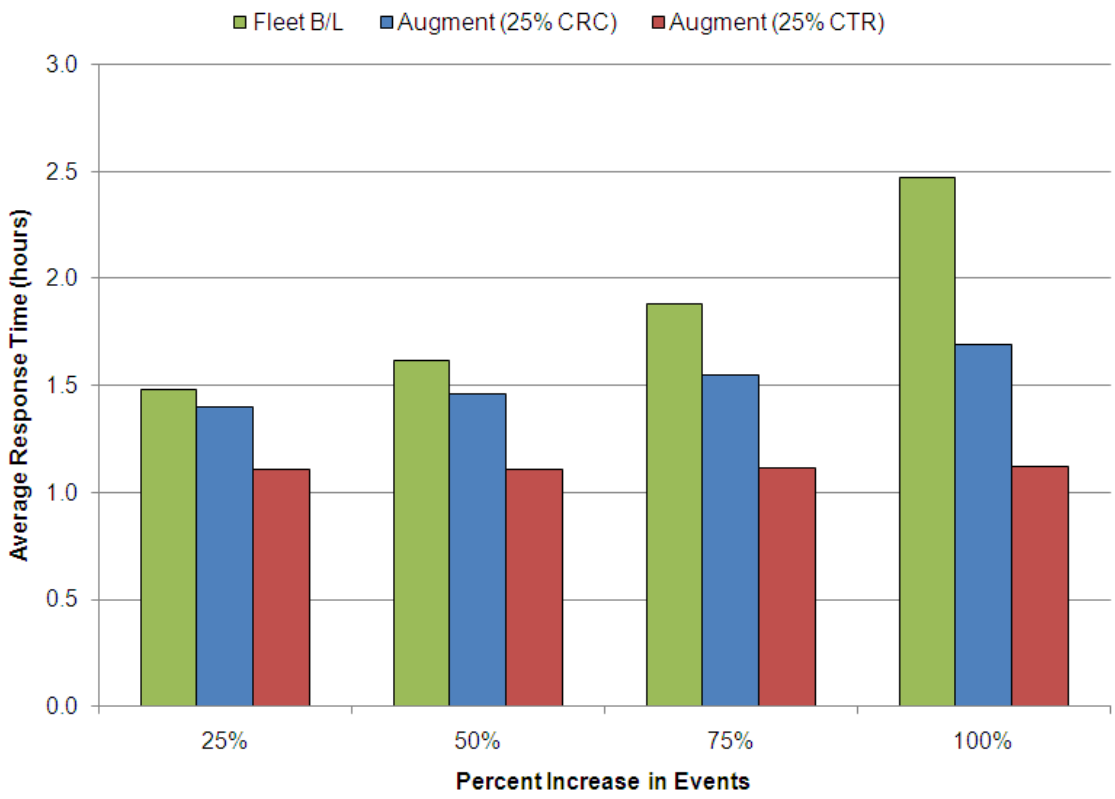


Figure 22. Cargo Transport Response Time - Demand Increase

Evacuation Productivity

This MOE evaluates the build-up, or amassing, of evacuees at the remote evacuation sites outside the disaster area.

The final destination of the 53,071 evacuees is the remote sites outside the disaster area. The goal is to evacuate the required number of people in the shortest amount of time possible. The baseline scenario provided 11 days to evacuate everyone which did not stress the CRC or CTR fleets evaluated. As shown in Figure 23, four run cases are presented that represent four significant fleet configurations (e.g. baseline, reduction/replacement, augmentation). The only significant difference in the evacuation productivity curves is at the very beginning of the scenario. At the beginning of the scenario, rotorcraft assets had to travel from remote bases to the disaster area which delayed the time in which rotorcraft assets could get fully engaged in performing evacuation missions.

Although the fleets evaluated were able to meet the event demands for evacuation, the number of missions required was significantly reduced when CTR aircraft are employed within the fleet. As shown in Figure 24, over 50% of the general evacuations missions were eliminated and over 40% of the MedEvac. This is due to the passenger and litter capacity provided by the CTR rotorcraft. The number of SAR missions was unaffected because no reduction is possible given the SAR mission assignment methodology imposed requiring one rotorcraft per event. The reduction in the overall missions required may translate into lower fuel and maintenance costs.

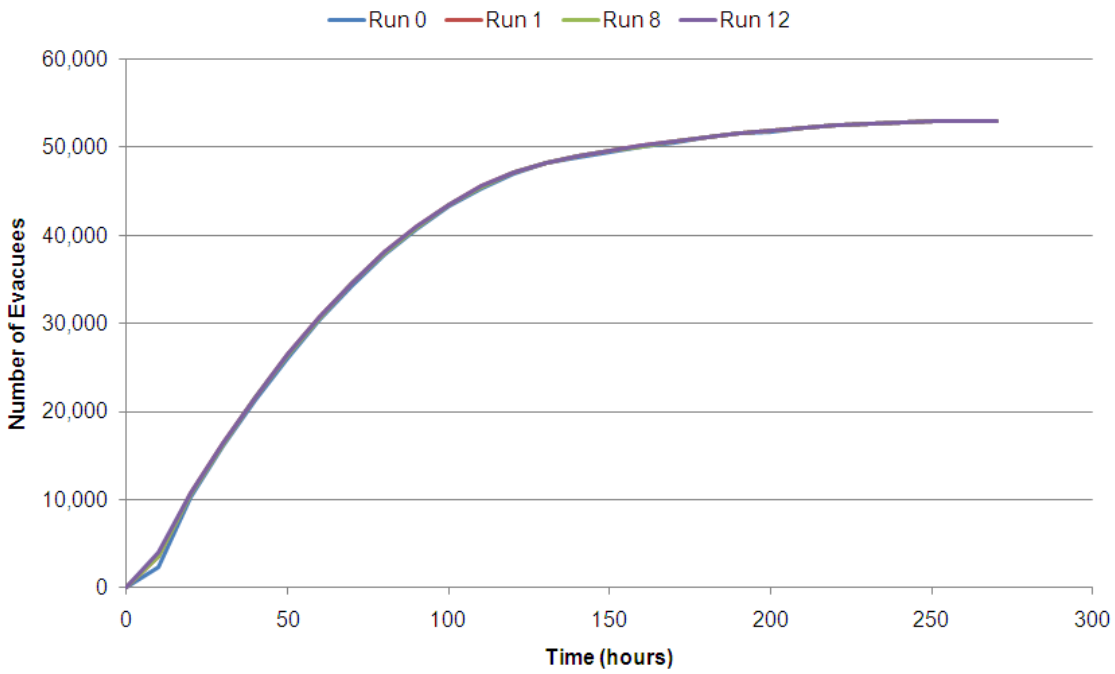


Figure 23. Evacuation Productivity - 11 Day Simulation Duration

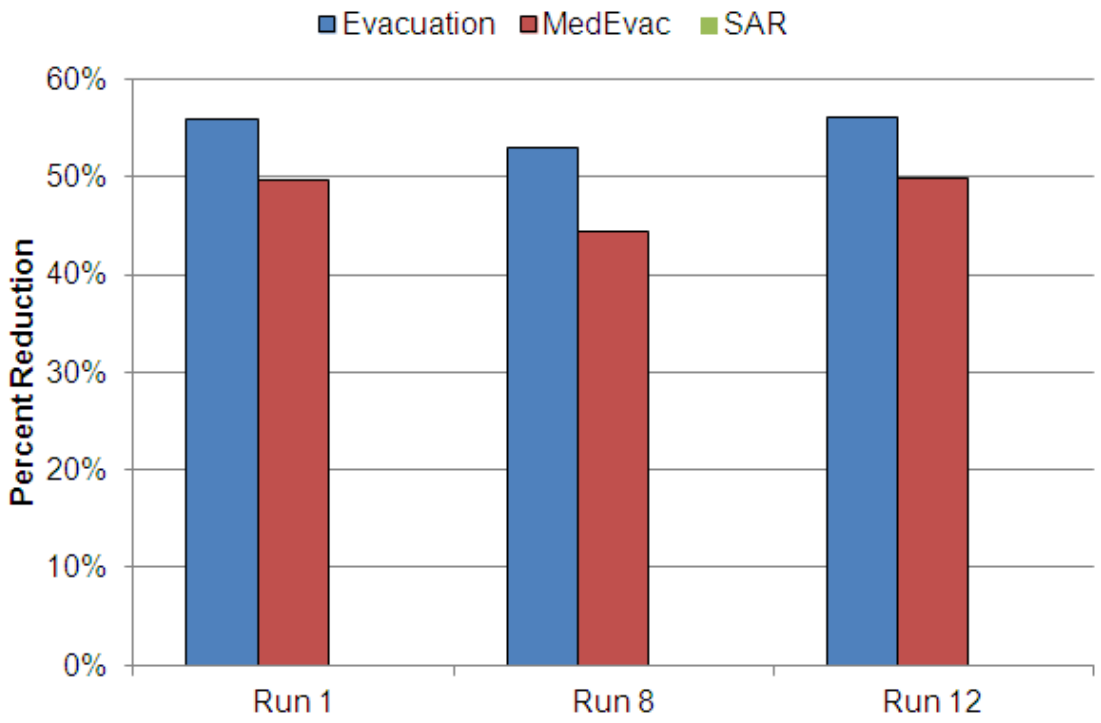


Figure 24. Reduction in Evac, MedEvac, and SAR Missions Compared to B/L

To stress the fleet demand and gain more insight into their capability, the simulation duration was shortened to 2 days (48 hours). As shown in Figure 25, each fleet was able to perform the evacuations with the 2 day time period. During this analysis, the quantity and combination of CRC and CTR aircraft proved important in the fleet's productivity capability.

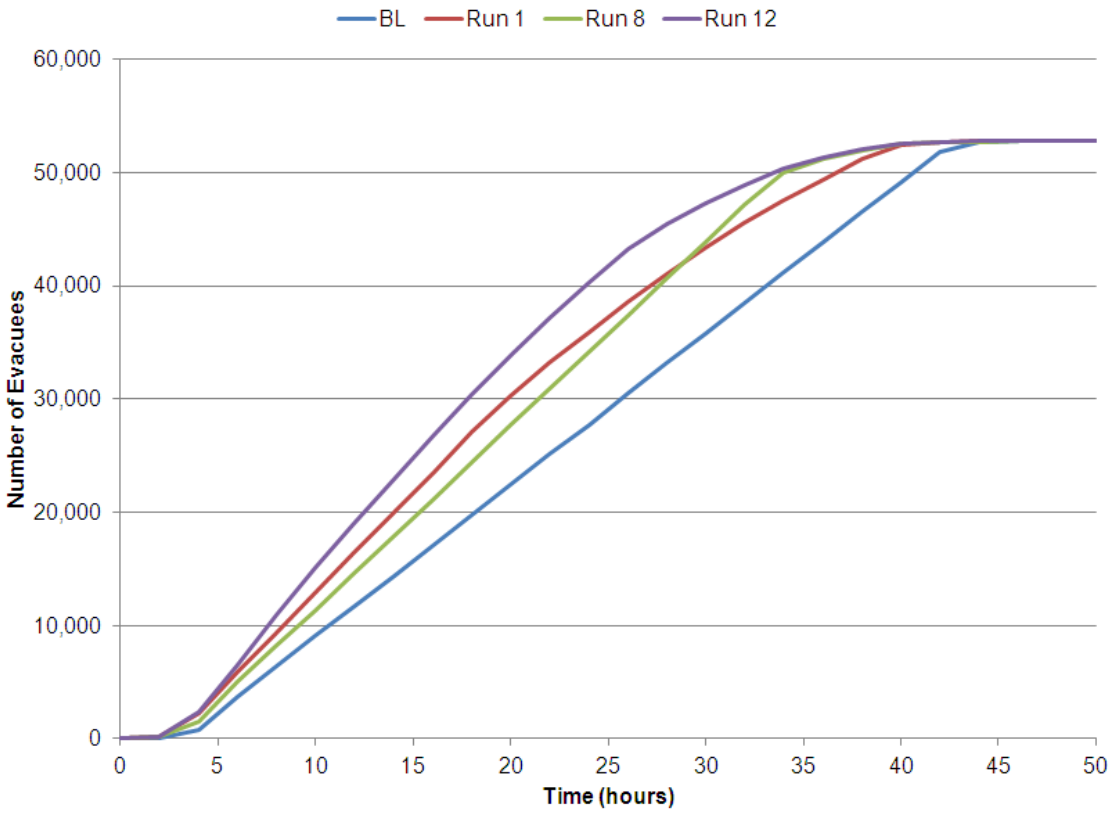


Figure 25. Evacuation Productivity – 2 Day Simulation Duration

Of the fleets evaluated, Run ID 12 provided the best evacuation productivity. This outcome was expected. The fleet contained the most rotorcraft (516) and consisted of the baseline CRC fleet augmented with 30% increase in rotorcraft of the CTR aircraft type in all classes (light, medium and heavy). The fleet was able to perform the assignments of the command and control function and evacuate people to the local and remote sites as required and within the simulation timeframe.

Run ID 1 and Run ID 8 provided an interesting fleet comparison and highlighted the importance of number and type of rotorcraft contained in the fleet. Run ID 1 consisted of 396 rotorcrafts with 25% of the baseline CRC class types replaced with CTR types. Run ID 8 consisted of 436 rotorcrafts with the baseline fleet augmented with 10% CTR types. Although the fleet defined in Run ID 1 had fewer rotorcrafts, the mission requirements at the beginning of the simulation best met the CTRs capability. These mission requirements allowed the CTR aircraft to evacuate people directly to the remote base and allowed remaining CTR and CRC aircraft to evacuate to the local site where FW aircraft ultimately evacuated them to the remote base. However as time progressed and event demand decreased, fewer missions allowed the CTR to utilize its range potential. Most missions consisted of evacuation of people to the local site for FW evacuation. At this point in the simulation, the fleet configuration defined for Run ID 8 proved more beneficial due to the quantity of rotorcraft it contained. More rotorcrafts available increased the rate at which FW aircraft evacuations were performed.

The greatest productivity rates for all fleets, as shown in Table 8, occurred at the beginning of the simulation when event demand of all types was at highest.

Table 8. Evacuation Productivity Improvement

Run ID	Productivity (Evacuees/Hour)	Productivity Improvement (%)
Run ID 0 (Baseline)	1333	N/A
Run ID 1 (25% CTR Replacement)	1761	32.1
Run ID 8 (10% CTR Augment)	1614	21.0
Run ID 12 (30% CTR Augment)	1974	48.1

Cargo Delivery Productivity

This MOE evaluates the delivery, or amassing, of cargo to destinations within the disaster area.

All transport missions are performed within the local area; no cargo is transported to or from remote bases or sites. Therefore, rotorcraft range is not an issue; the discriminator is rotorcraft speed, payload capacity and availability (not busy performing other missions). As shown in Figure 26, the four fleet configurations evaluated had no difficulty meeting the cargo transport demand during the 11 day simulation duration. However, more missions were required for certain fleet configurations.

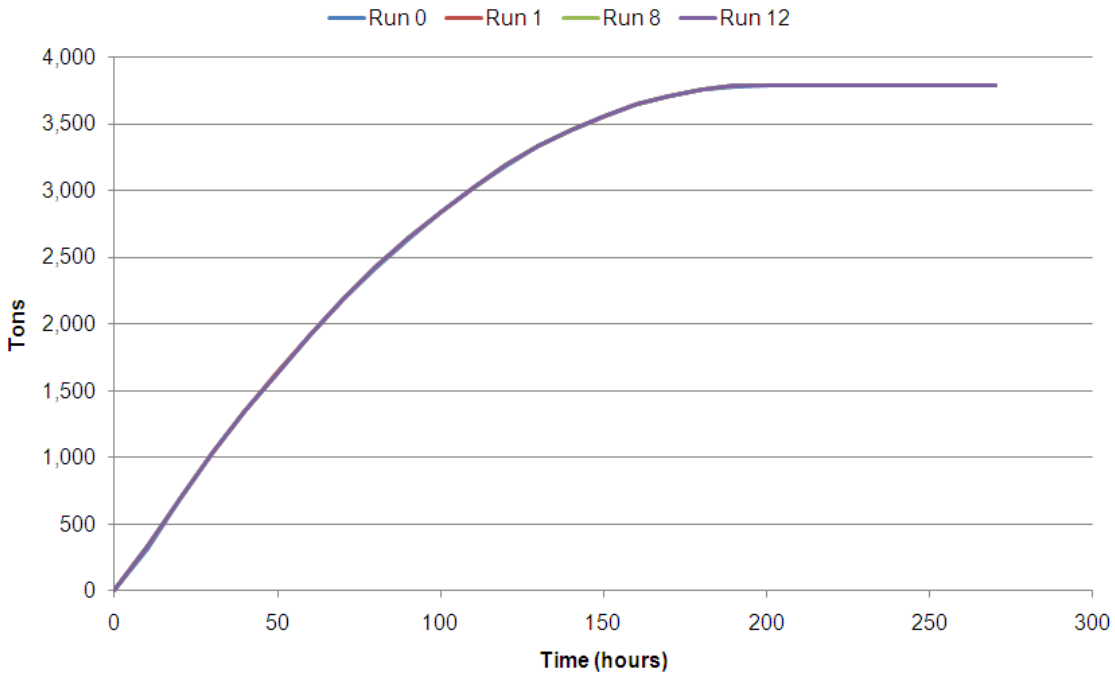


Figure 26. Cargo Transport Productivity - 11 Day Simulation Duration

As shown in Figure 27, the number of transport missions required was reduced when CTRs were introduced into the fleet. However, one might expect a greater reduction than is shown in the figure due to the greater speed of the CTR. As was explained for the transport response time MOE, a significant number of CTRs, although available, were not assigned to transport missions because of the command and control function's transport mission assignment logic, rotorcraft payload capacities and event payload requirements.

Figure 27. Reduction in Cargo Transport Missions Compared to B/L

In order to determine the maximum cargo transport productivity rate of the fleets, the fleet's productivity had to be stressed to its maximum; therefore, the simulation duration was again reduced to 2 days (48 hours). As shown in Figure 28, the baseline fleet was required to perform at a maximum productivity rate for most of the simulation. The remaining fleet configurations, consisting of CRC and CTR assets, operated initially at their maximum productivity rates and then were able to perform at lower productivity rates for the remainder of the scenario and easily meet the cargo transport demands with additional capability to spare.

The greatest productivity rates for all fleets, as shown in Table 9, occurred at the beginning of the simulation when event demand of all types was at highest.

Table 9. Cargo Transport Productivity Improvement

Run ID	Productivity (Tons/Hour)	Productivity Improvement (%)
Run ID 0 (Baseline)	96	N/A
Run ID 1 (25% CTR Replacement)	150	55.7
Run ID 8 (10% CTR Augment)	120	24.3
Run ID 12 (30% CTR Augment)	176	82.4

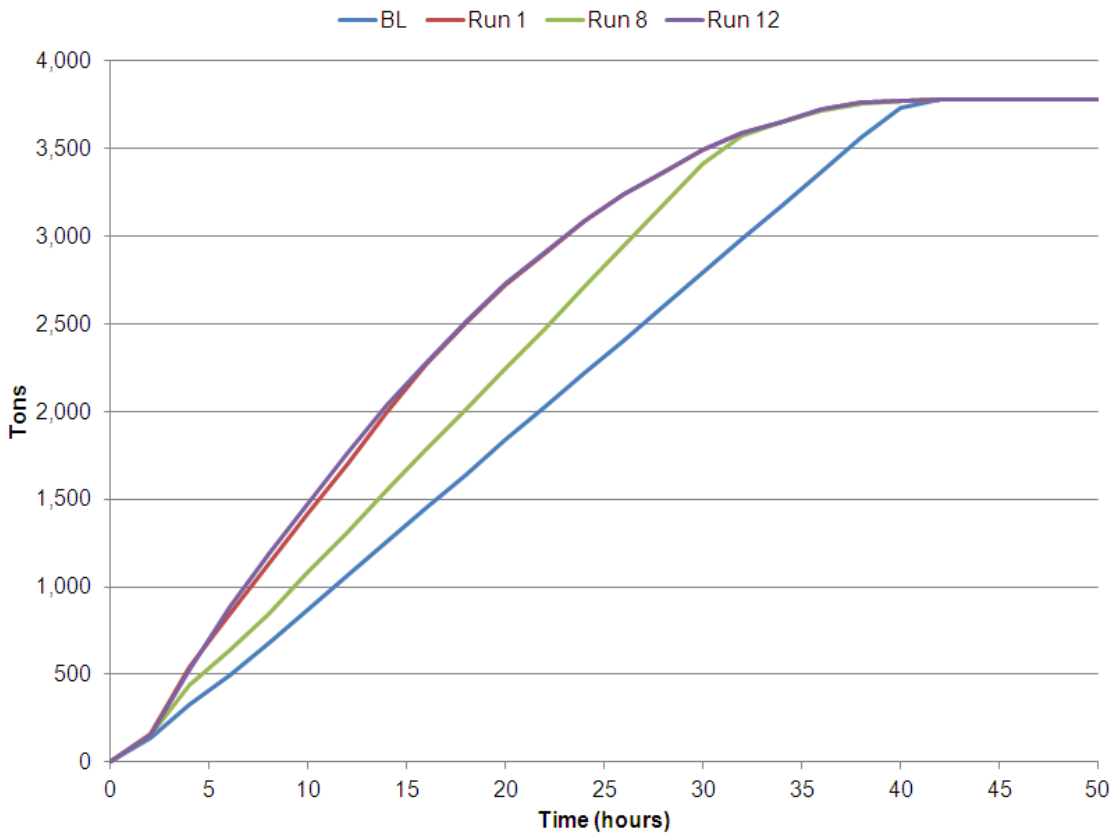


Figure 28. Cargo Transport Productivity – 2 Day Simulation Duration

Parameter Sensitivity Analysis

In this section, the MOE sensitivity to analysis parameters is investigated. A sensitivity analysis evaluates the relative importance of parameters within the analysis by evaluating their influence on the analysis results. Specifically, the MOEs may be influenced by the following:

- Aircraft quantity (rotorcraft and FW)
- Aircraft performance characteristics
- Fleet configuration (asset type, quantity)
- Event parameters (type, quantity)
- Analysis parameters (duration, probability distribution functions, etc.)
- Operational strategy
- Analysis assumptions

This analysis evaluated the baseline fleet's evacuation and cargo transport response time sensitivities to parameters decreased and increased by 25%. Specifically, these parameters along with baseline values and descriptions are provided in Table 10.

In Figures 29-33, the sensitivity to these analysis parameters are graphically presented in tornado, bar and x-y chart formats. Additionally, analyst observations were recorded and presented in the following sections.

Table 10. Sensitivity Analysis Parameters

Parameter	Baseline Value	Description
CTR (L,M,H) Quantity	0	quantity of light, medium, and heavy class CTR aircraft
CTR-H Quantity	0	quantity of heavy class CTR aircraft
CTR-M Quantity	0	quantity of medium class CTR aircraft
CTR-L Quantity	0	quantity of light class CTR aircraft
CRC (L,M,H) Quantity	396	quantity of light, medium, and heavy class CRC
CRC-H Quantity	68	quantity of heavy class CRC
CRC-M Quantity	233	quantity of medium class CRC
CRC-L Quantity	95	quantity of light class CRC
Simulation Duration	264 hours	length of time that events are being generated
Distance to Remote Base	650 nmi	mean distance from the remote base to the local base
Distance to Remote Site	400 nmi	mean distance from the local event to the remote site
FW Aircraft Quantity	Unlimited	quantity of fixed-wing aircraft located at the local site
All Event Type Quantity	25%	percentage of event quantity of all types (cargo transport, evac, SAR, MedEvac)
Cargo Transport Event Quantity	3,790 tons	quantity of cargo transport events
SAR Event Quantity	24,367 people	quantity of SAR events
Evacuation Event Quantity	18,078 people	quantity of evacuation events
MedEvac Event Quantity	10,626 people	quantity of MedEvac events
FW Pax Occupancy Threshold	150 people	queue size threshold required for fixed-wing aircraft to depart the remote site
R/C Pax Occupancy Threshold	75%	occupancy threshold required for rotorcraft to evacuate directly to the remote site
Event Creation Delta T	6 hours	time duration for creating individual mission requirements

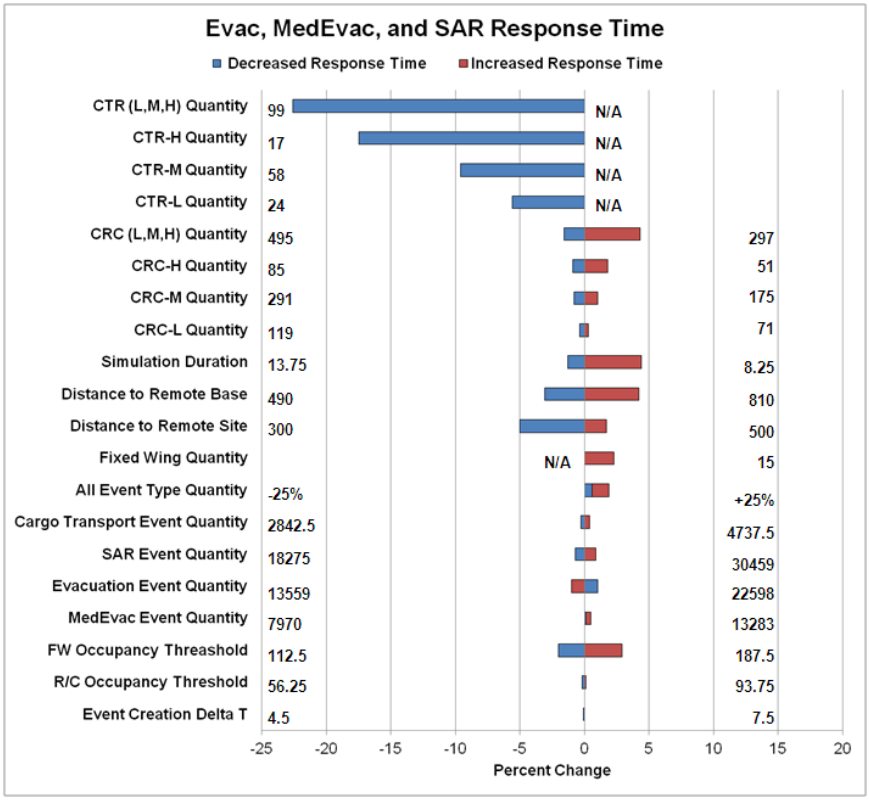


Figure 29. Evac, SAR, and MedEvac Response Time Parameter Sensitivity

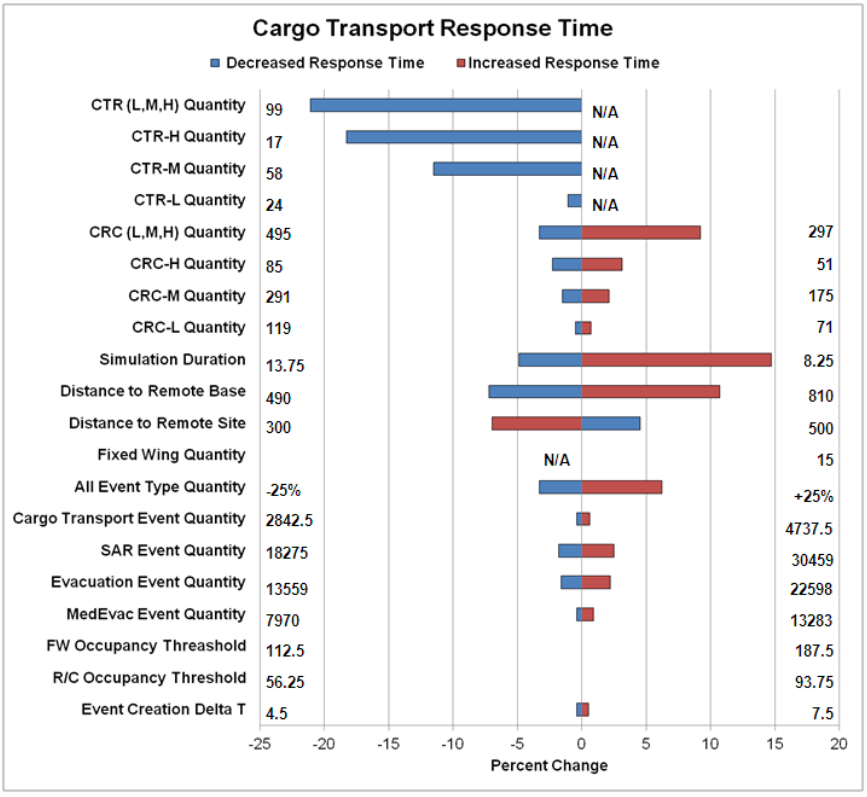


Figure 30. Cargo Transport Response Time Sensitivity

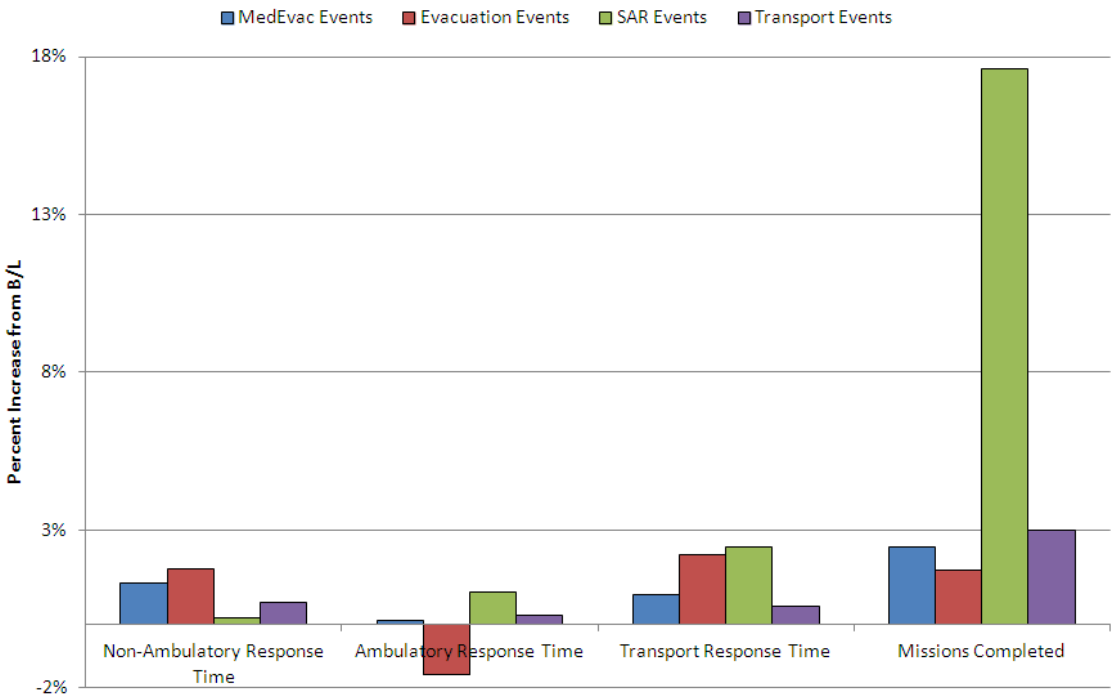


Figure 31. Event Type Sensitivity (+25%)

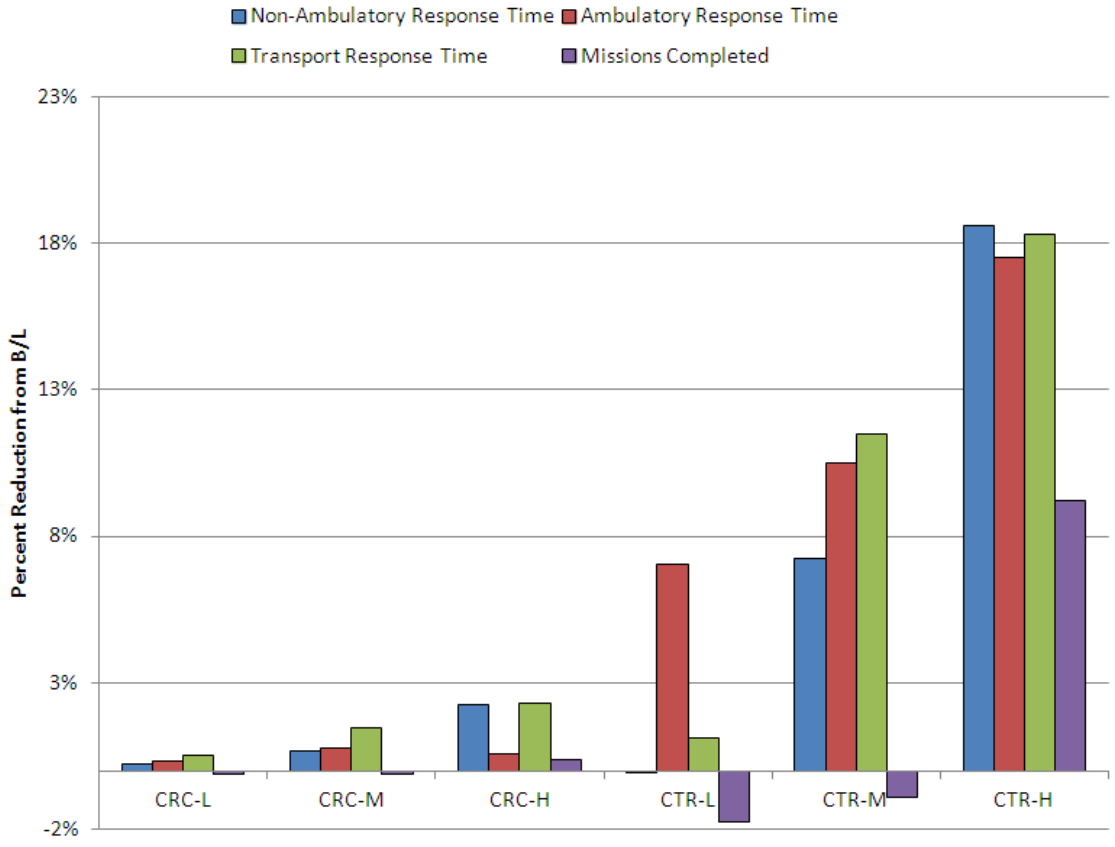


Figure 32. Fleet Type Sensitivity (+25%)

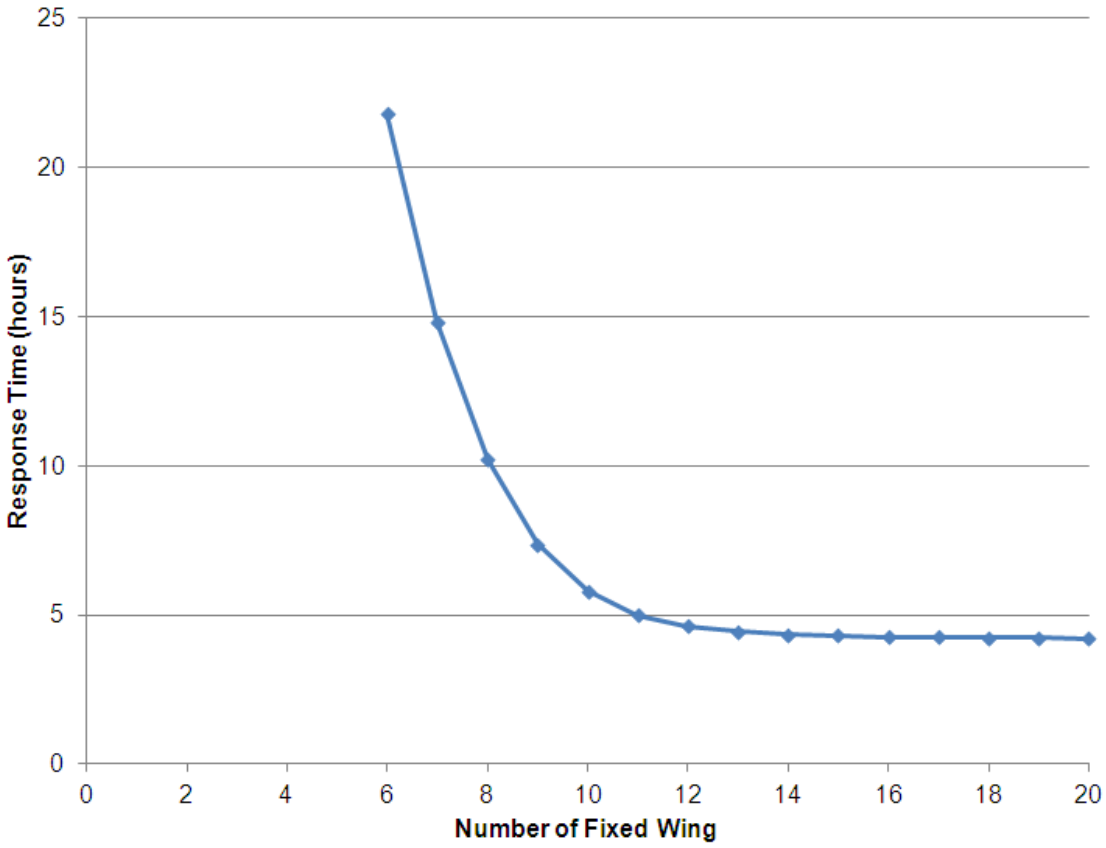


Figure 33. Fixed-Wing Aircraft Influence on Evac, SAR, and MedEvac Response Time

Evacuation, SAR, and MedEvac Response Time Parameter Sensitivity

The sensitivity to evacuation, SAR, and MedEvac response time is shown in Figure 29 with observations provided in the following sections.

CTR Quantity: A 25% increase in CTR aircraft had the greatest effect on reducing evacuation response time. When adding CTRs of each class type to the fleet, the CTR-H had the most positive influence on reducing response time. The CTR-H allowed the combining of multiple evacuation missions into a single mission. Additionally, the increased speed and range, offered by the CTR aircraft, allowed the evacuees to be transported quickly to the remote site. Since there were no CTR aircraft contained within the baseline fleet, the sensitivity to a decrease in CTR aircraft was not performed.

CRC Quantity: As documented in previous sections, increasing the number of CRCs in the fleet had a negligible effect on response time. However, it was observed that removing 25% of the CRCs, within each class, increased the evacuation response time by 4.3%. When decreasing the number of CRCs in the fleet by individual class types, the CRC-H reduction had the most negative influence (increase) on response time. This was due to the command and control function dividing larger evacuation missions into missions with smaller evacuee quantities and, as a result, the missions were assigned to lighter, slower rotorcraft assets.

Simulation Duration: Increasing the simulation duration by 25%, provided a 1.3% reduction in the response time. The negligible reduction is attributed to the baseline rotorcraft fleet being

very capable of performing the baseline mission. However, if the simulation duration is decreased 25%, the rotorcraft asset becomes moderately stressed and a 4.4% increase in response time is realized.

Distance to Remote Base: Decreasing and increasing the mean remote base distance has a more even effect on the response time, resulting in a 3.1% decrease in response time and a 4.2% increase in response time, respectively. As provided by the event creation methodology, the bulk of the events occur at the beginning of the scenario. These events start accumulating in the command and control function's event queue before rotorcraft assets have arrived at the local base and are available for mission assignments. Hence, the remote base distance influences the overall average response time by affecting the rotorcraft arrival time at the local base (disaster area) or time in which rotorcraft mission assignments can begin.

Distance to Remote Site: Decreasing the mean remote site distance allowed more rotorcraft to perform evacuation missions directly to the remote site, reducing the response time by 5%. Increasing the mean remote site distance only increased the response time by 1.7%. This is due to the reduction of rotorcraft assets within the baseline fleet that were capable of meeting the range requirements associated with the remote site location, specifically the CRC-H assets. The increased distance and range capability of the assigned rotorcraft caused more evacuation missions to be terminated at the local site rather than remote site.

Fixed-Wing Quantity: The simulation assumed an unlimited quantity of fixed-wing assets available at the local site for evacuation of people to the remote site. Therefore, the sensitivity of response time to increasing the quantity of fixed-wing aircraft was not necessary. It was determined that the minimum of 20 fixed-wing aircraft were required to have no negative effective on the baseline fleet's response time. When the availability of 15 fixed-wing aircraft were evaluated (25% reduction), a 2.3% increase in response time was realized as, on occasion, evacuees queued at the local site had to wait for the availability of a fixed-wing aircraft for evacuation to the remote site. This effect is presented graphically in Figure 33.

Event Quantity: A 25% increase or decrease in the quantity of events had little effect on evacuation response time. In fact, due to operational rules and assumptions modeled in the simulation, a slight increase in response time was noted for both sensitivity runs. The following simulation rules and assumptions that affected this sensitivity included:

- All MedEvac missions transfer the non-ambulatory people to the local site for immediate medical attention. Therefore, the capability of rotorcraft to transport evacuees directly to the remote was not a factor.
- SAR rotorcraft passenger occupancy or 75% required for remote site evacuation of passengers and event requirements of 1 to 10 passengers. Only CRC-L (8 Pax), CRC-M (18 Pax) and CTR-L (10 Pax) rotorcraft were allowed to perform the SAR mission. Therefore, it was extremely rare, in the baseline scenario, for a SAR mission to go directly to the remote because of the range requirement and the occupancy threshold requirement.
- The average evacuation response time was based upon the average response time for evacuating people in MedEvac, Evacuation and SAR missions.

Additionally, it was observed that when only evacuation events were increased or decreased, the trend is in the opposite direction (i.e. adding more evacuation events actually decreases response time). Considering the points presented it is understandable why increasing evacuation events will increase response time. It is also notable that approximately

62% of those additional evacuees are transported directly to the remote site and decrease the average evacuation response time.

Fixed-Wing Occupancy Threshold: Although an unlimited number of fixed-wing assets were assumed, an occupancy threshold was set. A threshold of 150 people needed to be in the queue before a fixed-wing aircraft is assigned to move awaiting evacuees to the remote site. Increasing and decreasing this parameter by 25% showed a modest sensitivity in the baseline scenario and ranging from a 2% reduction in response time to a 2.9% increase in response time.

Rotorcraft Occupancy Threshold: The rotorcraft occupancy threshold had little effect on the response time for the baseline scenario. Most rotorcrafts were already limited by range and incapable of evacuating people directly to the remote. When rotorcraft performed evacuations missions directly to the remote site, most were at 100% passenger occupancy.

Event Creation Delta T: Varying the time step, in which events are created, by 25% had negligible effect for the baseline scenario.

Cargo Transport Response Time Sensitivity

Figure 30 shows the cargo transport response time sensitivity. Most of the trends were comparable to the trends seen with evacuation response time. It should be noted that any parameter associated with the fixed-wing transport of people has no effect on the cargo transport time. These parameters include:

- Fixed-Wing Aircraft Quantity
- Fixed-Wing Aircraft Occupancy Threshold
- Rotorcraft Occupancy Threshold

Distance to Remote Site: Decreasing and increasing the mean distance to the remote site has a reverse effect on the cargo transport response time. Increasing the remote site distance actually decreases cargo transport time. This can be explained by examining the missions assigned to the CRC-H aircraft. As the distance to the remote site becomes greater, fewer evacuation missions transport evacuees directly to the remote site which results in the availability of more CRC-H aircraft at the local base for local cargo transport mission assignments.

Event Type

To learn more about the influence of event type on response time, event type sensitivity was performed. The events for each event type (MedEvac, Evacuation, SAR, and Transport) were increased by 25% holding all other parameters of the baseline scenario constant. Figure 31 shows the effect on non-ambulatory, ambulatory, and transport response times along with the number of missions required. An increase in SAR events has the least effect on non-ambulatory response time and the greatest effect on ambulatory and transport response time and the number of missions. The increase in the number of missions is a result of the number of rescuees per SAR mission being very small (1-10). These additional SAR missions utilize CRC-L and CRC-M aircraft which reduces the availability of rotorcraft for other missions.

The reverse trend observed for the ambulatory response time with respect to evacuation events was explained in the last section.

Fleet Type

As shown in Figure 32, the sensitivity to fleet configurations changes by class was investigated. In this analysis, six runs were performed with the number of rotorcraft in the fleet increased by 25% per class. Specifically, the runs included the following additions to the baseline feet:

- 24 CRC-L
- 58 CRC-M
- 17 CRC-H
- 24 CTR-L
- 58 CTR-M
- 17 CTR-H

As observed during the execution of the run matrix, adding CRCs to the baseline fleet had very little effect on reducing evacuation response time. However, the addition of CTR aircraft to the baseline fleet had a profound effect on response times for increases in each CTR class type. The following are specific observations made during this analysis:

- Adding light and medium rotorcraft, whether CRC or CTR, increases the number of missions. In the CRC case, a unique scenario occurs where a heavy was needed but unavailable. However, now we have extra light and medium rotorcraft available so a few additional missions are generated. In the CTR case, even more missions are added as these rotorcrafts can take evacuees directly to the remote, replacing a single CRC-H mission.
- Adding CTR-L aircraft had no effect on non-ambulatory response time. This is due to the utilization of the CTRs to perform the Evacuation and SAR missions. The CTRs were used because their range capability allowed evacuees to be transported directly to the remote site.

Cargo Payload and Fleet Configuration Sensitivity

During the course of the analysis, an additional scenario was evaluated as an excursion to the run matrix. In this excursion, the cargo transport tonnage was increased by 10-fold to 37,900 tons holding all other events constant. For the baseline fleet configuration, the distribution of missions flown under this event demand is shown in Figure 34 with a comparison to the baseline demand. Such an extreme cargo transport requirement could be considered as a type of disaster relief scenario where a substantial amount of heavy equipment, material, and assets are needed to augment, replace or rebuild the destroyed infrastructure. In this run excursion, all other event demands (SAR, Evacuation, and MedEvac) are held at the same levels as the baseline event demand as defined in Table 2.

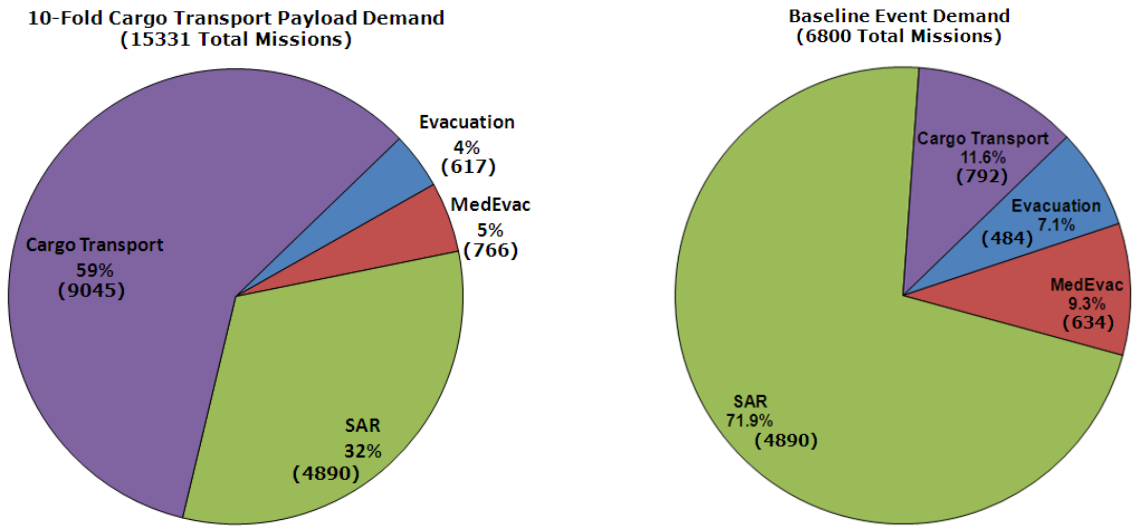


Figure 34. Missions by Type for 10-Fold Cargo Payload Demand vs. Baseline Payload

Relative to the baseline event demand, the number of evacuation missions increased slightly. This increase is due to heavy class rotorcraft not being available for assignment to these missions due to the increase infrastructure provisioning. Heavy class rotorcraft were primarily assigned to the cargo transport missions. With heavy rotorcraft engaged in cargo transport missions, more than one light or medium class rotorcraft were assigned to evacuation missions in order to meet some large event requirements. The number of SAR missions remained constant over each event demand which is a result of a maximum of only one SAR mission being required per SAR event.

With the cargo transport tonnage increase, the sensitivity to fleet configurations was evaluated. Specifically, the fleet configurations evaluated are defined as follows:

- Baseline fleet
- Baseline fleet with additional 25% CRC-L
- Baseline fleet with additional 25% CRC-M
- Baseline fleet with additional 25% CRC-H
- Baseline fleet with additional 25% light class rotorcraft of CTR-L type
- Baseline fleet with additional 25% medium class rotorcraft of CTR-M type
- Baseline fleet with additional 25% heavy class rotorcraft of CTR-H type

Figure 35, presents the percent reduction in response time and number of missions relative to the baseline. The most significant reduction occurs when CTR-M and CTR-H rotorcraft augment the baseline fleet. The addition of the heavier CTR rotorcraft allows more efficient utilization of the fleet with lighter rotorcraft assigned to Evac, MedEvac, and SAR missions while the heavy rotorcraft are assigned to transport cargo. As a result, a significant reduction in response time and number of missions required is realized.

If the baseline fleet is augmented by 30% of CTRs (Run ID 12), the maximum productivity rate (maximum number of people transported per hour) could be increased by 53.3%. These data points again show that the CTR's capability in improving the response time as well as productivity rate, in this scenario, though, with regards to a significantly different type of mission demand.

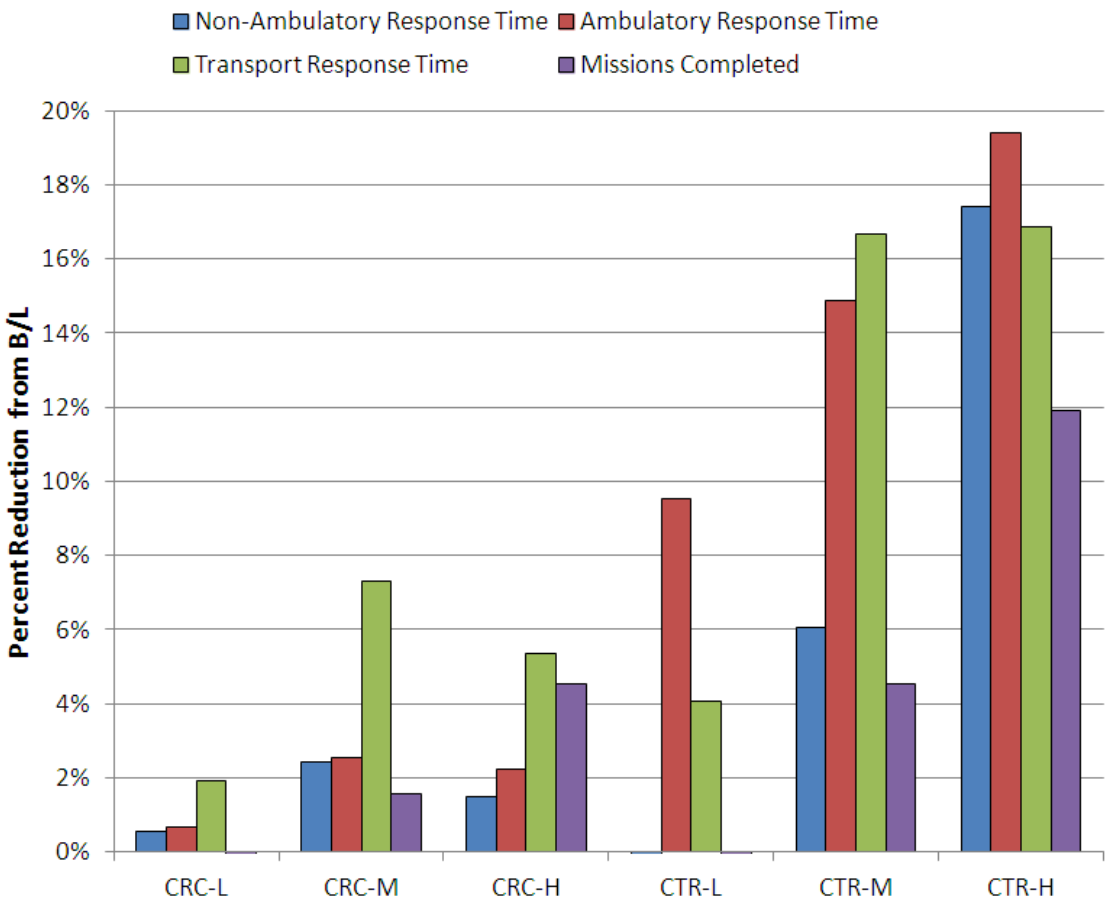


Figure 35. Fixed-Wing Aircraft Influence on Evac, SAR, and MedEvac Response Time

This run excursion, with extreme cargo transport requirement, emphasizes a key point that needs to be kept foremost in mind while interpreting the results from this study, which is: (1) additional simulations will need to be conducted in the future to gain an improved sense of the potentiality of a fleet of CTR aircraft responding to disaster relief missions and (2) there may be a spectrum of aircraft capabilities required to match the needs of an equally diverse set of emergency and disaster relief scenarios. Certain mission scenarios will be SAR dominated, for example the baseline Katrina-like scenario considered in this study, and some will be more cargo dominated, such as the extreme case just considered.

Conclusions and Recommendations

Analysis of conceptual commercial tiltrotor aircraft designs (Ref. 3) has provided a foundation upon which the operational suitability of these aircraft can be investigated within disaster scenarios. Tiltrotors offer significant improvements in speed and range for VTOL aircraft, and the potential for reduced response time in emergency operations. Results of this analysis suggest that these attributes can improve the effectiveness of a civil rotorcraft fleet performing hurricane post-disaster operations in conjunction with other air assets. As shown in Table 11, augmenting a fleet of conventional rotorcraft with a 10% increase in assets comprising tiltrotor aircraft has the potential to reduce overall mission response time significantly. In addition, such a fleet makeup may reduce reliance on fixed-wing transport segments as well as reduction of overall number of missions required to stabilize and recover from a disaster event. Rapid, integrated response may also reduce life and property loss, while

reduction in required flight operations may increase crew safety and reduce operational cost (i.e. fuel usage, maintenance, etc.).

Table 11. Effectiveness Benefit of 10% CRC & CTR Fleet Augmentation

Fleet	Evacuation Response Time Reduction	Cargo Transport Response Time Reduction	Reduction in Number of FW Flights	Reduction in Number of R/C Missions
Baseline Fleet + 10% CRC	1%	2%	0%	0%
Baseline Fleet + 10% CTR	18%	17%	13%	8%

Analysis conducted under this program provides credible insight into the benefits of augmenting existing rotorcraft operations with tiltrotors in simulated hurricane post-disaster operations. The NASA CTR team realizes the importance of exploring the issues necessary to address other mission scenarios consistent with national priorities. Comprehensive evaluations of the role of civil tiltrotors will need to include consideration of other disaster recovery missions, as well as related fleet operational strategies and CTR design considerations.

This program was necessarily limited in scope, and certain ground rules and assumptions were used to fix some aspects of the simulation conducted for hurricane response-centric aspects of this project. A complete assessment of analysis parameters, assumptions, and mission assignment logic (described in previous sections of this report) should be performed in future research to provide a consistent analysis framework for evaluating other scenario simulations, and to provide insight into scenario-specific sensitivities.

References

[1]JPDO, “Concept of Operations for the Next Generation Air Transportation Syste, Version 3.2,” Joint Planning and Development Office, Washington DC, 30 September 2010.

[2]Young, L., et al, “Civil Tiltrotor Aircraft Operations,” 2011

[3]Chung, W., et al, “Modeling High-Speed Civil Tiltrotor Transports in the Next Generation Airspace,” NASA CR 2011-215960, 2011.

[4]Young, L., et al, “A Study of Civil Tiltrotor Aircraft in NextGen Airspace,” AIAA-2010-9106, 10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference, Fort Worth, TX, September 2010

[5]Pritsker, A., *Introduction to Simulation and SLAM II* (3rd ed.), A Halsted Press Book, John Wiley & Sons, Systems Publishing Corporation, 1986.

[6]Fardink, P., “The Helicopter Response to Hurricane Katrina,” Vertiflite, Summer 2011

[7]U.S. Congress. “A Failure of Initiative: Final Report of the Select Bipartisan Committee to Investigate the Preparation for and Response to Hurricane Katrina,” U.S. House of Representatives Select Bipartisan Committee, 109th Congress, 2d Session, http://www.gpoacess.gov/congress/index.html, February 2006

[8]Boehm, W., Hylton, R., and Mehl, Major T., “In Katrina’s Wake: The National Guard on the Gulf Coast 2005,” Historical Services Division, Office of Public Affairs, National Guard Bureau.

[9]Helvarg, D., *RESCUE WARRIORS: The U.S. Coast Guard, America’s Forgotten Heroes*, New York: St. Martin’s Press, 2009.

[10]GAO Report to Congressional Committees, *Coast Guard: Observations on the Preparation, Response and Recovery Missions Related to Hurricane Katrina*, GAO-06-903, Washington, DC: GAO, July 2006.

[11]Price, S., “A Bright Light on the Darkest of Days: The U.S. Coast Guard’s Response to Hurricane Katrina,” Deputy Historian, U.S. Coast Guard.

[12]Bowman, S., Kapp, L., and Belasco, A., “Hurricane Katrina: DOD Disaster Response,” Congressional Research Service, CRS RL33095, September 2005

[13]U.S. Congress, *Hurricane Katrina*: A Nation Still Unprepared, Senate Report 109-322, 109th Congress, 2d Session, Washington, DC: GPO, 2006

[14]Landreneau, Major General B., “Hurricane Katrina: The Defense Department’s Role in the Response,” *National Guard Timeline of Significant Events Hurricane Katrina*, Senate Homeland Security and Governmental Affairs Committee, 109th Congress, 2d Session, February 9, 2006

[15]Wombwell, J., *Army Support during the Hurricane Katrina Disaster*, Fort Leavenworth, Kansas: Combat Studies Institute Press, 2005.

[16]McNally, Major J., “Hurricane Katrina Relief Operations – 498th Medical Company (AA) After Action Review,” September 23, 2008.

[17]Haulman, Dr. D., “The U.S. Air Force Response to Hurricane Katrina,” Air Force Historical Research Agency, November 17, 2006

[18]Director of Mobility Forces after Action Report, Joint Task Forces Katrina and Rita, October 18, 2005.

[19]Melnyk, Major L., “Katrina Lessons Learned,” *Soldiers Magazine*, June 20, 2006

[20]“Unprecedented Helicopter Force Takes Field in Katrina’s Wake,” *Rotor & Wing*, October 1, 2005

Acronyms

B/L – Baseline
CRC – Conventional Rotorcraft
CRC-H – Conventional Rotorcraft – Heavy
CRC-L – Conventional Rotorcraft – Light
CRC-M – Conventional Rotorcraft - Medium
CTR – Civil Tiltrotor
CTR-H – Civil Tiltrotor – Heavy
CTR –L – Civil Tiltrotor – Light
CTR-M – Civil Tiltrotor – Medium
Evac - Evacuation
FIFO – First In, First Out
FW – Fixed-Wing
ID - Identification
M^3S – Mathematical Monte-Carlo Modeling and Simulation
MedEvac – Medical Evacuation
MOE – Measure of Effectiveness
MTOW – Maximum Takeoff Weight
NextGen – Next-Generation Air Transportation System
Pax – Passengers
R/C - Rotorcraft
SAR – Search and Rescue
SLAM® - Simulation Language for Alternative Modeling
TOS – Time on Station
VTOL – Vertical Takeoff and Landing

Appendix A - Model Design

Table of Contents

List of Figures

List of Tables

Introduction

The purpose of this document is to capture the design of the discrete event simulation model which will assess the effectiveness of CTR assets, deployed from multiple bases, executing post-disaster relief operations associated with a hurricane disaster scenario.

The Mathematical Monte-Carlo Modeling & Simulation (M^3S) tool is an implementation of Simulation Language for Alternative Modeling (SLAM®). The following terminology will be used when discussing the design of simulation network:

- Design Entities
  - Represent units of traffic flowing through the system (e.g. aircraft, event, etc.)
  - Attributes are numerical values carried along with an entity to specify its characteristics (e.g. range, speed, payload, etc.)
- Design Entities
  - Activities (branches) represent time delays in entity movement (e.g. ingress, egress, etc.)
- Nodes
  - Nodes mark the start and finish of each activity (e.g. entity creation, statistic collection, etc.)
  - Used to modify entity flow and the values of the entity and network global attributes (attribute assignment, etc.)

Document Legend

- **Node names appear in black/bold**
- **Global attributes appear in blue/bold**
- **Event entity local attributes appear in green/bold**
- **Air vehicle entity local attributes appear in red/bold**

Trademark Acknowledgements

Operational Strategy and Top-Level Overview

An operational strategy defines the proposed activities of rotorcraft performing missions during post-disaster operations. This strategy defines how each type of aircraft will be used and the missions to be performed. Four types of missions will be modeled:

- MedEvac (non-ambulatory)
- Evacuation (ambulatory)
- Search and Rescue (SAR)
- Supply Transport

The post-disaster strategy modeled contains five mission locations. These locations define departure and arrival points for aircraft and are as follows:

- **Remote Base**: The initial location of rotorcraft. A safe distance from disaster site where assets are initially based. (e.g. DFW Airport)
- **Remote Site**: The final destination for evacuees; refueling available. (e.g. Houston Astrodome)

- **Local Base**: The location of rotorcraft within the disaster area; refueling available. (e.g. New Orleans NAS)
- **Local Site**: The intermediate staging location for evacuees or cargo drop-off site. (e.g. New Orleans International Airport)
- **Event Site**: The location of an event (e.g. evacuation pick-up, rescue, cargo pick-up)

Figure 1 shows an example of the relationship between the various locations in an evacuation mission:

1. At time = 0, all rotorcraft will depart their remote base and fly to an available local base. Event creation is also started at time = 0, therefore events may already be queued for rotorcraft asset assignment.
2. A rotorcraft assigned to a mission will egress to the local event site.
3. If the rotorcraft's performance characteristics allow it to take the evacuees straight to the remote site (3a), then this would be the optimal solution. However, most conventional rotorcraft will only be able to return the evacuees to the local site (3b).
4. If the rotorcraft needs more fuel before the next mission, it will return to a local base.
5. If the rotorcraft has enough fuel to perform another mission, it will egress to the next event site.
6. Evacuees that accumulate at the local site will be transferred via a fixed wing aircraft to the remote site.
7. The fixed wing aircraft will return to pickup additional evacuees

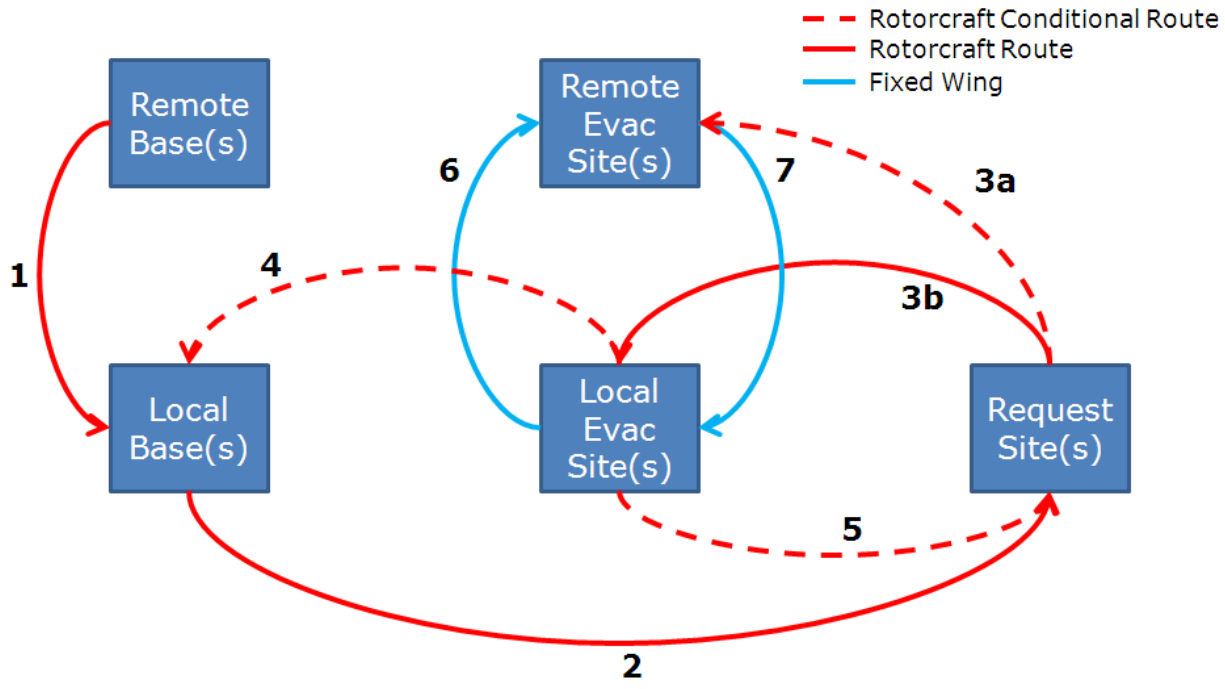


Figure 1. Operational Strategy

The simulation network (Figure 2) will create both vehicle and event entities which will enter queues in the command and control function. All vehicles will originate at a remote base outside the "area of operation". The area of operation will be defined as the area that contains local bases, local sites and location of events. The command and control function, a specialized super-node, will then merge vehicles (see Fleet Creation and Initialization) and events (see Event Creation and Initialization) into a single "mission" entity to be processed. A determination

will be made whether the mission continues on to the remote site or the local site. If the local site is the destination, then the people will enter a queue waiting to be transferred via a fixed wing aircraft to the remote site. Once the people arrive at the final destination, the response time measure of effectiveness can be computed.

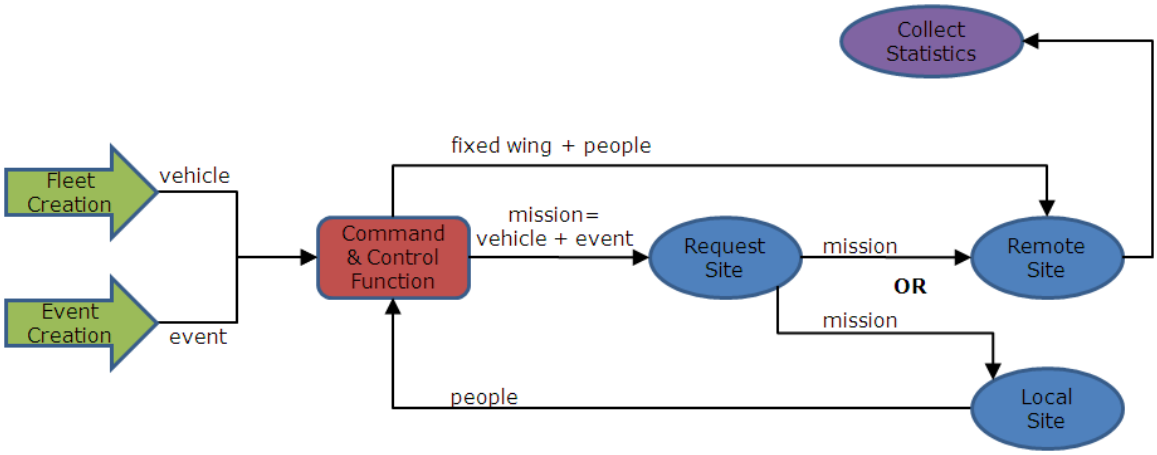


Figure 2. Top-Level Simulation Model

The remainder of this document will discuss the network in detail, logically separated in the following sections:

- Fleet Creation and Initialization
- Event Creation and Initialization
- Command and Control Function
- Evacuation Event
- MedEvac Event
- Search and Rescue (SAR) Event
- Transport Event
- Transfer People from Local to Remote

Fleet Creation and Initialization

The entire fleet mix for the simulation will be created at time zero. The analysts will have the capability to define the fleet mix and attributes for each of the following air vehicle types:

- Light Conventional Rotorcraft (CRC)
- Medium CRC
- Heavy CRC
- 10-passenger Civil Tiltrotor (CTR)
- 30-passenger CTR
- 120-passenger CTR
- Fixed Wing (FW) aircraft (for transfer of people from local to remote sites only)

Figure 3 shows the nodes in the network that creates and initializes the fleet. There is no time delay in any of the activity branches, thus allowing the fleet to be initialized at the start of the simulation.

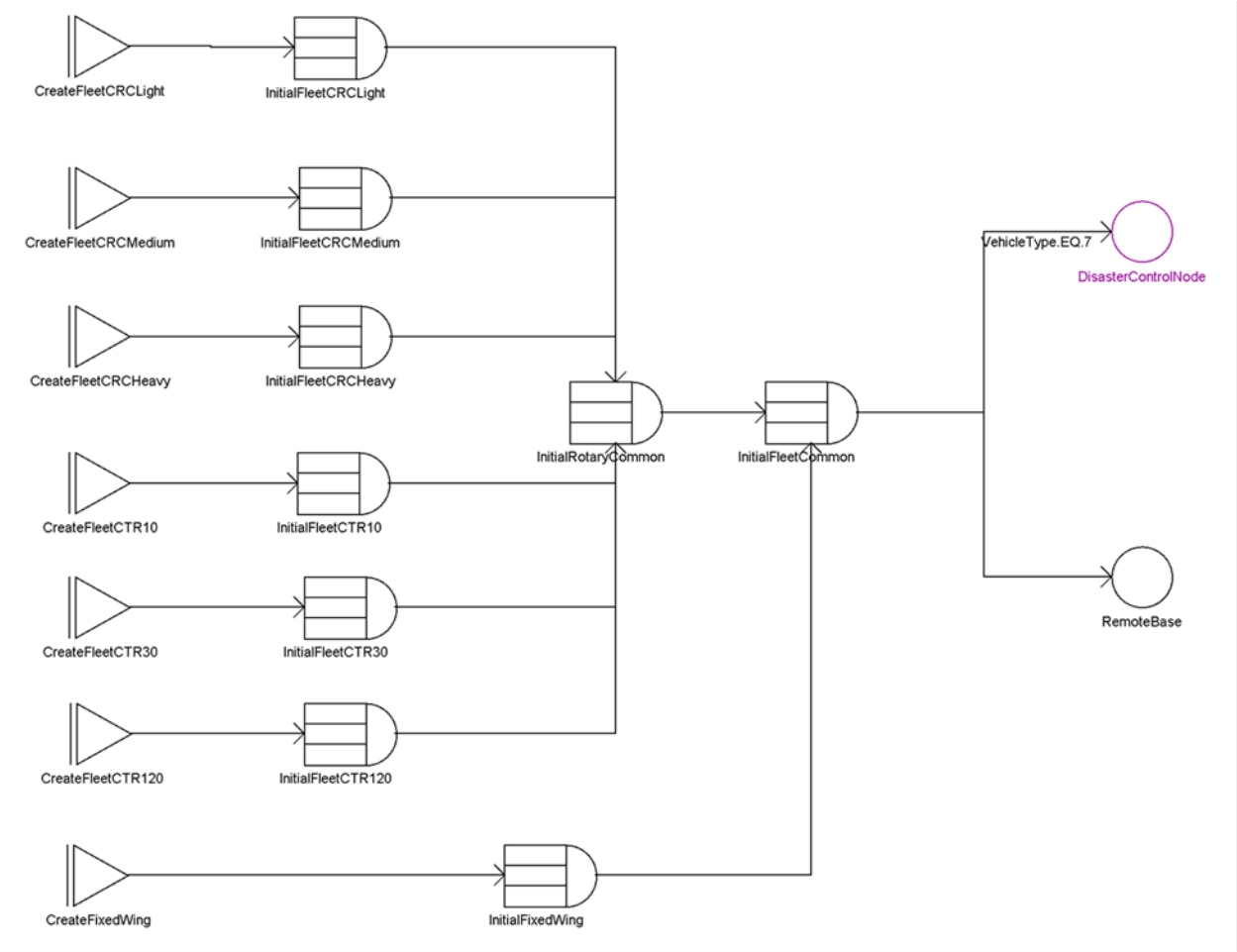


Figure 3. Fleet Creation and Initialization

The simulation methodology will use distances to determine air vehicle sortie time. Instead of using fuel burn calculations which would also require fuel capacity, power settings, altitude, etc.; the simulation will use mission range and speed to determine the total mission time and distance an air vehicle type can achieve.

Fleet Creation Nodes

The following global attributes will determine the size of the fleet:

- **FleetSizeCRCLight**: Number of *light conventional helicopters* to be used in the scenario
- **FleetSizeCRCMedium**: Number of *medium conventional helicopters* to be used in the scenario
- **FleetSizeCRCHeavy**: Number of *heavy conventional helicopters* to be used in the scenario
- **FleetSizeCTR10**: Number of *10 passenger CTRs* to be used in the scenario
- **FleetSizeCTR30**: Number of *30 passenger CTRs* to be used in the scenario
- **FleetSizeCTR120**: Number of *120 passenger CTRs* to be used in the scenario
- **FleetSizeFixedWing**: Number of *fixed wing aircraft* to be used in the scenario

The following creation nodes will create vehicle entities:

- **CreateFleetCRCLight**: Create **FleetSizeCRCLight** vehicle entities

- **CreateFleetCRCMedium**: Create **FleetSizeCRCMedium** vehicle entities
- **CreateFleetCRCHeavy**: Create **FleetSizeCRCHeavy** vehicle entities
- **CreateFleetCTR10**: Create **FleetSizeCTR10** vehicle entities
- **CreateFleetCTR30**: Create **FleetSizeCTR30** vehicle entities
- **CreateFleetCTR120**: Create **FleetSizeCTR120** vehicle entities
- **CreateFixedWing**: Create **FleetSizeFixedWing** vehicle entities

All vehicle entities will be created at t = 0 at a remote base a random distance from the disaster area with the following local attributes:

- **VehicleType**: 1=Light, 2=Medium, 3=Heavy, 4=CTR10, 5=CTR30, 6=CTR120,7=Fixed
- **Speed**: Average vehicle airspeed (knots)
- **Payload**: Maximum load (weight) for supply transport this vehicle can carry (lbs)
- **PaxCapacity**: Maximum passenger capacity
- **LitterCapacity**: Maximum litter capacity
- **MissionRange**: Distance this vehicle can travel on one tank of fuel (nmi)
- **TotalDistanceFlown**: Total distance flown for all missions (nmi)
- **MissionsCompleted**: Total number of missions completed
- **DistanceFromRemoteBase**: This is the randomly created initial range the vehicle must travel from a remote base to a local base (nmi)
- **MissionRangeLeft**: Range left on this tank of fuel (nmi)
- **DistanceToTravel**: Current distance to travel (nmi)
- **CurrentLocation**: 0=Local Base, 1=Local Site, 2=Remote Site
- **TailNumber**: Vehicle unique identifier

Note: To ensure that all the vehicles of a particular type are created in order (i.e. by tail number), each creation node in Figure 3 *will be offset by 0.01 hrs. This ensures the statistical print out of all vehicles in a nice logical order. For example,* ***CreateFleetCRCLight*** *will create vehicles at 0.01 hours,* ***CreateFleetCRCMedium*** *at 0.02 hours and so on.*

Fleet Assignment Nodes

The following specific fleet assignment nodes will populate the attributes; **VehicleType**, **Speed**, **Payload**, **PaxCapacity**, **LitterCapacity**, and **MissionRange** using global attributes:

- **InitialFleetCRCLight**: Uses global attributes **CRCLightSpeed, CRCLightPayload, CRCLightPaxCapacity, CRCLightLitterCapacity, CRCLightMissionRange**
- **InitialFleetCRCMedium**: Uses global attributes **CRCMediumSpeed, CRCMediumPayload, CRCMediumPaxCapacity, CRCMediumLitterCapacity, CRCMediumMissionRange**
- **InitialFleetCRCHeavy**: Uses global attributes **CRCHeavySpeed, CRCHeavyPayload, CRCHeavyPaxCapacity, CRCMediumLitterCapacity, CRCHeavyMissionRange**
- **InitialFleetCTR10**: Uses global attributes **CTR10Speed, CTR10Payload, CTR10PaxCapacity, CTR10LitterCapacity, CTR10MissionRange**
- **InitialFleetCTR30**: Uses global attributes **CTR30Speed, CTR30Payload, CTR30PaxCapacity, CTR30LitterCapacity, CTR30MissionRange**
- **InitialFleetCTR120**: Uses global attributes **CTR120Speed, CTR120Payload, CTR120PaxCapcity, CTR120LitterCapacity, CTR120MissionRange**

- **InitialFixedWing**: Uses global attributes **FixedWingSpeed** and **FixedWingPaxCapacity**. The local attributes Payload, LitterCapacity, and MissionRange are not used for the fixed wing vehicles.

The rotary common assignment node, **InitialRotaryCommon**, will populate the attributes:

- **DistanceFromRemoteBase**: using a uniform distribution between the global attributes **DistanceFromRemoteBaseMinimum** and **DistanceFromRemoteBaseMaximum**
- **DistanceToTravel**: Set to **DistanceFromRemoteBase**

The common fleet assignment node, **InitialFleetCommon**, will populate the attributes:

- **TotalDistanceFlown**: 0
- **MissionsCompleted**: 0
- **MissionRangeLeft**: Set to **MissionRange**
- **CurrentLocation**: 0
- Count: Count + 1
- **TailNumber**: Set to Count

Fleet Build Up

The simulation will model the initial fleet build up. Each air vehicle will be assigned a random distance from the area of operation to simulate the initial air vehicle location at a remote base (except for fixed wing vehicles which will reside locally). If the vehicle cannot traverse the distance on a single tank of fuel, then a refueling time delay will be modeled. Figure 4 shows the network flow for the fleet build up after initialization.

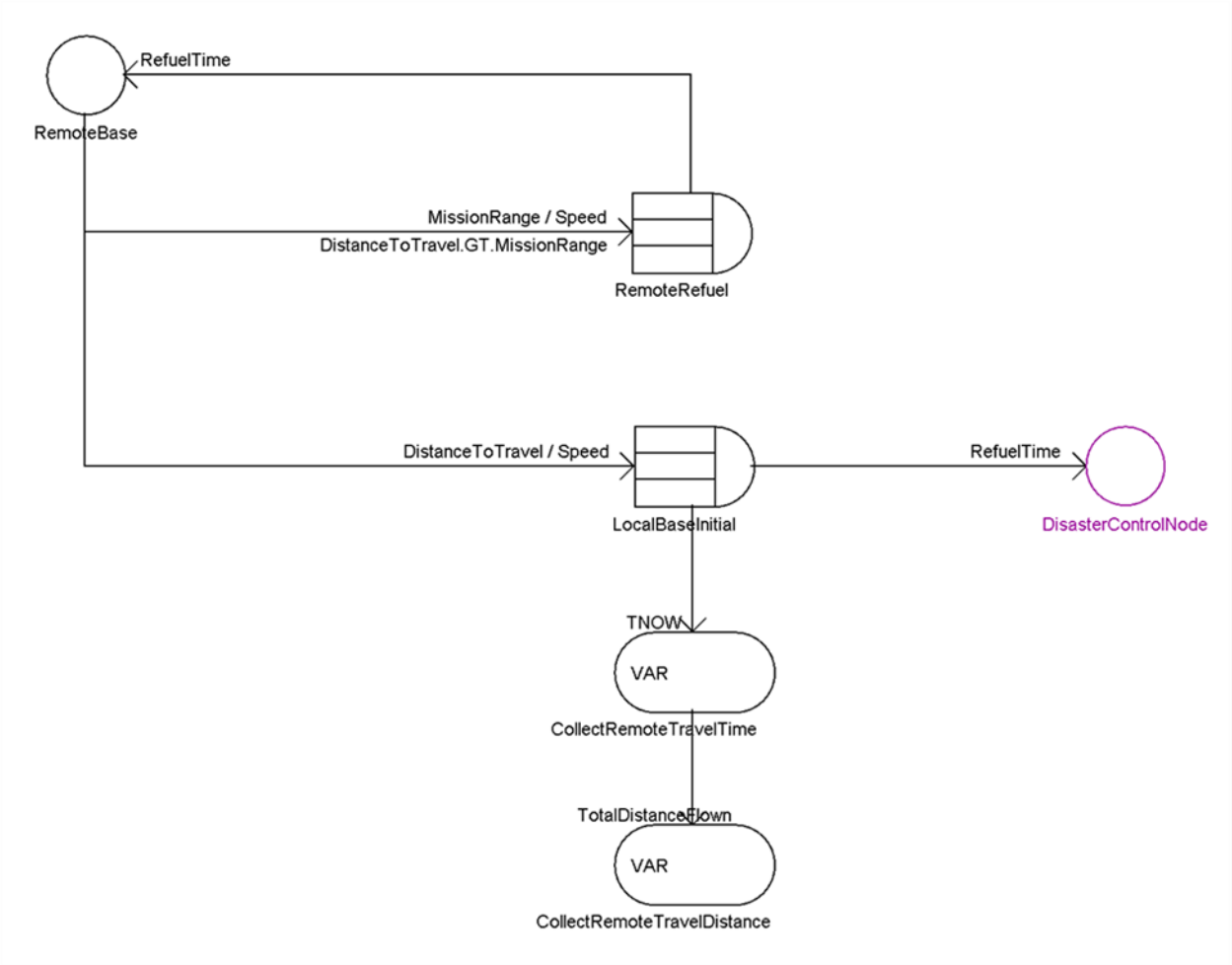


Figure 4. Fleet Build-Up from Remote Bases

Note: As seen in Figure 3*, the air vehicles of type equal 7 (fixed wing) flows directly to the command and control function queue. The fixed wing quantity and attributes will be the same for all fleet mixes to prevent influencing the MOE of initial time build up for the rotary wing fleet. Hence, the fixed wing vehicles will not be included in the initial build segment of the network.*

If the distance to travel from remote to the local base is greater than the mission range, then the air vehicle will stop to refuel. The **RemoteRefuel** node updates the vehicle attributes as follows:

- **TotalDistanceFlown**=**TotalDistanceFlown**+**MissionRange**
- **DistanceToTravel**=**DistanceToTravel**-**MissionRange**

The air vehicle will continue to stop and refuel as long as the current distance to travel is greater than the mission range of the vehicle. Once the vehicle arrives at the local base, the **LocalBaseInitial** node updates the vehicle attributes as follows:

- **TotalDistanceFlown**=**TotalDistanceFlown**+**DistanceToTravel**

Event Creation and Initialization

The simulation will model four major event types: Evacuation, MedEvac, Search and Rescue (SAR) and Transport (Figure 5). These events will result in requests for people to be recovered or supplies to be transported. In either case, the number of people or the amount of supplies may need to be broken into smaller events (missions) to be handled by more than one air vehicle. Depending upon the fleet mix, recovered people may build up at a local site. The objective is to move all people to a remote site; so the simulation will be required to generate internal "move" events to handle the evacuation of these people from a local site to a remote site. These move events are currently handled by the fixed wing aircraft only.

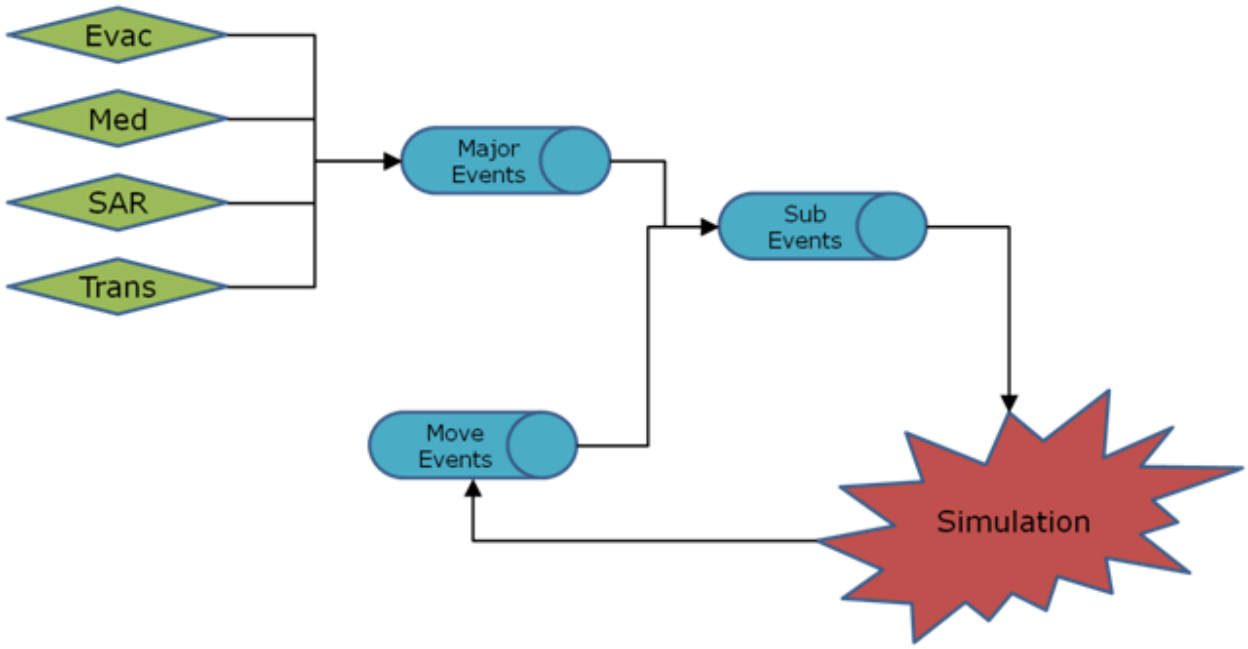
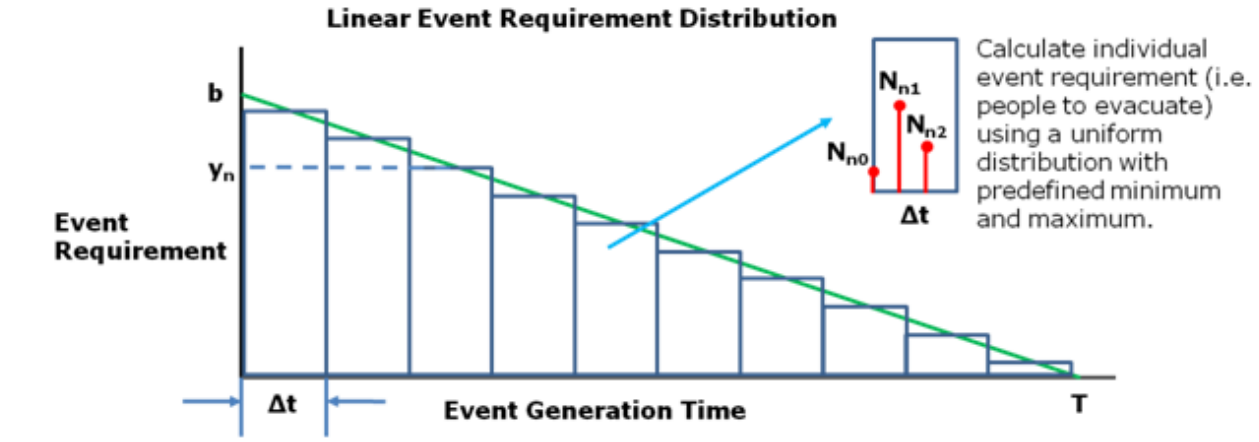


Figure 5. High Level Event Creation

Major Events

In any disaster scenario, most of the events would occur immediately afterwards. Therefore the event creation methodology (Figure 6) will use a linear ramp down distribution for the major event generation during the entire simulation time. The area (A) under the curve represents the total number of people to be recovered (or pounds of supplies to be transported). For a given Δt (e.g. 6 hours), this area is divided into rectangles where the number of people (or pounds) is determined for that particular time period. Then using a uniform distribution, events are generated until the computed value is reached. The events are then uniformly spaced within that time period. Therefore, there may be a few large events during one time period and many smaller events during the next time period.



$A = (b \times T) / 2; A$: Area

$b = (2 \times A) / T; b$: y intercept; T: event generation end time

$m = -b / T; m$: slope

$y_n = m \times [n \times \Delta t + \Delta t / 2] + b; n = 0, T / \Delta t - 1; y$: event requirement at time $t; \Delta t$: event requirement time duration

$N_n = y_n \times \Delta t; N$: individual mission requirement for Δt

Figure 6. Event Creation Methodology

The following global attributes are associated with creating events:

- **EvacuationTotalPeople**: Total number of people to be evacuated over the duration of the simulation
- **EvacuationPeopleMinimum**: Minimum number of people to be evacuated per event
- **EvacuationPeopleMaximum**: Maximum number of people to be evacuated per event
- **EvacuationTimeBetweenEvents**: Time between evacuation events (hr)
- **MedevacTotalPeople**: Total number of people to MedEvac over the duration of the simulation
- **MedevacPeopleMinimum**: Minimum number of people to be recovered per MedEvac event
- **MedevacPeopleMaximum**: Maximum number of people to be recovered per MedEvac event
- **MedevacTimeBetweenEvents**: Time between MedEvac events (hr)
- **SARTotalPeople**: Total number of people to search for over the duration of the simulation
- **SARPeopleMinimum**: Minimum number of people to be recovered per SAR event
- **SARPeopleMaximum**: Maximum number of people to be recovered per SAR event
- **SARSearchRangeMinimum**: Minimum range vehicle will travel searching before finding people (nmi)
- **SARSearchRangeMaximum**: Maximum range vehicle will travel searching before finding people (nmi)
- **SARTimeBetweenEvents**: Time between SAR events (hr)
- **TransportTotalPayload**: Total quantity of payload to be transported over the duration of the simulation (lb)
- **TransportPayloadMinimum**: Minimum weight of supplies per transport event

- **TransportPayloadMaximum**: Maximum weight of supplies per transport event
- **TransportTimeBetweenEvents**: Time between transport events (hr)

Figure 7 shows the nodes in the network that create and initialize the events. All of the events are funneled into the command and control function.

Event Creation Nodes

The following creation nodes will create event entities:

- **CreateEvacuation**: Evacuation events
- **CreateMedevac**: MedEvac events
- **CreateSAR**: SAR events
- **CreateTransport**: Transport events

The create node uses two interfaces (NextTime and NextEntity) to create the next entity at a given time. A special Java class (EventNextTime) was created which implements these two interfaces and performs the methodology described above (See Appendix D). The attribute, **EventRequirement**, is generated for the event created.

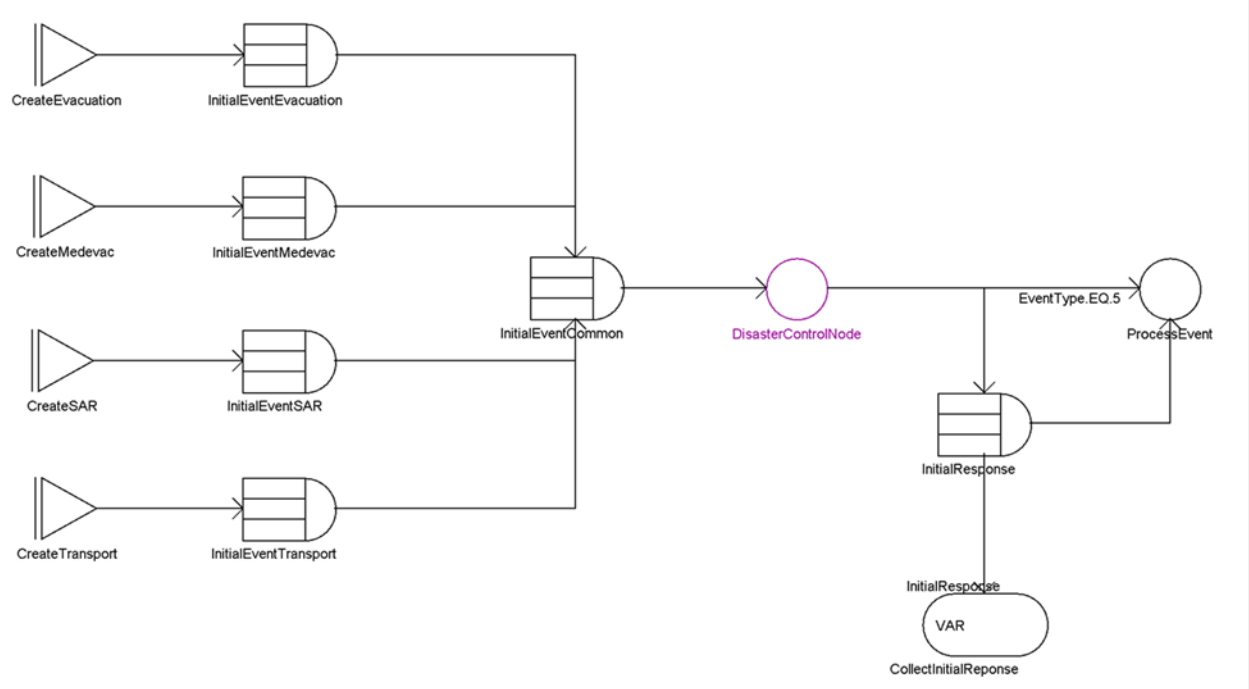


Figure 7. Event Creation and Initialization

Event Assignment Nodes

The event assignment nodes initialize the specific events:

- **InitialEventEvacuation**
  - **EventType**: 1
  - **Priority**: 2
  - **PeopleToBeSaved**: EventRequirement
  - **Ambulatory**: 1

- **InitialEventMedevac**
  - **EventType**: 2
  - **Priority**: 1
  - **PeopleToBeSaved**: EventRequirement
  - **Ambulatory**: 0
- **InitialEventSAR**
  - EventType: 3
  - Priority: 3
  - PeopleToBeSaved: EventRequirement
  - SearchRangeToFind: Uniform distribution based on SARSearchRangeMinimum and SARSearchRangeMaximum
  - **Ambulatory**: 1
- **InitialEventTransport**
  - **EventType**: 4
  - **Priority**: 4
  - **PayLoadToTransport**: EventRequirement
- **InitialEventCommon**: This node initializes the four attributes discussed in the section "Bases and Sites":
  - **DistanceLocalToEvent:** Distance from local base (or site) to the event computed using a uniform distribution based on **DistanceBetweenLocalAndEventMinimum** and **DistanceBetweenLocalAndEventMaximum**
  - **DistanceRemoteToEvent:** Distance from remote site to the event computed using a uniform distribution based on **DistanceBetweenRemoteAndEventMinimum** and **DistanceBetweenRemoteAndEventMaximum**
  - **DistanceEventToLocal:** Distance from event to local site computed using a uniform distribution based on **DistanceBetweenLocalAndEventMinimum** and **DistanceBetweenLocalAndEventMaximum**
  - **DistanceEventToRemote:** Distance from event to remote site computed using a uniform distribution based on **DistanceBetweenRemoteAndEventMinimum** and **DistanceBetweenRemoteAndEventMaximum**
  - **TimeCreated:** TNOW (current simulation time)

Note: To ensure that the same events are generated regardless of the fleet mix, all the uniform distributions in the event creation will use the CTRRand function which will use its unique, initial random number seed.

Command and Control Function

The command and control function consists of the following lists and queues (Figure 8):

- **Vehicle List**: Contains the fleet mix of rotary air vehicles.
- **Event Queue**: Contains list of events (Evacuation, MedEvac, SAR, and Transport). These events are ordered first by event priority and then by time.
- **Fixed Wing List**: Contains the list of fixed wing air vehicles. Fixed wing vehicles are only used to move/transfer people from a local site to a remote site.

- **Local People Queue**: Contains a list of people transported from an evacuation, MedEvac or SAR event but not taken to a remote site. The order of the queue is FIFO.

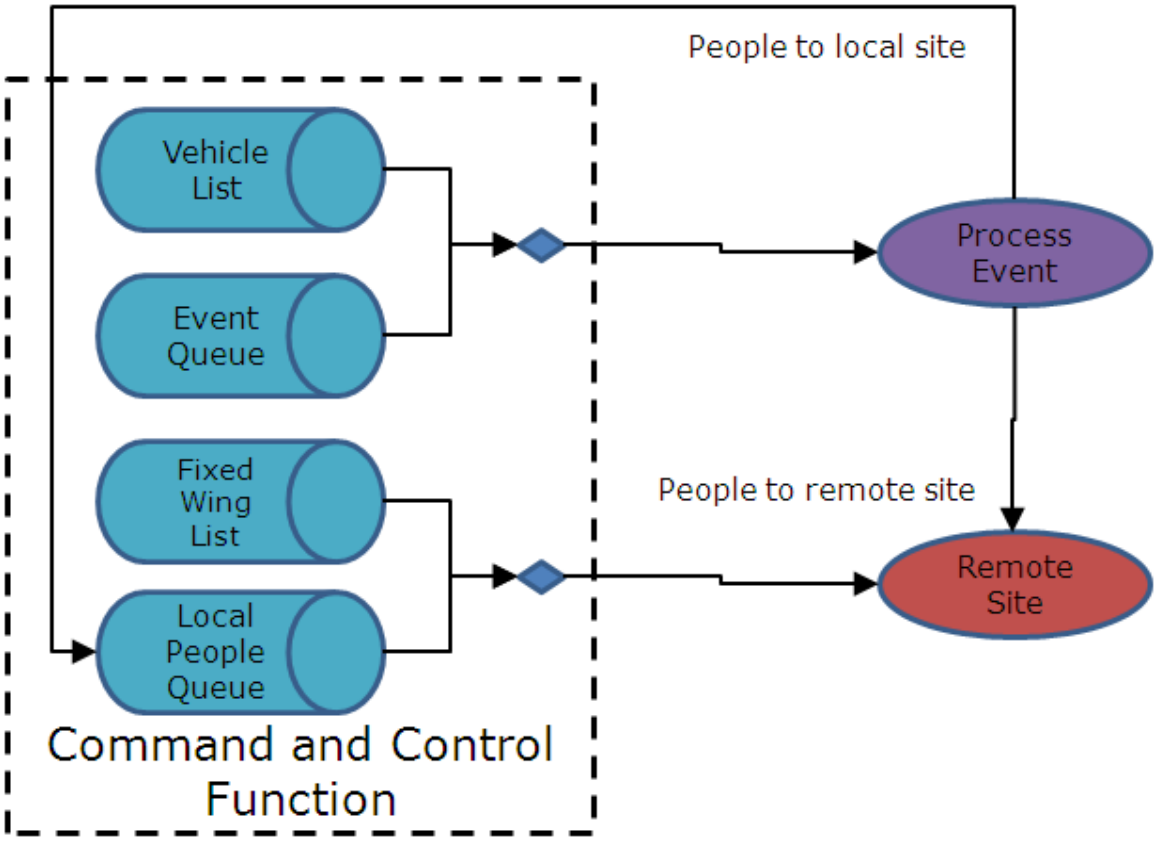


Figure 8. Command and Control Function

The command and control function performs two major tasks by merging a vehicle list and queue:

- **Missions**: The vehicle list and event queue are used to generate missions. As events and/or vehicles enter the command and control function, the controller determines if any vehicles, currently in the list, satisfy the requirements for any of the events. If the event involves people (Evacuation or SAR), a determination is made on whether the people are transported to a local site or to a remote site. If the people are transported to a local site, then the people enter the local people queue.
- **People Transfer Event**: The fixed wing list and local people queue are used to generate people transfer events. If the number of people in the queue reaches a certain threshold and at least one fixed wing vehicle is available, then a fixed wing vehicle is loaded to transfer the people to a remote site. The EventType is 5.

Each event generated (Evacuation, MedEvac, SAR, and Transport) may be too large for one air vehicle (.e.g. evacuate 1,000 people). In general, the command and control function first sorts all the available air vehicles by the amount of mission range (or fuel) remaining. When more than one vehicle is available to transport the remaining people (or supplies), the controller chooses the vehicle with the closest capacity (passenger or useful load) but enough capacity to perform the mission. For example, if 160 people are to be evacuated and CTR-120 is available it would be sent on a mission leaving 40 remaining. If a rotorcraft of each class is still available;

then the CRC-Heavy would be assigned to the final 40 evacuees as it is the closest and has enough capacity (43 passenger capacity) to evacuate the remaining people.

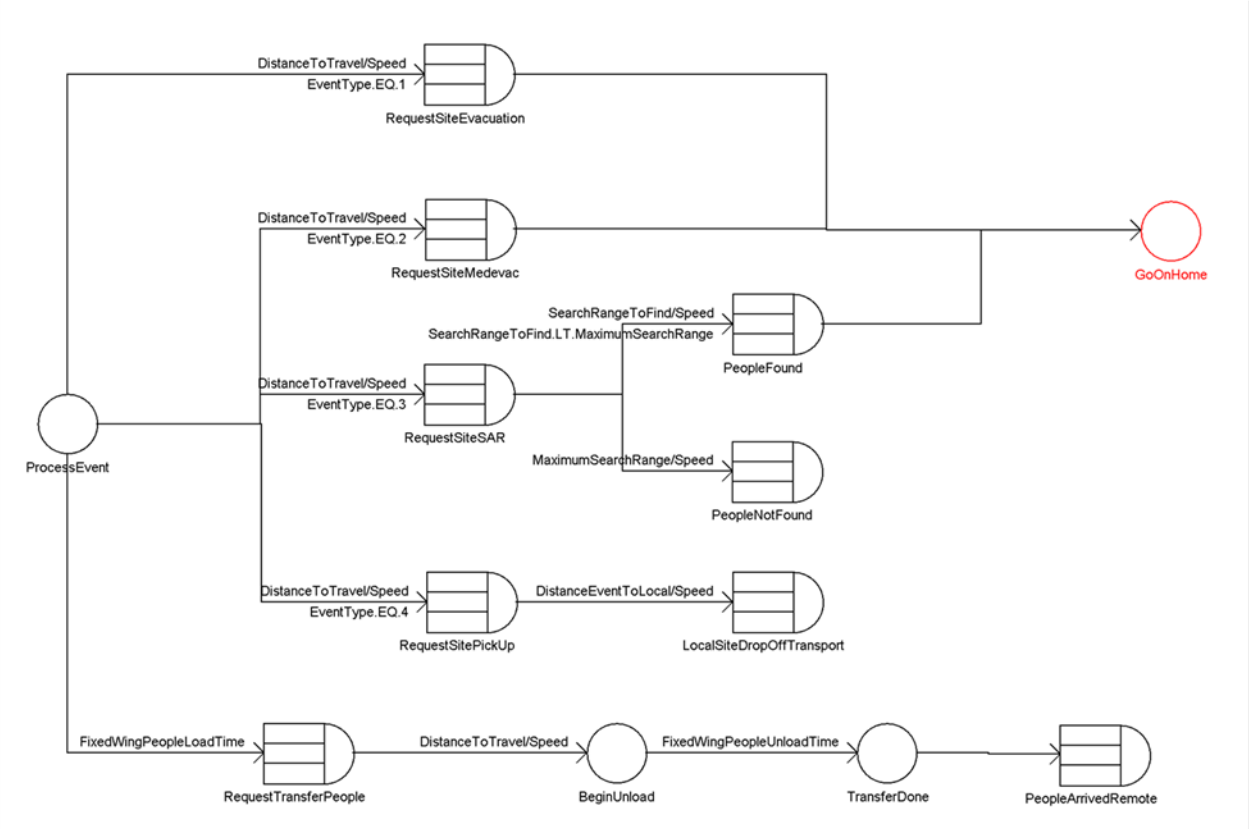


Figure 9. Process Events

The remainder of this section discusses steps the command and control function takes when processing the various event types.

Search and Rescue (SAR)

1. The current available vehicle list is reduced to a list of vehicles that satisfy the SAR threshold capacity, air vehicles with passenger capacity greater than this threshold are not considered for SAR missions. *Note: It is assumed that the distribution used to generate the event will not generate a number greater than this threshold.*
2. This reduced list is then sorted by time on station (TOS) remaining (or mission range left). *Note: An air vehicle may have performed a short mission without refueling.*
3. The air vehicle with the most TOS remaining (or mission range left in our case) must have a mission range left greater than the SAR range threshold; otherwise no air vehicles can currently perform this mission.
4. The attribute, **MaximumSearchRange**, is set to the air vehicles remaining TOS. This attribute is compared to the randomly created search time to determine whether the air vehicle finds the people or not.

Evacuation

1. First, it is determined if any air vehicles can take the evacuees straight to the remote. If that is the case, air vehicles that meet this requirement are then sorted by passenger capacity.
2. The command and control function makes assignments until the event requirement (total evacuees) is met or aircraft are unavailable.
3. Of the remaining air vehicles, it is determined if any can make it to the event, back to the local site and refuel. The air vehicles that satisfy this requirement are then sorted by passenger capacity.
4. Step 2 is then repeated for these vehicles.

MedEvac

1. Similar to Evacuation logic, except these events always transport people to the local site, the vehicle list is reduced to those air vehicles that can perform the mission.
2. The reduced list is sorted by litter capacity.
3. The command and control function makes assignments until the event requirement is met or air vehicles are unavailable.

Transport

1. The air vehicle list is reduced to those air vehicles which can perform the mission.
2. The reduced list is sorted by payload size.
3. The command and control function makes assignments until the event requirement (total payload) is met or air vehicles are unavailable.

Note: If the command and control function runs out of air vehicles before the event requirement is met then the event is placed back in the event queue with the reduced event requirement. For example, if there was an evacuation event for 450 people and after assignments were made there were still 65 people left from this event, then the event would still exist in the queue but with only a 65 people requirement.

Bases and Sites

Defining actual physical site locations could skew the simulation results (Figure 10); therefore, the simulation is based entirely on the random generation of distances. Because local bases and a single local site would be randomly placed in the area of operation, the distance to an event will be the same random variable whether the vehicle is at a local base or local site. Therefore, each event will have four distances generated. The distances to and from the event may be different as the (local or remote) site they are returning to may be different than the one from which they originated. All of these will be randomized using uniform distributions (Figure 11):

- **DistanceLocalToEvent**: Distance from local base (or site) to the event
- **DistanceRemoteToEvent**: Distance from remote site to the event is needed in case the vehicle originates from a remote site
- **DistanceEventToLocal**: Distance from event to local site
- **DistanceEventToRemote**: Distance from event to remote site

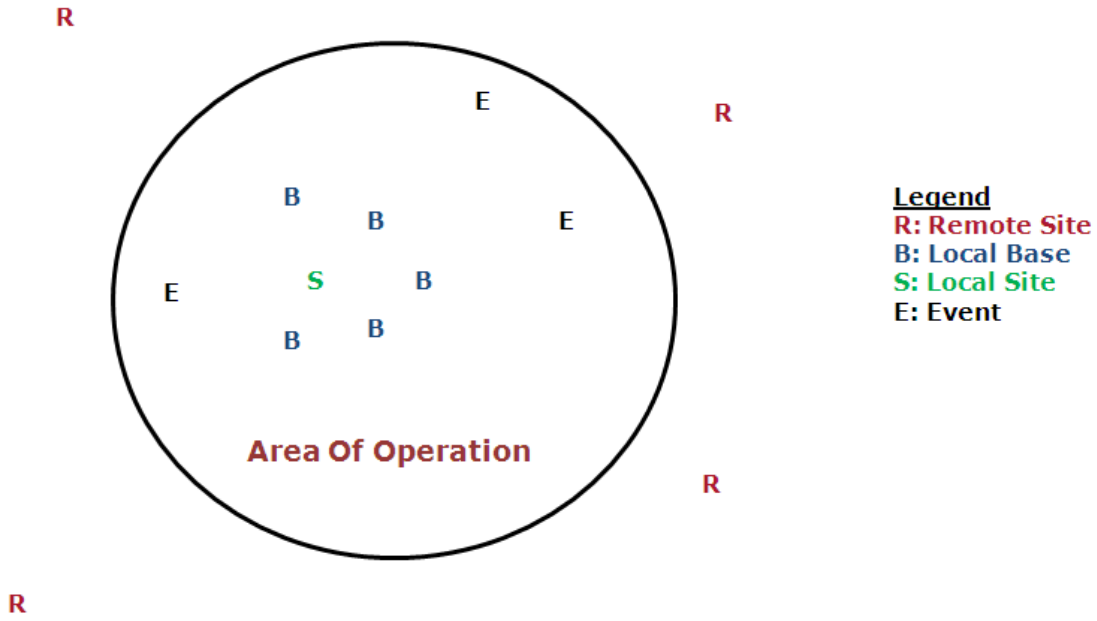


Figure 10. Area of Operation (AOO)

Figure 12 shows the various scenarios that can occur throughout the simulation depending upon the air vehicles current (starting) location, the mission, and the characteristics of the air vehicle.

- **Scenario 1**: The air vehicle originates from a local base, transits to the event, and back to the local site.
- **Scenario 2**: The air vehicle originates from the local site, transits to the event, and back to the local site.
- **Scenario 3**: The air vehicle originates from the local base, transits to the event, and to a remote site (SAR and Evacuation missions only),
- **Scenario 4**: The air vehicle originates from the local site, transits to the event, and to a remote site (SAR and Evacuation missions only).
- **Scenario 5**: The air vehicle originates from a local base, transits to the event, and returns to a local base (SAR people not found mission only).
- **Scenario 6**: The air vehicle originates from a remote site, transits to the event, and to the local site.
- **Scenario 7**: The air vehicle originates from a remote site, transits to the event, and back to a remote site (SAR and Evacuation missions only).

Another uniform distribution will exist to determine the distance between the local site and local base when a vehicle needs to refuel. This distribution will use the global attributes:

- **DistanceBetweenBaseAndSiteMinimum**: Minimum distance between a local base and local site (nmi)
- **DistanceBetweenBaseAndSiteMaximum**: Maximum distance between a local base and local site (nmi)

The simulation will keep track of where a vehicle is relative to a local site, local base, or remote site for refueling considerations. Fuel is assumed to be available at remote sites and

local bases but not the local site. A vehicle will only be assigned to an event if it is determined that it has sufficient fuel for the following conditions:

- To ingress to an event and egress to a remote site or
- To ingress to an event, egress to a local site, and have enough fuel to return to local base (**DistanceBetweenBaseAndSiteMaximum**)

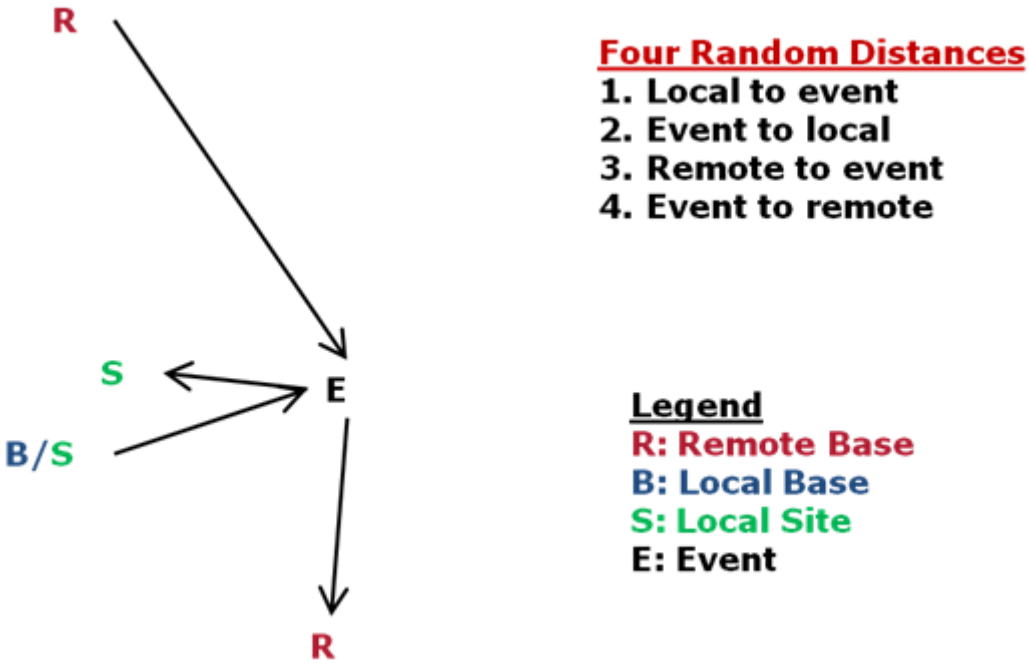


Figure 11. Event Distances

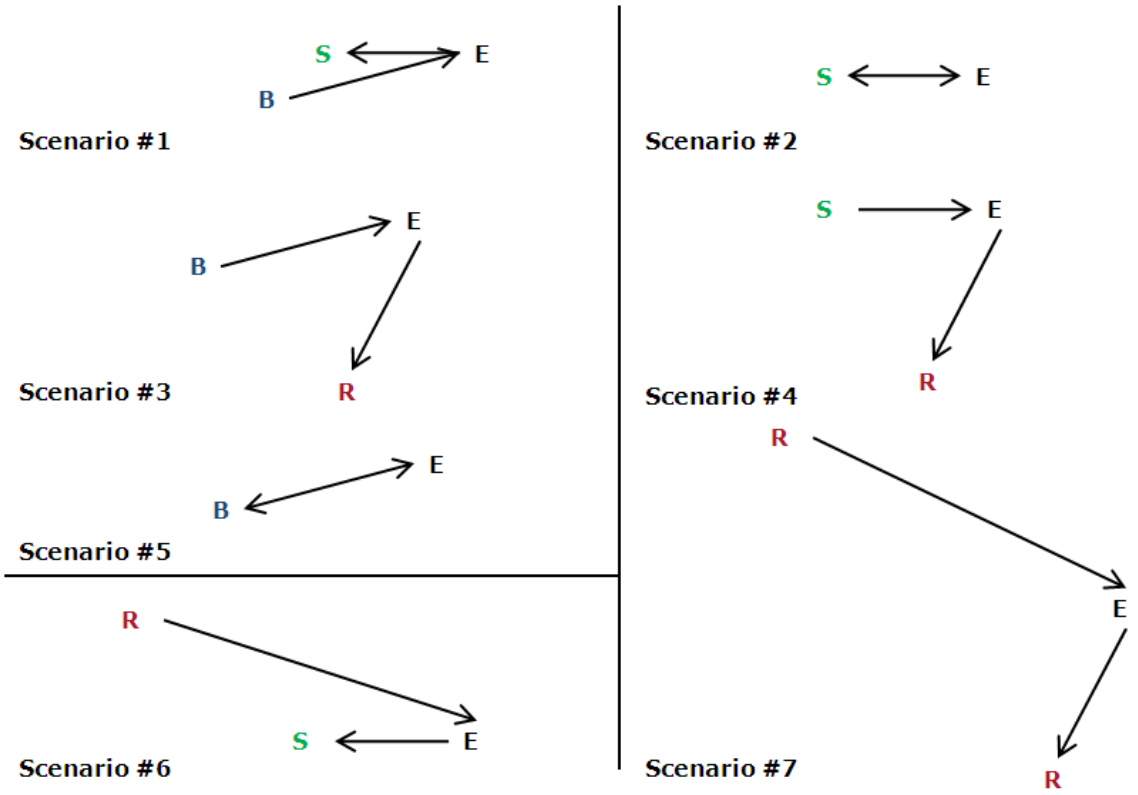


Figure 12. Event Scenarios

Refueling Time

During the simulation, a single turnaround time will be used for a delay in refueling defined by the global attribute **RefuelTime**.

Evacuation Event

Once the command and control function merges a vehicle with an evacuation event, the combined entity (mission) is pushed out of the queue on to the **RequestSiteEvacuation** node (Figure 13). The command and control function will set the **DistanceToTravel** attribute to **DistanceLocalToEvent** or **DistanceRemoteToEvent** depending upon the current location of the air vehicle. The duration time is then **DistanceToTravel** / **Speed**.

RequestSiteEvacuation Assignment Node

This node updates mission range left:

1. MissionRangeLeft = MissionRangeLeft-DistanceToTravel

Two activities (branches) emanate from this node. Three conditions must be met for the branch to evacuate people directly to remote site is to be taken:

1. The **MissionRangeLeft** must be greater than **DistanceEventToRemote** then, otherwise go to local site.
2. Ambulatory must be true (which it is for an evacuation event/mission)
3. The **PercentCapacity** attribute must be greater than the **RemoteThreshold**.

LocalSiteDropOffEvacuation Assignment Node

This node updates the mission range left; number of missions completed, and sets the current location to “Local” site:

1. MissionRangeLeft = MissionRangeLeft - DistanceEventToLocal

2. MissionsCompleted = MissionsCompleted + 1
3. CurrentLocation = 1

Two activities (branches) emanate from this node. The entity will be cloned with one entity proceeding to represent the people that must be still transported to the remote site and the other entity, the air vehicle. The **LocalSite** node is a specialized node which checks to determine if the air vehicle needs to return to the local base to refuel. If so, the **RefuelFlag** is set true and the conditional branch to **LocalBase** is taken.

RemoteSiteDropOffEvacuation Assignment Node

This node updates the mission range left, number of missions, sets the current location to "Remote" site, and refuels the vehicle (which updates total distance flown and resets mission range left):

1. MissionRangeLeft = MissionRangeLeft - DistanceEventToRemote
2. MissionsCompleted = MissionsCompleted + 1
3. CurrentLocation = 2
4. Temp = MissionRange - MissionRangeLeft
5. TotalDistanceFlown = TotalDistanceFlown + Temp
6. MissionRangeLeft = MissionRange

LocalBase Assignment Node

1. MissionRangeLeft = MissionRangeLeft – DistanceToTravel
2. CurrentLocation = 1

Refuel Assignment Node

This node refuels the air vehicle:

1. Temp = MissionRange - MissionRangeLeft
2. TotalDistanceFlown = TotalDistanceFlown + Temp
3. MissionRangeLeft = MissionRange

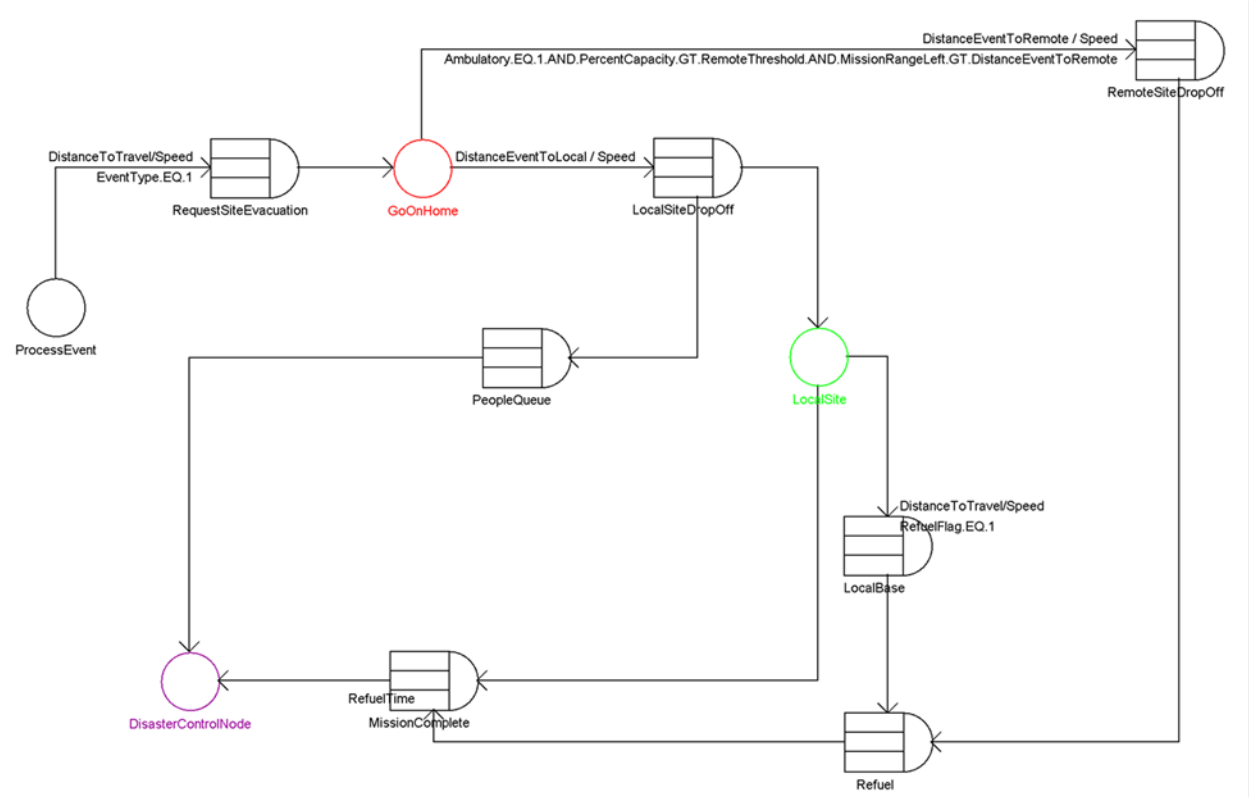


Figure 13. Evacuation Event Network

MedEvac Event

Once the command and control function merges a vehicle with a MedEvac event, the combined entity (mission) is pushed out of the queue on to the **RequestSiteMedevac** node (Figure 14). The command and control function will set the **DistanceToTravel** attribute to **DistanceLocalToEvent** or **DistanceRemoteToEvent** depending upon the current location of the air vehicle. The duration time is then **DistanceToTravel** / **Speed**. The mission will then proceed to the **GoOnHome** node and then follow the same network flow, as seen in Figure 13.

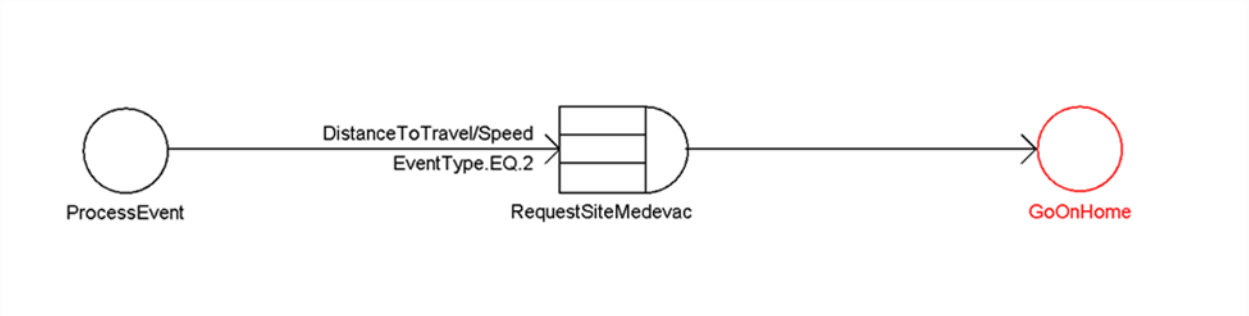


Figure 14. MedEvac Event Network

Search and Rescue (SAR) Event

Once the command and control function merges a vehicle with a SAR event, the combined entity (mission) is pushed out of the queue on to the **RequestSiteSAR** node (Figure 15). The command and control function will set the **DistanceToTravel** attribute to **DistanceLocalToEvent** or **DistanceRemoteToEvent** depending upon the current location of

the air vehicle. The duration time is then **DistanceToTravel** / **Speed**. The SAR part of the network then has a conditional check to determine if the people were found or not. A random search range was created for the event/mission. If the air vehicle's maximum search range is greater than this time, the people were found and the mission entity proceeds to the **GoOnHome** node (Figure 13). If the people were not found, a conditional check is performed to see if the air vehicle will return to the same local base or a different local base.

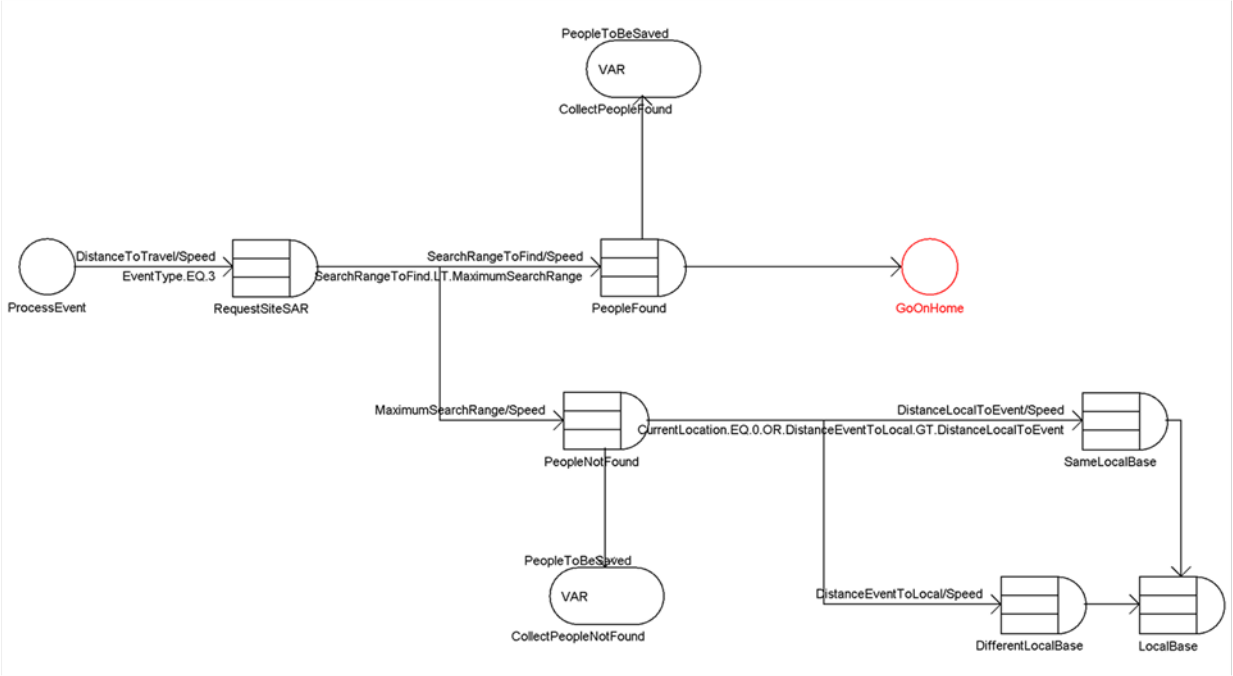


Figure 15. Search and Rescue Event Network

Transport Event

Once the command and control function merges a vehicle with a transport event, the combined entity (mission) is pushed out of the queue on to the **RequestSitePickUp** node (Figure 16). The command and control function will set the **DistanceToTravel** attribute to **DistanceLocalToEvent** or **DistanceRemoteToEvent** depending upon the current location of the air vehicle. The duration time is then **DistanceToTravel** / **Speed**. The mission then proceeds to drop off the cargo and proceed to the **LocalSite** node (Figure 13).

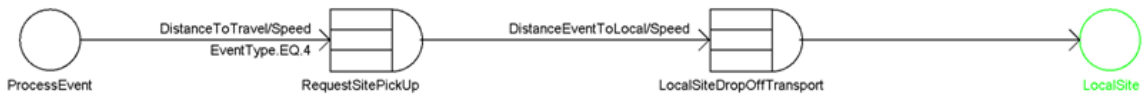


Figure 16. Transport Event Network

Transfer People from Local to Remote

As stated above, the ultimate goal of the evacuation, MedEvac, and SAR events is to transport the people to a remote site. A primary MOE of this goal is the total response time

(Figure 17), that is the time from which the event was generated until the people arrived at the remote site, $t_4 - t_0$. There are intermediate steps in achieving this goal:

- **t_0**: Time the event was created
- **t_1**: Time a sub-event (mission) was created merging a vehicle with an event
- **t_2**: Time the people were found/picked-up
- **t_3**: Time the people were dropped off a local site (optional)
- **t_4**: Time the people arrived at a remote site

The delayed response time, $t_1 - t_0$, represents the time the controller waits for an air vehicle to meet the requirements for the event.

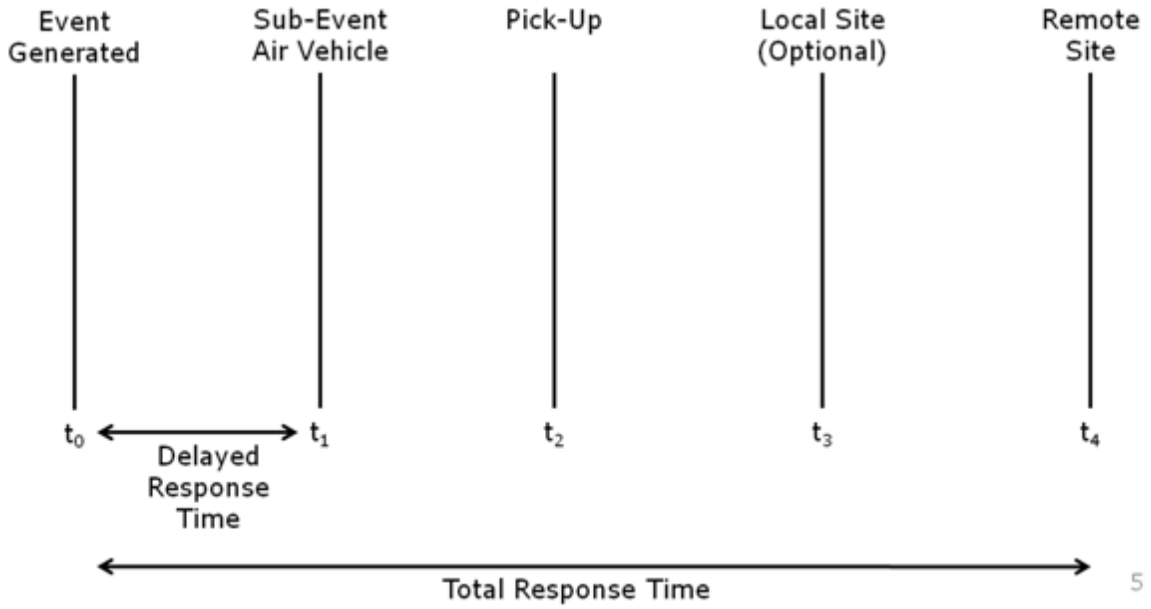


Figure 17. Response Time

Different fleet mixes will result in different missions. For example, an evacuation event of 100 people could result in the following missions, dependent upon the availability of air vehicles (Table 1).

Table 1. Available Fleet

Fleet Mix #1	**Fleet Mix #2**
25 Passenger Vehicle	90 Passenger Vehicle
35 Passenger Vehicle	10 Passenger Vehicle
15 Passenger Vehicle	
25 Passenger Vehicle	

Therefore, the total response time may be different for various groups of people in the initial event. In fact, some response times may be vastly different if one vehicle took passengers to a local site (t_3) while another vehicle was capable of taking passengers directly to the remote site (t_4). To account for this, the network will compute and report an average response time for all people. The following formula will determine the average response time:

$$Average\ Response\ time = \sum_{i=1}^{N} P_i RT_i \Big/ \sum_{i=1}^{N} P_i$$

Where P_i is the number of people in a group of people and RT_i is the response time for this group of people.

Transfer People Network

People arrive at a remote site by different means (Figure 18). They are either brought straight from the event site by a rotary wing vehicle or they are brought from the local site via a fixed wing vehicle. The rotary wing vehicles will flow through the **RemoteSiteDropOff** node, whereas, the fixed wing vehicle will flow through the **RequestTransferPeople** node (Figure 9) and then the **BeginUnload** node. The **RemoteSiteDropOff** node updates the vehicles mission range left and current location:

- **MissionRangeLeft**=**MissionRangeLeft**-**DistanceEventToRemote**
- **CurrentLocation**=2

The fixed wing vehicle mission is cloned with the fixed wing vehicle returning back to a local site via the **FixedWingSendHome** node and the people statistically accounted for in the **PeopleArrivedRemote** node, which makes the following assignments:

- **TotalSavedAmbulatory**=**TotalSavedAmbulatory**+**NumberSavedAmb**
- ResponseTime=TNOW-**TimeCreatedAmb**
- Temp=ResponseTime***NumberSavedAmb**
- **WeightedResponseAmb**=**WeightedResponseAmb**+Temp
- **TotalSavedNonAmbulatory**=**TotalSavedNonAmbulatory**+**NumberSavedNonAmb**
- ResponseTime=TNOW-**TimeCreatedNonAmb**
- Temp=ResponseTime***NumberSavedNonAmb**
- **WeightedResponseNonAmb**=**WeightedResponseNonAmb**+Temp

The average response time for ambulatory and non-ambulatory people is computed in the two nodes:

- **SetAmbulatory**: **AveResponseTimeAmb**=**WeightedResponseAmb**/**TotalSavedAmbulatory**
- **SetNonAmbulatory**: **AveResponseTimeNonAmb**=**WeightedResponseNonAmb**/**TotalSavedNonAmbulatory**

The **RecordMOEs** node takes snap shots of global variables at specified time intervals during the simulation. The following global variables are recorded:

- **TotalMissionsCompleted**
- **TotalPayload**
- **AveResponseTimeTransport**
- **TotalSavedNonAmbulatory**
- **AveResponseTimeNonAmb**
- **TotalSavedAmbulatory**
- **AveResponseTimeAmb**
- **SARPeopleFound**
- **SARPeopleNotFound**

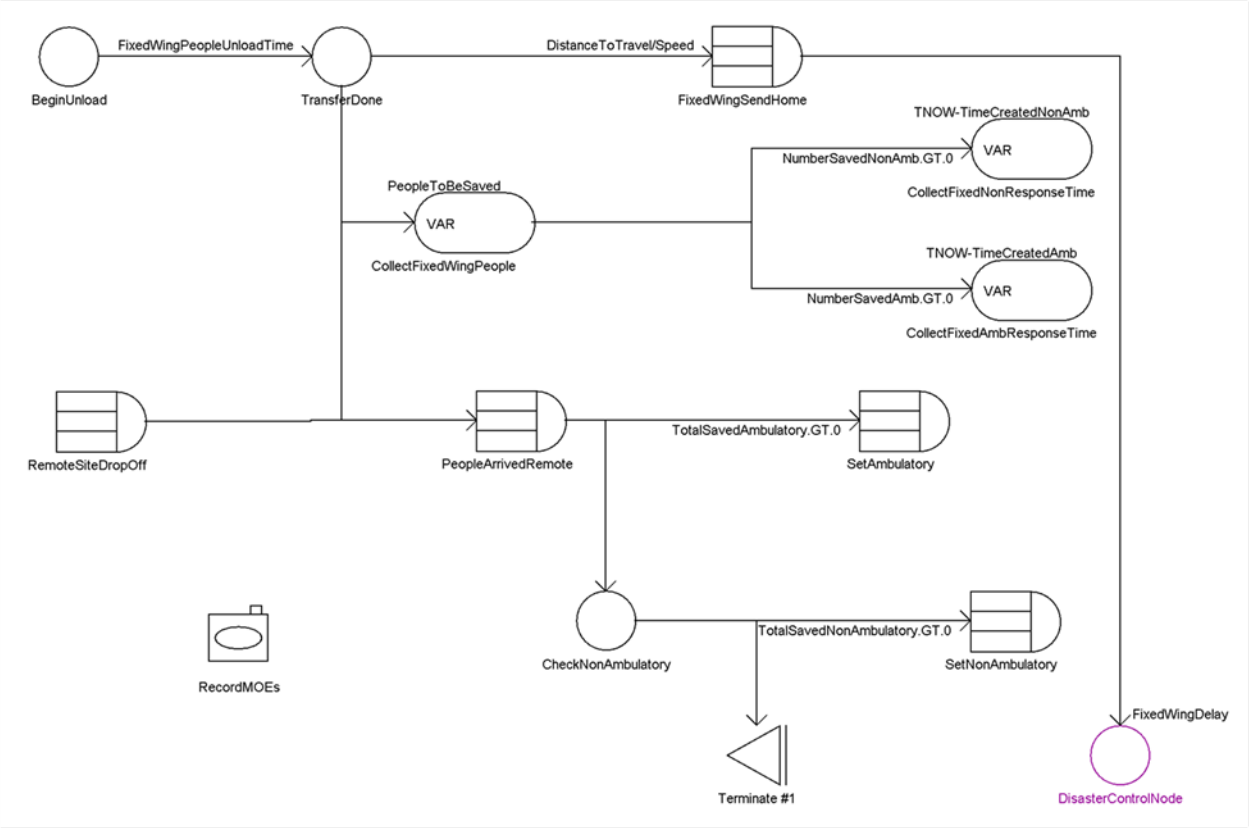


Figure 18. Transfer People Event

Appendix B - Global Attributes

Attributes

The simulation network created within the M^{3}S tool was designed with great flexibility for performing various sensitivity studies. To achieve this flexibility many attributes can be defined by the analyst and numerous measures of effectiveness attributes can be examined. This appendix contains a complete list of the input and output global attributes used in modeling post-disaster relief operations.

* Indicates Output Attribute

- **AveResponseTimeAmb***: Average response time for all ambulatory people. This is defined as the average time between event generation and the time people arrived at a remote site (hr)
- **AveResponseTimeNonAmb***: Average response time for all non ambulatory people. This is defined as the average time between event generation and the time people arrived at a remote site (hr)
- **AveResponseTimeTransport***: Average response time for the transport of supplies, etc. This is defined as the average time between event generation and the time the supplies arrived at a local site (hr)
- **CRCHeavyLitterCapacity**: The maximum litter capacity for heavy conventional helicopters
- **CRCHeavyMissionRange**: The mission range for heavy conventional helicopters. This defines the maximum endurance of the aircraft (nmi)
- **CRCHeavyPaxCapacity**: The maximum passenger capacity for heavy conventional helicopters
- **CRCHeavyPayload**: The maximum payload for heavy conventional helicopters (lb)
- **CRCHeavySpeed**: The cruise speed for heavy conventional helicopters (knots)
- **CRCLightLitterCapacity**: The maximum litter capacity for light conventional helicopters
- **CRCLightMissionRange**: The mission range for light conventional helicopters. This defines the maximum endurance of the aircraft (nmi)
- **CRCLightPaxCapacity**: The maximum passenger capacity for light conventional helicopters
- **CRCLightPayload**: The maximum payload for light conventional helicopters (lb)
- **CRCLightSpeed**: The cruise speed for light conventional helicopters (knots)
- **CRCMediumLitterCapacity**: The maximum litter capacity for medium conventional helicopters
- **CRCMediumMissionRange**: The mission range for medium conventional helicopters. This defines the maximum endurance of the aircraft (nmi)
- **CRCMediumPaxCapacity**: The maximum passenger capacity for medium conventional helicopters
- **CRCMediumPayload**: The maximum payload for medium conventional helicopters (lb)
- **CRCMediumSpeed**: The cruise speed for medium conventional helicopters (knots)
- **CTR10LitterCapacity**: The maximum litter capacity for 10 passenger CTRs

- **CTR10MissionRange**: The mission range for 10 passenger CTRs. This defines the maximum endurance of the aircraft (nmi)
- **CTR10PaxCapacity**: The maximum passenger capacity for 10 passenger CTRs
- **CTR10Payload**: The maximum payload for 10 passenger CTRs (lb)
- **CTR10Speed**: The cruise speed for 10 passenger CTRs (knots)
- **CTR30LitterCapacity**: The maximum litter capacity for 30 passenger CTRs
- **CTR30MissionRange**: The mission range for 30 passenger CTRs. This defines the maximum endurance of the aircraft (nmi)
- **CTR30PaxCapacity**: The maximum passenger capacity for 30 passenger CTRs
- **CTR30Payload**: The maximum payload for 30 passenger CTRs (lb)
- **CTR30Speed**: The cruise speed for 30 passenger CTRs (knots)
- **CTR120LitterCapacity**: The maximum litter capacity for 120 passenger CTRs
- **CTR120MissionRange**: The mission range for 120 passenger CTRs. This defines the maximum endurance of the aircraft (nmi)
- **CTR120PaxCapacity**: The maximum passenger capacity for 120 passenger CTRs
- **CTR120Payload**: The maximum payload for 120 passenger CTRs (lb)
- **CTR120Speed**: The cruise speed for 120 passenger CTRs (knots)
- **DistanceBetweenBaseAndSiteMinimum**: Minimum distance between a local base and local site (nmi)
- **DistanceBetweenBaseAndSiteMaximum**: Maximum distance between a local base and local site (nmi)
- **DistanceBetweenLocalAndEventMinimum:** Minimum distance between the local site/base and the event (nmi)
- **DistanceBetweenLocalAndEventMaximum:** Maximum distance between the local site/base and the event (nmi)
- **DistanceBetweenRemoteAndEventMinimum:** Minimum distance between the remote site and the event (nmi)
- **DistanceBetweenRemoteAndEventMaximum:** Maximum distance between the remote site and the event (nmi)
- **DistanceFromRemoteBaseMinimum**: Minimum distance from remote base to local base (nmi)
- **DistanceFromRemoteBaseMaximum**: Maximum distance from remote base to local base (nmi)
- **DistanceTransferToRemote**: The distance that fixed-winged vehicles must travel to transfer the people from the local sites to the remote sites (nmi)
- **EvacuationCreationLength**: Length of time evacuation events will be created (hr)
- **EvacuationPeopleMinimum**: Minimum number of people to be evacuated per event
- **EvacuationPeopleMaximum**: Maximum number of people to be evacuated per event
- **EvacuationTimeBetweenEvents**: Time between evacuation event generation (hr)
- **EvacuationTotalPeople**: Total number of people to be evacuated over the duration of the simulation
- **EventStartTime**: Start creating events at this time. This attribute can be used to delay the creation of events (hr)
- **FixedWingDelay**: Time to refuel and prepare for next flight (hr)

- **FixedWingPaxCapacity**: The maximum passenger capacity for fixed wing aircraft
- **FixedWingPeopleLoadTime**: Time to load people onto a fixed wing aircraft (hr)
- **FixedWingPeopleUnLoadTime**: Time to unload people off a fixed wing aircraft (hr)
- **FixedWingSpeed**: The cruise speed for fixed wing aircraft (knots)
- **FleetSizeCRCHeavy**: Number of *heavy conventional helicopters* to be used in the scenario
- **FleetSizeCRCLight**: Number of *light conventional helicopters* to be used in the scenario
- **FleetSizeCRCMedium**: Number of *medium conventional helicopters* to be used in the scenario
- **FleetSizeCTR10**: Number of *10 passenger CTRs* to be used in the scenario
- **FleetSizeCTR30**: Number of *30 passenger CTRs* to be used in the scenario
- **FleetSizeCTR120**: Number of *120 passenger CTRs* to be used in the scenario
- **FleetSizeFixedWing**: Number of *fixed wing aircraft* to be used in the scenario
- **MedevacCreationLength**: Length of time Medevac events will be created (hr)
- **MedevacPeopleMinimum**: Minimum number of people to be transported per Medevac event
- **MedevacPeopleMaximum**: Maximum number of people to be transported per Medevac event
- **MedevacTimeBetweenEvents**: Time between Medevac event generation (hr)
- **MedevacTotalPeople**: Total number of people to Medevac over the duration of the simulation
- **PeopleThreshold**: The number of people required before a fixed wing moves people from local to remote site.
- **RefuelTime**: Turnaround time to be used to model delay in refueling an air vehicle (hr)
- **RemoteThreshold**: The number of people being rescued must be greater than this threshold value to be moved to a remote base. Otherwise, they are automatically taken to a local site regardless of the vehicle's mission range left.
- **SARCreationLength**: Length of time SAR events will be created (hr)
- **SARPeopleFound***: Total SAR people found
- **SARPeopleMinimum**: Minimum number of people to be recovered per SAR event
- **SARPeopleMaximum**: Maximum number of people to be recovered per SAR event
- **SARPeopleNotFound***: Total SAR people not found
- **SARRangeRequirement**: Minimum range a vehicle must travel be able to remain on station. For a particular SAR event distance, if a vehicle cannot meet this minimum range requirement, the vehicle will not be assigned to this event.
- **SARSearchRangeMinimum**: Minimum range vehicle will travel searching before finding people (nmi)
- **SARSearchRangeMaximum**: Maximum range vehicle will travel searching before finding people (nmi)
- **SARThresholdCapacity**: Only vehicles with a passenger capacity at or below this value will be assigned SAR missions.
- **SARTimeBetweenEvents**: Time between SAR event generation (hr)
- **SARNumberTotalPeople**: Total number of people to search for over the duration of the simulation

- **TotalMissionsCompleted***: Total missions completed by all air vehicles (excluding fixed wing)
- **TotalPayload***: Total weight of supplies to be transported (lb)
- **TotalSavedAmbulatory***: Total number of ambulatory people saved (Evacuation and SAR)
- **TotalSavedFixedWing***: Total number of ambulatory and non-ambulatory people transferred by fixed wing
- **TotalSavedNonAmbulatory***: Total number of non-ambulatory people saved (Medevcac)
- **TransportCreationLength**: Length of time transport events will be created (hr)
- **TransportPayloadMinimum**: Minimum weight of supplies per transport event (lb)
- **TransportPayloadMaximum**: Maximum weight of supplies per transport event (lb)
- **TransportTimeBetweenEvents**: Time between transport event generation (hr)
- **TransportTotalPayload**: Total quantity of payload to be transported over the duration of the simulation (lb)
- **WeightedResponseAmb***: The sum of the number of ambulatory people in a mission multiplied by the total response time (hr)
- **WeightedResponseNonAmb***: The sum of the number of non-ambulatory people in a mission multiplied by the total response time (hr)
- **WeightedResponseTransport***: The sum of the weight of transported load in a mission multiplied by the total response time (lb x hr)
- **Count**: This is a temporary global attribute which is incremented as air vehicle entities are created used to assign unique tail numbers.

Appendix C - Network XML

XML

This appendix contains the input (in XML) for the M^{3}S tool.

```
<?xml version="1.0" ?>
- <CVE Version="1.0">
+ <Windows>
- <Model Name="CTR Disaster" Class="com.bht.engr.mac.m3s.ui.M3SFacade">
- <Configuration Iterations="100" Seed="8175280540" NodeRuns="100" SnapRuns="100">
  <Name>CTR Disaster</Name>
  <Description />
    </Configuration>
- <Network>
- <Nodes>
- <Node Name="CreateFleetCRCLight"
    Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCRCLight" EndNode="InitialFleetCRCLight" />
    </ActivityList>
- <NextTime>
  <MaximumCreations Value="FleetSizeCRCLight" />
  <FirstCreationTime Value="0.01" />
  <TimeBetweenCreations Value="0" />
    </NextTime>
    </Node>
- <Node Name="CreateFleetCRCMedium"
    Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCRCMedium" EndNode="InitialFleetCRCMedium" />
    </ActivityList>
- <NextTime>
  <MaximumCreations Value="FleetSizeCRCMedium" />
  <FirstCreationTime Value="0.02" />
  <TimeBetweenCreations Value="0" />
    </NextTime>
    </Node>
- <Node Name="CreateFleetCRCHeavy"
    Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCRCHeavy" EndNode="InitialFleetCRCHeavy" />
    </ActivityList>
- <NextTime>
  <MaximumCreations Value="FleetSizeCRCHeavy" />
  <FirstCreationTime Value="0.03" />
  <TimeBetweenCreations Value="0" />
    </NextTime>
    </Node>
- <Node Name="CreateFleetCTR10" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCTR10" EndNode="InitialFleetCTR10" />
```

```
    </ActivityList>
- <NextTime>
  <MaximumCreations Value="FleetSizeCTR10" />
  <FirstCreationTime Value="0.04" />
  <TimeBetweenCreations Value="0" />
    </NextTime>
    </Node>
- <Node Name="CreateFleetCTR30" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCTR30" EndNode="InitialFleetCTR30" />
    </ActivityList>
- <NextTime>
  <MaximumCreations Value="FleetSizeCTR30" />
  <FirstCreationTime Value="0.05" />
  <TimeBetweenCreations Value="0" />
    </NextTime>
    </Node>
- <Node Name="CreateFleetCTR120" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCTR120" EndNode="InitialFleetCTR120" />
    </ActivityList>
- <NextTime>
  <MaximumCreations Value="FleetSizeCTR120" />
  <FirstCreationTime Value="0.06" />
  <TimeBetweenCreations Value="0" />
    </NextTime>
    </Node>
- <Node Name="InitialFleetCTR10" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialRotaryCommon4" EndNode="InitialRotaryCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="VehicleType=4" />
  <Assignment Assign="Speed=CTR10Speed" />
  <Assignment Assign="Payload=CTR10Payload" />
  <Assignment Assign="MissionRange=CTR10MissionRange" />
  <Assignment Assign="PaxCapacity=CTR10PaxCapacity" />
  <Assignment Assign="LitterCapacity=CTR10LitterCapacity" />
    </AssignmentList>
    </Node>
- <Node Name="InitialFleetCTR30" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialRotaryCommon5" EndNode="InitialRotaryCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="VehicleType=5" />
  <Assignment Assign="Speed=CTR30Speed" />
  <Assignment Assign="Payload=CTR30Payload" />
  <Assignment Assign="MissionRange=CTR30MissionRange" />
  <Assignment Assign="PaxCapacity=CTR30PaxCapacity" />
  <Assignment Assign="LitterCapacity=CTR30LitterCapacity" />
```

```
    </AssignmentList>
    </Node>
- <Node Name="InitialFleetCTR120" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialRotaryCommon6" EndNode="InitialRotaryCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="VehicleType=6" />
  <Assignment Assign="Speed=CTR120Speed" />
  <Assignment Assign="Payload=CTR120Payload" />
  <Assignment Assign="MissionRange=CTR120MissionRange" />
  <Assignment Assign="PaxCapacity=CTR120PaxCapacity" />
  <Assignment Assign="LitterCapacity=CTR120LitterCapacity" />
    </AssignmentList>
    </Node>
- <Node Name="InitialFleetCRCHeavy"
     Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialRotaryCommon3" EndNode="InitialRotaryCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="VehicleType=3" />
  <Assignment Assign="Speed=CRCHeavySpeed" />
  <Assignment Assign="Payload=CRCHeavyPayload" />
  <Assignment Assign="MissionRange=CRCHeavyMissionRange" />
  <Assignment Assign="PaxCapacity=CRCHeavyPaxCapacity" />
  <Assignment Assign="LitterCapacity=CRCHeavyLitterCapacity" />
    </AssignmentList>
    </Node>
- <Node Name="InitialFleetCRCMedium"
     Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialRotaryCommon2" EndNode="InitialRotaryCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="VehicleType=2" />
  <Assignment Assign="Speed=CRCMediumSpeed" />
  <Assignment Assign="Payload=CRCMediumPayload" />
  <Assignment Assign="MissionRange=CRCMediumMissionRange" />
  <Assignment Assign="PaxCapacity=CRCMediumPaxCapacity" />
  <Assignment Assign="LitterCapacity=CRCMediumLitterCapacity" />
    </AssignmentList>
    </Node>
- <Node Name="InitialFleetCRCLight" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialRotaryCommon1" EndNode="InitialRotaryCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="VehicleType=1" />
  <Assignment Assign="Speed=CRCLightSpeed" />
  <Assignment Assign="Payload=CRCLightPayload" />
```

```
  <Assignment Assign="MissionRange=CRCLightMissionRange" />
  <Assignment Assign="PaxCapacity=CRCLightPaxCapacity" />
  <Assignment Assign="LitterCapacity=CRCLightLitterCapacity" />
    </AssignmentList>
    </Node>
- <Node Name="InitialFleetCommon" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList MaxBranches="1">
  <Activity Name="GoControlNodeFromIntialFleet" EndNode="DisasterControlNode"
     Condition="VehicleType.EQ.7" />
  <Activity Name="GoRemoteBase" EndNode="RemoteBase" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="TotalDistanceFlown=0" />
  <Assignment Assign="MissionsCompleted=0" />
  <Assignment Assign="MissionRangeLeft=MissionRange" />
  <Assignment Assign="CurrentLocation=0" />
  <Assignment Assign="Count=Count+1" />
  <Assignment Assign="TailNumber=Count" />
    </AssignmentList>
    </Node>
- <Node Name="CreateEvacuation" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialEventEvacuation" EndNode="InitialEventEvacuation" />
    </ActivityList>
- <NextTime Class="EventNextTime">
  <MaximumCreations Value="EvacuationTotalPeople" />
  <FirstCreationTime Value="EventStartTime" />
  <TimeBetweenCreations Value="EvacuationTimeBetweenEvents" />
  <MinimumQuantity Value="EvacuationPeopleMinimum" />
  <MaximumQuantity Value="EvacuationPeopleMaximum" />
  <CreationLength Value="EvacuationCreationLength" />
    </NextTime>
  <NextEntity UseNextTime="true" />
    </Node>
- <Node Name="CreateMedevac" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialEventMedevac" EndNode="InitialEventMedevac" />
    </ActivityList>
- <NextTime Class="EventNextTime">
  <MaximumCreations Value="MedevacTotalPeople" />
  <FirstCreationTime Value="EventStartTime" />
  <TimeBetweenCreations Value="MedevacTimeBetweenEvents" />
  <MinimumQuantity Value="MedevacPeopleMinimum" />
  <MaximumQuantity Value="MedevacPeopleMaximum" />
  <CreationLength Value="MedevacCreationLength" />
    </NextTime>
  <NextEntity UseNextTime="true" />
    </Node>
- <Node Name="CreateSAR" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialEventSAR" EndNode="InitialEventSAR" />
```

```
    </ActivityList>
- <NextTime Class="EventNextTime">
  <MaximumCreations Value="SARTotalPeople" />
  <FirstCreationTime Value="EventStartTime" />
  <TimeBetweenCreations Value="SARTimeBetweenEvents" />
  <MinimumQuantity Value="SARPeopleMinimum" />
  <MaximumQuantity Value="SARPeopleMaximum" />
  <CreationLength Value="SARCreationLength" />
    </NextTime>
  <NextEntity UseNextTime="true" />
    </Node>
- <Node Name="CreateTransport" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialEventTransport" EndNode="InitialEventTransport" />
    </ActivityList>
- <NextTime Class="EventNextTime">
  <MaximumCreations Value="TransportTotalPayload" />
  <FirstCreationTime Value="EventStartTime" />
  <TimeBetweenCreations Value="TransportTimeBetweenEvents" />
  <MinimumQuantity Value="TransportPayloadMinimum" />
  <MaximumQuantity Value="TransportPayloadMaximum" />
  <CreationLength Value="TransportCreationLength" />
    </NextTime>
  <NextEntity UseNextTime="true" />
    </Node>
- <Node Name="InitialEventEvacuation"
     Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialEventCommon1" EndNode="InitialEventCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="EventType=1" />
  <Assignment Assign="Priority=1" />
  <Assignment Assign="Ambulatory=1" />
  <Assignment Assign="PeopleToBeSaved=EventRequirement" />
    </AssignmentList>
    </Node>
- <Node Name="InitialEventMedevac"
     Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialEventCommon2" EndNode="InitialEventCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="EventType=2" />
  <Assignment Assign="Priority=1" />
  <Assignment Assign="Ambulatory=0" />
  <Assignment Assign="PeopleToBeSaved=EventRequirement" />
    </AssignmentList>
    </Node>
- <Node Name="InitialEventSAR" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
```

```
<Activity Name="GoInitialEventCommon3" EndNode="InitialEventCommon" />
  </ActivityList>
- <AssignmentList>
<Assignment Assign="EventType=3" />
<Assignment Assign="Priority=1" />
<Assignment Assign="PeopleToBeSaved=EventRequirement" />
<Assignment
  Assign="SearchRangeToFind=USERF(CTRRand(SARSearchRangeMinimum,SARSear
  chRangeMaximum))" />
<Assignment Assign="Ambulatory=1" />
  </AssignmentList>
  </Node>
- <Node Name="InitialEventTransport"
  Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
<Activity Name="GoInitialEventCommon4" EndNode="InitialEventCommon" />
  </ActivityList>
- <AssignmentList>
<Assignment Assign="EventType=4" />
<Assignment Assign="Priority=1" />
<Assignment Assign="PayloadToTransport=EventRequirement" />
  </AssignmentList>
  </Node>
- <Node Name="InitialEventCommon"
  Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
<Activity Name="GoMasterController2" EndNode="DisasterControlNode" />
  </ActivityList>
- <AssignmentList>
<Assignment
  Assign="DistanceLocalToEvent=USERF(CTRRand(DistanceBetweenLocalAndEventM
  inimum,DistanceBetweenLocalAndEventMaximum))" />
<Assignment
  Assign="DistanceRemoteToEvent=USERF(CTRRand(DistanceBetweenRemoteAndEv
  entMinimum,DistanceBetweenRemoteAndEventMaximum))" />
<Assignment
  Assign="DistanceEventToLocal=USERF(CTRRand(DistanceBetweenLocalAndEventM
  inimum,DistanceBetweenLocalAndEventMaximum))" />
<Assignment
  Assign="DistanceEventToRemote=USERF(CTRRand(DistanceBetweenRemoteAndEv
  entMinimum,DistanceBetweenRemoteAndEventMaximum))" />
<Assignment Assign="TimeCreated=TNOW" />
  </AssignmentList>
  </Node>
- <Node Name="RequestSiteMedevac"
  Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
<Activity Name="GoDropOffFromMedevac" EndNode="GoOnHome" />
  </ActivityList>
- <AssignmentList>
<Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceToTravel" />
```

```
    </AssignmentList>
    </Node>
- <Node Name="RequestSiteEvacuation"
    Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoDropOffFromEvac" EndNode="GoOnHome" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceToTravel" />
    </AssignmentList>
    </Node>
- <Node Name="RequestSiteSAR" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList MaxBranches="1">
  <Activity Name="GoPeopleFound" EndNode="PeopleFound"
    Duration="SearchRangeToFind/Speed"
    Condition="SearchRangeToFind.LT.MaximumSearchRange" />
  <Activity Name="GoPeopleNotFound" EndNode="PeopleNotFound"
    Duration="MaximumSearchRange/Speed" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceToTravel" />
    </AssignmentList>
    </Node>
- <Node Name="RequestSitePickUp" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoLocalDropOffTransport" EndNode="LocalSiteDropOffTransport"
    Duration="DistanceEventToLocal/Speed" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft= MissionRangeLeft-DistanceToTravel" />
    </AssignmentList>
    </Node>
- <Node Name="LocalSiteDropOffTransport"
    Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoProceedLocal2" EndNode="LocalSite" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceEventToLocal" />
  <Assignment Assign="CurrentLocation=1" />
  <Assignment Assign="TotalPayload=TotalPayload+PayloadToTransport" />
  <Assignment Assign="ResponseTime=TNOW-TimeCreated" />
  <Assignment Assign="Temp=ResponseTime*PayloadToTransport" />
  <Assignment
    Assign="WeightedResponseTransport=WeightedResponseTransport+Temp" />
  <Assignment
    Assign="AveResponseTimeTransport=WeightedResponseTransport/TotalPayload"
    />
    </AssignmentList>
    </Node>
- <Node Name="LocalSiteDropOff" Type="com.bht.engr.mac.m3s.input.AssignNode">
```

```
- <ActivityList>
  <Activity Name="GoProceedLocal1" EndNode="LocalSite" />
  <Activity Name="AddPeopleToQueue" EndNode="PeopleQueue" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceEventToLocal" />
  <Assignment Assign="CurrentLocation=1" />
    </AssignmentList>
    </Node>
- <Node Name="RemoteSiteDropOff" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoRefuelFromRemoteSite" EndNode="Refuel" />
  <Activity Name="GoPeopleArrivedRemote2" EndNode="PeopleArrivedRemote" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceEventToRemote"
    />
  <Assignment Assign="CurrentLocation=2" />
    </AssignmentList>
    </Node>
- <Node Name="Refuel" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoMCFromRefuel" EndNode="MissionComplete" Duration="RefuelTime"
    />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="Temp=MissionRange-MissionRangeLeft" />
  <Assignment Assign="TotalDistanceFlown=TotalDistanceFlown+Temp" />
  <Assignment Assign="MissionRangeLeft=MissionRange" />
    </AssignmentList>
    </Node>
- <Node Name="RemoteBase">
- <ActivityList MaxBranches="1">
  <Activity Name="GoRemoteRefuel" EndNode="RemoteRefuel" Duration="MissionRange /
    Speed" Condition="DistanceToTravel.GT.MissionRange" />
  <Activity Name="GoLocalBaseInitial" EndNode="LocalBaseInitial"
    Duration="DistanceToTravel / Speed" />
    </ActivityList>
    </Node>
- <Node Name="RemoteRefuel" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoAgainRemoteBase" EndNode="RemoteBase" Duration="RefuelTime"
    />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="TotalDistanceFlown=TotalDistanceFlown+MissionRange" />
  <Assignment Assign="DistanceToTravel=DistanceToTravel-MissionRange" />
    </AssignmentList>
    </Node>
- <Node Name="LocalBaseInitial" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
```

```
<Activity Name="GoMasterController1" EndNode="DisasterControlNode"
   Duration="RefuelTime" />
<Activity Name="GoCollectRemoteTravelTime" EndNode="CollectRemoteTravelTime" />
   </ActivityList>
- <AssignmentList>
<Assignment Assign="TotalDistanceFlown=TotalDistanceFlown+DistanceToTravel" />
   </AssignmentList>
   </Node>
- <Node Name="CollectRemoteTravelTime"
   Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
- <ActivityList>
<Activity Name="GoCollectRemoteTravelDistance"
   EndNode="CollectRemoteTravelDistance" />
   </ActivityList>
<Variable Expression="TNOW" />
   </Node>
- <Node Name="CollectRemoteTravelDistance"
   Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
<Variable Expression="TotalDistanceFlown" />
   </Node>
- <Node Name="DisasterControlNode" Type="DisasterControlNode">
- <ActivityList MaxBranches="1">
<Activity Name="GoProcessEvent1" EndNode="ProcessEvent"
   Condition="EventType.EQ.5" />
<Activity Name="GoComputeInitialResponse" EndNode="InitialResponse" />
   </ActivityList>
   </Node>
- <Node Name="LocalSite" Type="CheckFuelNode">
- <ActivityList MaxBranches="1">
<Activity Name="GoLocalBase" EndNode="LocalBase"
   Duration="DistanceToTravel/Speed" Condition="RefuelFlag.EQ.1" />
<Activity Name="GoMCFromLocalSite" EndNode="MissionComplete" />
   </ActivityList>
   </Node>
- <Node Name="GoOnHome">
- <ActivityList MaxBranches="1">
<Activity Name="GoRemoteSite" EndNode="RemoteSiteDropOff"
   Duration="DistanceEventToRemote / Speed"
   Condition="Ambulatory.EQ.1.AND.PercentCapacity.GT.RemoteThreshold.AND.Missio
   nRangeLeft.GT.DistanceEventToRemote" />
<Activity Name="GoLocalSite" EndNode="LocalSiteDropOff"
   Duration="DistanceEventToLocal / Speed" />
   </ActivityList>
   </Node>
- <Node Name="PeopleQueue" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
<Activity Name="GoMCFromPeopleQueue" EndNode="DisasterControlNode" />
   </ActivityList>
- <AssignmentList>
<Assignment Assign="PeopleToQueue=PeopleToBeSaved" />
   </AssignmentList>
```

```
    </Node>
- <Node Name="CollectTransportResponseTime"
    Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
  <Variable Expression="ResponseTime" />
    </Node>
- <Node Name="PeopleFound" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoDropOffFromPeopleFound" EndNode="GoOnHome" />
  <Activity Name="GoCollectPeopleFound" EndNode="CollectPeopleFound" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="SARPeopleFound=SARPeopleFound+PeopleToBeSaved" />
  <Assignment Assign="NumberSavedAmb=PeopleToBeSaved" />
  <Assignment Assign="TimeCreatedAmb=TimeCreated" />
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-SearchRangeToFind" />
    </AssignmentList>
    </Node>
- <Node Name="PeopleNotFound" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList MaxBranches="2">
  <Activity Name="GoCollectPeopleNotFound" EndNode="CollectPeopleNotFound" />
  <Activity Name="GoSameLocalBase" Show="true" EndNode="SameLocalBase"
    Duration="DistanceLocalToEvent/Speed"
    Condition="CurrentLocation.EQ.0.OR.DistanceEventToLocal.GT.DistanceLocalToEven
    t" />
  <Activity Name="GoDifferentLocalBase" EndNode="DifferentLocalBase"
    Duration="DistanceEventToLocal/Speed" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="SARPeopleNotFound=SARPeopleNotFound+PeopleToBeSaved"
    />
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-MaximumSearchRange" />
    </AssignmentList>
    </Node>
- <Node Name="CollectPeopleNotFound"
    Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
  <Variable Expression="PeopleToBeSaved" />
    </Node>
- <Node Name="CollectPeopleFound"
    Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
  <Variable Expression="PeopleToBeSaved" />
    </Node>
- <Node Name="RequestTransferPeople"
    Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoBeginUnload" EndNode="BeginUnload"
    Duration="DistanceToTravel/Speed" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="TotalSavedFixedWing=TotalSavedFixedWing+PeopleToBeSaved"
    />
    </AssignmentList>
```

```
    </Node>
- <Node Name="CreateFixedWing" Type="com.bht.engr.mac.m3s.input.CreateNode">
- <ActivityList>
  <Activity Name="GoInitialFixedWing" EndNode="InitialFixedWing" />
    </ActivityList>
- <NextTime>
  <MaximumCreations Value="FleetSizeFixedWing" />
  <FirstCreationTime Value="0.07" />
  <TimeBetweenCreations Value="0" />
    </NextTime>
    </Node>
- <Node Name="InitialFixedWing" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCommon2" EndNode="InitialFleetCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="VehicleType=7" />
  <Assignment Assign="Speed=FixedWingSpeed" />
  <Assignment Assign="PaxCapacity=FixedWingPaxCapacity" />
  <Assignment Assign="MissionRange=0" />
    </AssignmentList>
    </Node>
- <Node Name="PeopleArrivedRemote"
    Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoSetAmbulatory" EndNode="SetAmbulatory"
    Condition="TotalSavedAmbulatory.GT.0" />
  <Activity Name="GoCheckNonAmbulatory" EndNode="CheckNonAmbulatory" />
    </ActivityList>
- <AssignmentList>
  <Assignment
    Assign="TotalSavedAmbulatory=TotalSavedAmbulatory+NumberSavedAmb" />
  <Assignment Assign="ResponseTime=TNOW-TimeCreatedAmb" />
  <Assignment Assign="Temp=ResponseTime*NumberSavedAmb" />
  <Assignment Assign="WeightedResponseAmb=WeightedResponseAmb+Temp" />
  <Assignment
    Assign="TotalSavedNonAmbulatory=TotalSavedNonAmbulatory+NumberSavedNon
    Amb" />
  <Assignment Assign="ResponseTime=TNOW-TimeCreatedNonAmb" />
  <Assignment Assign="Temp=ResponseTime*NumberSavedNonAmb" />
  <Assignment Assign="WeightedResponseNonAmb=WeightedResponseNonAmb+Temp"
    />
    </AssignmentList>
    </Node>
- <Node Name="TransferDone">
- <ActivityList>
  <Activity Name="GoPeopleArrivedRemote1" EndNode="PeopleArrivedRemote" />
  <Activity Name="GoFixedWingSendHome" EndNode="FixedWingSendHome"
    Duration="DistanceToTravel/Speed" />
  <Activity Name="GoToCollectFixedWingPeople" EndNode="CollectFixedWingPeople" />
    </ActivityList>
```

```
    </Node>
- <Node Name="FixedWingSendHome"
    Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="ReturnFixedWingToControlNode" EndNode="DisasterControlNode"
    Duration="FixedWingDelay" />
    </ActivityList>
    </Node>
- <Node Name="MissionComplete" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList MaxBranches="2">
  <Activity Name="GoControllerFromMissionCompleted" EndNode="DisasterControlNode"
    />
  <Activity Name="GoCountEvac" Show="true" Accumulate="true"
    EndNode="TerminateEvac" Condition="EventType.EQ.1" />
  <Activity Name="GoCountMedEvac" Show="true" Accumulate="true"
    EndNode="TerminateMedEvac" Condition="EventType.EQ.2" />
  <Activity Name="GoCountSAR" Show="true" Accumulate="true"
    EndNode="TerminateSAR" Condition="EventType.EQ.3" />
  <Activity Name="GoCountTransport" Show="true" Accumulate="true"
    EndNode="TerminateTransport" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionsCompleted=MissionsCompleted+1" />
  <Assignment Assign="TotalMissionsCompleted=TotalMissionsCompleted+1" />
    </AssignmentList>
    </Node>
- <Node Name="InitialResponse" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoProcessEvent2" EndNode="ProcessEvent" />
  <Activity Name="GoCollectInitialResponse" EndNode="CollectInitialReponse" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="InitialResponse=TNOW-TimeCreated" />
    </AssignmentList>
    </Node>
- <Node Name="ProcessEvent">
- <ActivityList MaxBranches="1">
  <Activity Name="GoRequestEvacuation" EndNode="RequestSiteEvacuation"
    Duration="DistanceToTravel/Speed" Condition="EventType.EQ.1" />
  <Activity Name="GoRequestMedevac" EndNode="RequestSiteMedevac"
    Duration="DistanceToTravel/Speed" Condition="EventType.EQ.2" />
  <Activity Name="GoRequestSAR" EndNode="RequestSiteSAR"
    Duration="DistanceToTravel/Speed" Condition="EventType.EQ.3" />
  <Activity Name="GoRequestPickUp" EndNode="RequestSitePickUp"
    Duration="DistanceToTravel/Speed" Condition="EventType.EQ.4" />
  <Activity Name="GoRequestTransferPeople" EndNode="RequestTransferPeople"
    Duration="FixedWingPeopleLoadTime" />
    </ActivityList>
    </Node>
- <Node Name="CollectInitialReponse"
    Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
```

```
  <Variable Expression="InitialResponse" />
    </Node>
- <Node Name="RecordMOEs" Type="com.bht.engr.mac.m3s.input.SnapshotNode">
- <NextTime>
  <MaximumCreations Value="30" />
  <FirstCreationTime Value="0" />
  <TimeBetweenCreations Value="10" />
    </NextTime>
- <SnapList>
  <Snap Attribute="TotalMissionsCompleted" />
  <Snap Attribute="TotalPayload" />
  <Snap Attribute="AveResponseTimeTransport" />
  <Snap Attribute="TotalSavedNonAmbulatory" />
  <Snap Attribute="AveResponseTimeNonAmb" />
  <Snap Attribute="TotalSavedAmbulatory" />
  <Snap Attribute="AveResponseTimeAmb" />
  <Snap Attribute="SARPeopleFound" />
  <Snap Attribute="SARPeopleNotFound" />
    </SnapList>
    </Node>
- <Node Name="InitialRotaryCommon"
    Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoInitialFleetCommon1" EndNode="InitialFleetCommon" />
    </ActivityList>
- <AssignmentList>
  <Assignment
    Assign="DistanceFromRemoteBase=RAND(DistanceFromRemoteBaseMinimum,Dist
    anceFromRemoteBaseMaximum)" />
  <Assignment Assign="DistanceToTravel=DistanceFromRemoteBase" />
    </AssignmentList>
    </Node>
- <Node Name="SetAmbulatory" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <AssignmentList>
  <Assignment
    Assign="AveResponseTimeAmb=WeightedResponseAmb/TotalSavedAmbulatory"
    />
    </AssignmentList>
    </Node>
- <Node Name="SetNonAmbulatory" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <AssignmentList>
  <Assignment
    Assign="AveResponseTimeNonAmb=WeightedResponseNonAmb/TotalSavedNonA
    mbulatory" />
    </AssignmentList>
    </Node>
- <Node Name="CheckNonAmbulatory">
- <ActivityList>
  <Activity Name="GoSetNonAmbulatory" EndNode="SetNonAmbulatory"
    Condition="TotalSavedNonAmbulatory.GT.0" />
  <Activity Name="GoTerminate1" EndNode="Terminate #1" />
```

```
    </ActivityList>
    </Node>
  <Node Name="Terminate #1" Type="com.bht.engr.mac.m3s.input.TerminateNode" />
- <Node Name="BeginUnload">
- <ActivityList>
  <Activity Name="GoTransferDone" EndNode="TransferDone"
     Duration="FixedWingPeopleUnloadTime" />
    </ActivityList>
    </Node>
- <Node Name="LocalBase" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoRefuelFromLocalBase" EndNode="Refuel" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceToTravel" />
  <Assignment Assign="CurrentLocation=0" />
    </AssignmentList>
    </Node>
- <Node Name="SameLocalBase" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoLocalBaseFromSame" EndNode="LocalBase" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceLocalToEvent" />
  <Assignment Assign="DistanceToTravel=0" />
    </AssignmentList>
    </Node>
- <Node Name="DifferentLocalBase" Type="com.bht.engr.mac.m3s.input.AssignNode">
- <ActivityList>
  <Activity Name="GoFromBaseDifferentLocalBase" EndNode="LocalBase" />
    </ActivityList>
- <AssignmentList>
  <Assignment Assign="MissionRangeLeft=MissionRangeLeft-DistanceEventToLocal" />
  <Assignment Assign="DistanceToTravel=0" />
    </AssignmentList>
    </Node>
- <Node Name="CollectFixedWingPeople"
     Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
- <ActivityList>
  <Activity Name="GoCollectFixedNonResponseTime"
     EndNode="CollectFixedNonResponseTime" Condition="NumberSavedNonAmb.GT.0"
     />
  <Activity Name="GoCollectFixedAmbResponseTime"
     EndNode="CollectFixedAmbResponseTime" Condition="NumberSavedAmb.GT.0" />
    </ActivityList>
  <Variable Expression="PeopleToBeSaved" />
    </Node>
- <Node Name="CollectFixedNonResponseTime"
     Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
  <Variable Expression="TNOW-TimeCreatedNonAmb" />
    </Node>
```

```
- <Node Name="CollectFixedAmbResponseTime"
    Type="com.bht.engr.mac.m3s.input.VariableCollectNode">
  <Variable Expression="TNOW-TimeCreatedAmb" />
    </Node>
  <Node Name="TerminateEvac" Type="com.bht.engr.mac.m3s.input.TerminateNode" />
  <Node Name="TerminateMedEvac"
    Type="com.bht.engr.mac.m3s.input.TerminateNode" />
  <Node Name="TerminateSAR" Type="com.bht.engr.mac.m3s.input.TerminateNode" />
  <Node Name="TerminateTransport"
    Type="com.bht.engr.mac.m3s.input.TerminateNode" />
    </Nodes>
  <ResourceProviders />
  <Gates />
  <ServiceProviders />
  <Unuseds />
    </Network>
- <GlobalAttributeList>
- <GlobalAttribute Name="AveResponseTimeAmb" Value="0">
  <Description>Average response time for all ambulatory people. This is defined as the
    average time between event generation and the time people arrived at a remote
    site (hrs)</Description>
    </GlobalAttribute>
- <GlobalAttribute Name="AveResponseTimeNonAmb" Value="0">
  <Description>Average response time for all non ambulatory people. This is defined as
    the average time between event generation and the time people arrived at a
    remote site (hrs)</Description>
    </GlobalAttribute>
- <GlobalAttribute Name="AveResponseTimeTransport" Value="0">
  <Description>Average response time for the transport of supplies, etc. This is defined
    as the average time between event generation and the time the supplies arrived at
    a local site (hrs)</Description>
    </GlobalAttribute>
- <GlobalAttribute Name="CRCHeavyLitterCapacity" Value="21">
  <Description>The maximum litter capacity for heavy conventional
    helicopters</Description>
    </GlobalAttribute>
- <GlobalAttribute Name="CRCHeavyMissionRange" Value="513">
  <Description>The mission range for heavy conventional helicopters. This defines the
    maximum endurance of the aircraft (nmi)</Description>
    </GlobalAttribute>
- <GlobalAttribute Name="CRCHeavyPaxCapacity" Value="43">
  <Description>The maximum passenger capacity for heavy conventional
    helicopters</Description>
    </GlobalAttribute>
- <GlobalAttribute Name="CRCHeavyPayload" Value="19462">
  <Description>The maximum payload for heavy conventional helicopters
    (lbs)</Description>
    </GlobalAttribute>
- <GlobalAttribute Name="CRCHeavySpeed" Value="143">
  <Description>The cruise speed for heavy conventional helicopters
    (knots)</Description>
```

```
</GlobalAttribute>
- <GlobalAttribute Name="CRCLightLitterCapacity" Value="3">
<Description>The maximum litter capacity for light conventional
helicopters</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCLightMissionRange" Value="330">
<Description>The mission range for light conventional helicopters. This defines the
maximum endurance of the aircraft (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCLightPaxCapacity" Value="8">
<Description>The maximum passenger capacity for light conventional
helicopters</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCLightPayload" Value="1669">
<Description>The maximum payload for light conventional helicopters
(lbs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCLightSpeed" Value="129">
<Description>The cruise speed for light conventional helicopters (knots)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCMediumLitterCapacity" Value="10">
<Description>The maximum litter capacity for medium conventional
helicopters</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCMediumMissionRange" Value="388">
<Description>The mission range for medium conventional helicopters. This defines the
maximum endurance of the aircraft (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCMediumPaxCapacity" Value="18">
<Description>The maximum passenger capacity for medium conventional
helicopters</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCMediumPayload" Value="6254">
<Description>The maximum payload for medium conventional helicopters
(lbs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CRCMediumSpeed" Value="138">
<Description>The cruise speed for medium conventional helicopters
(knots)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CTR10LitterCapacity" Value="5">
<Description>The maximum litter capacity for 10 passenger CTRs</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CTR10MissionRange" Value="821">
<Description>The mission range for 10 passenger CTRs. This defines the maximum
endurance of the aircraft (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CTR10PaxCapacity" Value="10">
<Description>The maximum passenger capacity for 10 passenger CTRs</Description>
</GlobalAttribute>
- <GlobalAttribute Name="CTR10Payload" Value="2200">
```

<Description>**The maximum payload for 10 passenger CTRs (lbs)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR10Speed**" Value="**280**">
<Description>**The cruise speed for 10 passenger CTRs (knots)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR30LitterCapacity**" Value="**15**">
<Description>**The maximum litter capacity for 30 passenger CTRs**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR30MissionRange**" Value="**804**">
<Description>**The mission range for 30 passenger CTRs. This defines the maximum endurance of the aircraft (nmi)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR30PaxCapacity**" Value="**30**">
<Description>**The maximum passenger capacity for 30 passenger CTRs**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR30Payload**" Value="**6600**">
<Description>**The maximum payload for 30 passenger CTRs (lbs)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR30Speed**" Value="**340**">
<Description>**The cruise speed for 30 passenger CTRs (knots)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR120LitterCapacity**" Value="**60**">
<Description>**The maximum litter capacity for 120 passenger CTRs**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR120MissionRange**" Value="**1331**">
<Description>**The mission range for 120 passenger CTRs. This defines the maximum endurance of the aircraft (nmi)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR120PaxCapacity**" Value="**120**">
<Description>**The maximum passenger capacity for 120 passenger CTRs**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR120Payload**" Value="**26399**">
<Description>**The maximum payload for 120 passenger CTRs (lbs)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**CTR120Speed**" Value="**345**">
<Description>**The cruise speed for 120 passenger CTRs (knots)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**DistanceBetweenBaseAndSiteMinimum**" Value="**1**">
<Description>**Minimum distance between a local base and local site (nmi)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**DistanceBetweenBaseAndSiteMaximum**" Value="**50**">
<Description>**Maximum distance between a local base and local site (nmi)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**DistanceBetweenLocalAndEventMinimum**" Value="**0**">
<Description>**Minimum distance between the local site/base and the event (nmi)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**DistanceBetweenLocalAndEventMaximum**" Value="**150**">

<Description>Maximum distance between the local site/base and the event (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="DistanceBetweenRemoteAndEventMinimum" Value="300">
<Description>Minimum distance between the remote site and the event (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="DistanceBetweenRemoteAndEventMaximum" Value="500">
<Description>Maximum distance between the remote site and the event (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="DistanceFromRemoteBaseMinimum" Value="300">
<Description>Minimum distance from remote base to local base (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="DistanceFromRemoteBaseMaximum" Value="1000">
<Description>Maximum distance from remote base to local base (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="DistanceTransferToRemote" Value="400">
<Description>The distance that fixed-winged vehicles must travel to transfer the people from the local sites to the remote sites (nmi)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="EvacuationCreationLength" Value="264">
<Description>Length of time evacuation events will be created (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="EvacuationPeopleMinimum" Value="1">
<Description>Minimum number of people to be evacuated per event</Description>
</GlobalAttribute>
- <GlobalAttribute Name="EvacuationPeopleMaximum" Value="500">
<Description>Maximum number of people to be evacuated per event</Description>
</GlobalAttribute>
- <GlobalAttribute Name="EvacuationTimeBetweenEvents" Value="6">
<Description>Time between evacuation event generation (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="EvacuationTotalPeople" Value="18078">
<Description>Total number of people to be evacuated over the duration of the simulation</Description>
</GlobalAttribute>
- <GlobalAttribute Name="EventStartTime" Value="0">
<Description>Start creating events at this time. This attribute can be used to delay the creation of events (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FixedWingDelay" Value="0">
<Description>Time to refuel and prepare for next flight (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FixedWingPaxCapacity" Value="175">
<Description>The maximum passenger capacity for fixed wing aircraft</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FixedWingPeopleLoadTime" Value="0.75">
<Description>Time to load people onto a fixed wing aircraft (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FixedWingPeopleUnloadTime" Value="0.75">

```
<Description>Time to unload people off a fixed wing aircraft (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FixedWingSpeed" Value="400">
<Description>The cruise speed for fixed wing aircraft (knots)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FleetSizeCRCLight" Value="95">
<Description>Number of light conventional helicopters to be used in the
scenario</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FleetSizeCRCMedium" Value="233">
<Description>Number of medium conventional helicopters to be used in the
scenario</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FleetSizeCRCHeavy" Value="68">
<Description>Number of heavy conventional helicopters to be used in the
scenario</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FleetSizeCTR10" Value="0">
<Description>Number of 10 passenger CTRs to be used in the scenario</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FleetSizeCTR30" Value="0">
<Description>Number of 30 passenger CTRs to be used in the scenario</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FleetSizeCTR120" Value="0">
<Description>Number of 120 passenger CTRs to be used in the scenario</Description>
</GlobalAttribute>
- <GlobalAttribute Name="FleetSizeFixedWing" Value="50">
<Description>Number of fixed wing aircraft to be used in the scenario</Description>
</GlobalAttribute>
- <GlobalAttribute Name="MedevacCreationLength" Value="132">
<Description>Length of time MedEvac events will be created (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="MedevacPeopleMinimum" Value="1">
<Description>Minimum number of people to be transported per Medevac
event</Description>
</GlobalAttribute>
- <GlobalAttribute Name="MedevacPeopleMaximum" Value="100">
<Description>Maximum number of people to be transported per Medevac
event</Description>
</GlobalAttribute>
- <GlobalAttribute Name="MedevacTimeBetweenEvents" Value="6">
<Description>Time between Medevac event generation (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="MedevacTotalPeople" Value="10626">
<Description>Total number of people to Medevac over the duration of the
simulation</Description>
</GlobalAttribute>
- <GlobalAttribute Name="PeopleThreshold" Value="150">
<Description>The number of people required before a fixed wing moves people from
local to remote site.</Description>
</GlobalAttribute>
```

- <GlobalAttribute Name="**RefuelTime**" Value="**0.5**">
<Description>**Turnaround time to be used to model delay in refueling an air vehicle (hrs)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**RemoteThreshold**" Value="**0.75**">
<Description>**The number of people being rescued must be greater than this threshold value to be moved to a remote base. Otherwise, they are automatically taken to a local site regardless of the vehicle's mission range left.**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARCreationLength**" Value="**132**">
<Description>**Length of time SAR events will be created (hrs)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARPeopleFound**" Value="**0**">
<Description>**Total SAR people found**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARPeopleMinimum**" Value="**1**">
<Description>**Minimum number of people to be recovered per SAR event**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARPeopleMaximum**" Value="**10**">
<Description>**Maximum number of people to be recovered per SAR event**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARPeopleNotFound**" Value="**0**">
<Description>**Total SAR people not found**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARRangeRequirement**" Value="**25**">
<Description>**Minimum range a vehicle must travel be able to remain on station. For a particular SAR event distance, if a vehicle cannot meet this minimum range requirement, the vehicle will not be assigned to this event.**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARSearchRangeMinimum**" Value="**0**">
<Description>**Minimum range vehicle will travel searching before finding people (nmi)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARSearchRangeMaximum**" Value="**50**">
<Description>**Maximum range vehicle will travel searching before finding people (nmi)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARThresholdCapacity**" Value="**20**">
<Description>**Only vehicles with a passenger capacity at or below this value will be assigned SAR missions.**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARTimeBetweenEvents**" Value="**6**">
<Description>**Time between SAR event generation (hrs)**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**SARTotalPeople**" Value="**24367**">
<Description>**Total number of people to search for over the duration of the simulation**</Description>
</GlobalAttribute>
- <GlobalAttribute Name="**TotalMissionsCompleted**" Value="**0**">
<Description>**Total missions completed by all air vehicles (excluding fixed wing)**</Description>

```
</GlobalAttribute>
- <GlobalAttribute Name="TotalPayload" Value="0">
<Description>Total weight of supplies to be transported (lbs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TotalSavedAmbulatory" Value="0">
<Description>Total number of ambulatory people saved (Evacuation and
SAR)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TotalSavedFixedWing" Value="0">
<Description>Total number of ambulaotory and non-ambulatory people transferred by
fixed wing</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TotalSavedNonAmbulatory" Value="0">
<Description>Total number of non-ambulatory people saved (Medevcac)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TransportCreationLength" Value="198">
<Description>Length of time transport events will be created (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TransportPayloadMinimum" Value="500">
<Description>Minimum weight of supplies per transport event (lbs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TransportPayloadMaximum" Value="20000">
<Description>Maximum weight of supplies per transport event (lbs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TransportTimeBetweenEvents" Value="6">
<Description>Time between transport event generation (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="TransportTotalPayload" Value="7580000">
<Description>Total quantity of payload to be transported over the duration of the
simulation (lbs) 3790 tons = 7580000 lbs</Description>
</GlobalAttribute>
- <GlobalAttribute Name="WeightedResponseAmb" Value="0">
<Description>The sum of the number of ambulatory people in a mission multiplied by
the total response time (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="WeightedResponseNonAmb" Value="0">
<Description>The sum of the number of non-ambulatory people in a mission
multiplied by the total response time (hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="WeightedResponseTransport" Value="0">
<Description>The sum of the weight of transported load in a mission multiplied by the
total response time (lbs x hrs)</Description>
</GlobalAttribute>
- <GlobalAttribute Name="Count" Value="0">
<Description>This is a temporary global attribute which is incremented as air vehicle
entities are created used to assign unique tail numbers.</Description>
</GlobalAttribute>
</GlobalAttributeList>
</Model>
</CVE>
```

Appendix D - Java Source Code

Table of Contents

Introduction

The M^{3}S tool provides a capability where user-defined nodes can be plugged into the simulation network. These nodes implement various M^{3}S interfaces, which helps provide specificity in the model. This appendix contains the specialized java source code written to support the CTR post-disaster relief operations simulation. The classes included in this appendix are as follows:

- **CheckFuelNode**: This class is the local site check fuel node. It extends the M^{3}S Node class. When air vehicles arrive at this node the Disaster control node will be invoked to check whether this air vehicle needs to return to a local base or go on another mission without refueling.
- **CheckFuelNodeCanvas**: This class extends the M^{3}S GoOnNodeCanvas class adding no extra functionality while representing the check fuel node on the M^{3}S canvas.
- **CheckFuelNodeEditDbx**: This class extends the M^{3}S GoOnNodeEditDbx dialog adding no extra functionality. This dialog allows the user to change properties of the node.
- **CTRRand**: This class extends the M^{3}S M3SComputeValue class and is used during event creation to produce random numbers ensuring that the event generation will be the same for any particular fleet mix.
- **DisasterControlNode**: This class is the master controller for the simulation. It extends the M^{3}S Node class and maintains two queues (event and local people) and two air vehicle lists (available rotary and available fixed wing).
- **DisasterControlNodeCanvas**: This class extends the M^{3}S GoOnNodeCanvas class adding no extra functionality while representing the disaster control node on the M^{3}S canvas.
- **DisasterControlNodeEditDbx**: This class extends the M^{3}S GoOnNodeEditDbx dialog adding no extra functionality. This dialog allows the user to change properties of the node.
- **DisasterStencil**: This class extends the M^{3}S NetworkStencil class and creates a DisasterControlNode when dragged onto the M^{3}S canvas.
- **EventNextTime**: This class extends the M^{3}S IntervalNextTime class and implements the NextEntity interface. This class creates events using a linear degradation.
- **EventNextTimeEditDbx**: This class extends the M^{3}S IntervalNextTimeEditDbx class adding functionality for the user to edit the parameters needed for the EventNextTime class.
- **PeopleContainer**: This class contains a list of PeopleGroup objects. This is a list of all people rescued and waiting at a local site to be transferred to a remote base.
- **PeopleGroup**: This class contains the following information about a group of people; the number, the time the event was created to rescue these people, and whether they are ambulatory or not.
- **VehicleCapacityCompartor**: The DisasterControlNode uses this class to sort available air vehicles by their capacity (passenger or pay load).
- **VehicleRangeComparator**: The DisasterControlNode uses this class to sort available air vehicles by their current mission range left.
- **VehicleTailComparator**: The DisasterControlNode uses this class to sort all air vehicles by their tail number when generating the final report.

- **VehicleTOSComparator**: The DisasterControlNodes uses this class to sort all air vehicles by TOS which is measured by range left after range to event, range to home, and range to refuel are subtracted out.

CheckFuelNode

```
import com.bht.engr.mac.m3s.input.Entity;
import com.bht.engr.mac.m3s.input.Node;
import com.bht.engr.mac.m3s.input.SimulationEngine;


public final class CheckFuelNode extends Node
{
private DisasterControlNode m_masterNode = null;

//---------------------------------
// Override Node initialize method
//---------------------------------
public void initialize( SimulationEngine simEngine )
 throws Exception
{
 m_masterNode = (DisasterControlNode)simEngine.getNetwork().getNode(
DisasterControlNode.NODE_NAME );

 super.initialize( simEngine );
}

//---------------------------------
// Override Node execute method
//---------------------------------
public void execute( SimulationEngine simEngine, double tNow, Entity entity, String msg )
 throws Exception
{
 m_masterNode.checkFuel( simEngine, entity, tNow );

 super.execute( simEngine, tNow, entity, msg );
}
}
```

CheckFuelNodeCanvas

```
import com.bht.engr.mac.m3s.ui.canvas.GoOnNodeCanvas;

public final class CheckFuelNodeCanvas extends GoOnNodeCanvas
{
}
```

CheckFuelNodeEditDbx

```
import com.bht.engr.mac.m3s.ui.edit.GoOnNodeEditDbx;

public class CheckFuelNodeEditDbx extends GoOnNodeEditDbx
{
private final static String TITLE = "Check Fuel Node";

public CheckFuelNodeEditDbx()
{
super( TITLE );
}
}
```

CTRRand

```
import com.bht.engr.mac.m3s.input.Entity;
import com.bht.engr.mac.m3s.input.SimulationAttributes;
import com.bht.engr.mac.m3s.input.SimulationEngine;
import com.bht.engr.mac.m3s.input.computeValue.M3SComputeValue;
import com.bht.engr.mac.util.MethodException;

public class CTRRand extends M3SComputeValue
{
private SimulationAttributes m_sa;
private String minName;
private String maxName;
private DisasterControlNode m_masterNode;

public CTRRand( String args )
{
// Find index of comma that separates the min and max values
int index = args.indexOf( "," );

// Obtain the min and max value.
minName = args.substring( 0, index ).trim();
maxName = args.substring( index + 1, args.length() ).trim();
}

public void initialize( SimulationEngine simEngine )
throws Exception
{
m_sa = simEngine.getGlobalAttributes();
m_masterNode = (DisasterControlNode)simEngine.getNetwork().getNode(
DisasterControlNode.NODE_NAME );
}

public double getValue( String name, Object obj )
throws Exception
{
try
{
Entity entity = (Entity)obj;

double min = determineValue( minName, entity );
double max = determineValue( maxName, entity );

return m_masterNode.getRandom( min, max );

} catch ( Exception e ) {
throw new MethodException( "CTRRand.getValue",
"Error getting value of - " + name, e );
}
}
```

```
private double determineValue( String name, Entity entity )
throws Exception
{
if ( m_sa.isExist( name ) ) return m_sa.getValue( name );

return entity.getValue( name );
}
}
```

DisasterControlNode

```
import com.bht.engr.mac.m3s.input.*;
import com.bht.engr.mac.m3s.input.priority.*;
import com.bht.engr.mac.m3s.input.reports.*;
import com.bht.engr.mac.m3s.input.util.TimedObject;
import com.bht.engr.mac.util.PrintfFormat;
import static com.bht.engr.mac.util.XMLCreateHelper.*;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Enumeration;
import java.util.List;
import java.util.Random;

public class DisasterControlNode extends Node
{
public final static String NODE_NAME = "DisasterControlNode";

// Air Vehicle Attributes
private final static String DIST_TO_TRAVEL     = "DistanceToTravel";
private final static String VEHICLE_TYPE       = "VehicleType";
public  final static String TAIL_NUMBER        = "TailNumber";
private final static String PAYLOAD            = "Payload";
public  final static String SPEED              = "Speed";
public  final static String PAX_CAPACITY       = "PaxCapacity";
public  final static String LITTER_CAPACITY    = "LitterCapacity";
public  final static String CURRENT_LOCATION   = "CurrentLocation";
public  final static String MISSION_RANGE_LEFT = "MissionRangeLeft";
private final static String MISSION_RANGE      = "MissionRange";
private final static String COMPLETED          = "MissionsCompleted";
private final static String TOTAL_DISTANCE     = "TotalDistanceFlown";
private final static String REFUEL_FLAG        = "RefuelFlag";
private final static String TIME_IN_QUEUE      = "TimeInQueue"; // Only used in this node

// Event Attributes
private final static String EVENT_TYPE   = "EventType";
private final static String PRIORITY     = "Priority";
private final static String LOC_TO_EVT   = "DistanceLocalToEvent";
private final static String EVT_TO_LOC   = "DistanceEventToLocal";
private final static String REM_TO_EVT   = "DistanceRemoteToEvent";
private final static String EVT_TO_REM   = "DistanceEventToRemote";
private final static String TIME_CREATED = "TimeCreated";

private final static String TRANSPORT    = "PayloadToTransport";

private final static String PEOPLE_SAVED = "PeopleToBeSaved";
private final static String AMBULATORY   = "Ambulatory";

// Merged Attributes
private final static String AMB_NUM      = "NumberSavedAmb";
```

```
private final static String AMB_TC       = "TimeCreatedAmb";
private final static String NON_AMB_NUM  = "NumberSavedNonAmb";
private final static String NON_AMB_TC   = "TimeCreatedNonAmb";

private final static String PEOPLE_TO_QUEUE  = "PeopleToQueue";
private final static String MAXIMUM_SR       = "MaximumSearchRange";
private final static String PERCENT_CAPACITY = "PercentCapacity";

// Global Attributes
private final static String SAR_CAPACITY_THRESHOLD = "SARThresholdCapacity";
private final static String SAR_RANGE_THRESHOLD    = "SARRangeRequirement";
private final static String PEOPLE_THRESHOLD       = "PeopleThreshold";
private final static String FIXED_TO_REMOTE        = "DistanceTransferToRemote";
private final static String MIN_LOCAL_RANGE        = "DistanceBetweenBaseAndSiteMinimum";
private final static String MAX_LOCAL_RANGE        =
"DistanceBetweenBaseAndSiteMaximum";

// Locations
public final static int LOCAL_BASE  = 0;
public final static int LOCAL_SITE  = 1;
public final static int REMOTE_SITE = 2;

// Report variables
private final static String EOL = "\n";
private final static String COLON = " : ";
private PrintfFormat pfs = new PrintfFormat( "%50s" );
private PrintfFormat pfd3 = new PrintfFormat( "%3d" );
private PrintfFormat pfd4 = new PrintfFormat( "%4d" );
private PrintfFormat pff3 = new PrintfFormat( "%3.0f" );
private PrintfFormat pff4 = new PrintfFormat( "%4.0f" );
private PrintfFormat pff5 = new PrintfFormat( "%5.0f" );
private PrintfFormat pff10 = new PrintfFormat( "%10.3f" );

private String[] reportAttributes = {
TAIL_NUMBER,
VEHICLE_TYPE,
COMPLETED,
TOTAL_DISTANCE
};

private boolean[] reportAverage = {
false, false, true, true
};

private PrintfFormat[] reportFormats = {
pff4, pff3, pff5, pff5 };

private List<double[]> m_stats = new ArrayList<double[]>();

// ------------------------------------------------
// The air vehicles are not in a queue but a list
```

```
// of available assets
// ------------------------------------------------
private List<Entity> m_vehList = new ArrayList<Entity>();
private List<Entity> m_fixedWingList = new ArrayList<Entity>();
private Queue m_eventQueue = null;
private PeopleContainer m_peopleCont = new PeopleContainer();

// ------------------------------------------------
// Global attributes used for simualtion performance
// ------------------------------------------------
private int peopleThreshold;
private int distToRemote;
private int sarCapacityThreshold;
private double sarRangeThreshold;
private double minLocalRange;
private double maxLocalRange;

// ------------------------------------------------
// Random number generator for events only
// So events will be the same for any fleet mix
// ------------------------------------------------
private Random m_random;

private double m_lastTimeCheckedRemote = 0;

/*************************
 * VEHICLE VERIFICATION
*************************/
private StringBuffer m_vehBuf = new StringBuffer();

//---------------------------------
// Override Node initialize method
//---------------------------------
public void initialize( SimulationEngine simEngine )
throws Exception
{
super.initialize( simEngine );

long seed = simEngine.getRandomGenerator().getRandomObject().nextLong();
m_random = new Random( seed );

m_eventQueue = new Queue();
m_eventQueue.setPriority( new MinPriority( PRIORITY ) );

SimulationAttributes sa = simEngine.getGlobalAttributes();

peopleThreshold = (int)sa.getValue( PEOPLE_THRESHOLD );
distToRemote    = (int)sa.getValue( FIXED_TO_REMOTE );
sarCapacityThreshold = (int)sa.getValue( SAR_CAPACITY_THRESHOLD );
sarRangeThreshold = sa.getValue( SAR_RANGE_THRESHOLD );
minLocalRange = sa.getValue( MIN_LOCAL_RANGE );
```

```
maxLocalRange = sa.getValue( MAX_LOCAL_RANGE );

m_stats.clear();
}

//---------------------------------
// Override Node init method
//---------------------------------
public void init( SimulationEngine simEngine )
throws Exception
{
super.init( simEngine );

m_eventQueue.init();
m_peopleCont.init();
m_vehList.clear();
m_fixedWingList.clear();

m_lastTimeCheckedRemote = 0;

/**************************
 * VEHICLE VERIFICATION
m_vehBuf.append( "  TN VT PC LC  PAY  SPD   MR      TNOW" );
m_vehBuf.append( EOL );
**************************/
}

//---------------------------------------
// Override Node setTimeEnd() method
//---------------------------------------
public void setTimeEnd( double tEnd )
throws Exception
{
super.setTimeEnd( tEnd );

// Always sort the vehicles in order by tail number
Collections.sort( m_vehList, new VehicleTailComparator() );

if ( m_stats.isEmpty() )
{
for ( Entity veh : m_vehList )
{
 double[] vehAttr  = new double[ reportAttributes.length ];
 initList( vehAttr );
 m_stats.add( vehAttr );
}

}

int i = 0;
for ( Entity veh : m_vehList )
```

```
{
double[] vehAttr  = m_stats.get( i );

for ( int n = 0; n < reportAttributes.length; n++ )
{
double val = veh.getValue( reportAttributes[n] );

if ( reportAverage[n] )
{
vehAttr[n] += val;
}
else
{
vehAttr[n] = val;
}
}

i++;
}
}

//-----------------------------------------
// Override Node generateNodeReport() method
//-----------------------------------------
public void generateNodeReport( String type, int iteration, StringBuffer buf )
throws Exception
{
if ( isShowOn() && type.equals( ReportNames.SNAPSHOT ) )
{
double[] totalsGrand = new double[ reportAttributes.length ];
double[] totalsSub   = new double[ reportAttributes.length ];

initList( totalsGrand );
initList( totalsSub );

StringBuffer temp = new StringBuffer();

for ( double[] vehAttr : m_stats )
{
StringBuffer row = new StringBuffer();

for ( int n = 0; n < reportAttributes.length; n++ )
{
String attr = createAttribute( DefaultSnapshotReport.NAME, reportAttributes[n] );

double val = vehAttr[n];
// If this value is an average over all iterations then compute average
if ( reportAverage[n] )
{
val /= iteration;
totalsGrand[n] += val;
```

```
 totalsSub[n] += val;
 }

 createEntryWithAttribute( row, DefaultSnapshotReport.DATA,
 reportFormats[n].sprintf( val ), attr );
}

 createEntry( temp, DefaultSnapshotReport.SET, row.toString() );
}

 StringBuffer row = new StringBuffer();

 for ( int n = 0; n < reportAttributes.length; n++ )
 {
 String attr = createAttribute( DefaultSnapshotReport.NAME, reportAttributes[n] );

 createEntryWithAttribute( row, DefaultSnapshotReport.DATA,
 reportFormats[n].sprintf( totalsGrand[n] ), attr );
}

 createEntry( temp, DefaultSnapshotReport.SET, row.toString() );

 createEntry( buf, DefaultSnapshotReport.SNAP_REPORT, temp.toString() );

 /*************************
  * VEHICLE VERIFICATION
 System.out.print( m_vehBuf );
 *************************/
}

 super.generateNodeReport( type, iteration, buf );
}

 //---------------------------------
 // Override Node execute method
 //---------------------------------
 public void execute( SimulationEngine simEngine, double tNow, Entity entity, String msg )
 throws Exception
 {
 if ( entity.isExist( PEOPLE_TO_QUEUE ) )
 {
 int peopleInQueue = (int)entity.getValue( PEOPLE_TO_QUEUE );
 int ambulatory    = (int)entity.getValue( AMBULATORY );
 double tc         = entity.getValue( TIME_CREATED );
 m_peopleCont.getList().add( new PeopleGroup( peopleInQueue, tc, ambulatory ) );
 //System.out.println( "People added to queue: " + peopleInQueue + " : " + tc );
 processPeople( simEngine, tNow, msg );
 }
 //-----------------------------------
 // Is this a vehicle...add back to list
 //-----------------------------------
```

```
else if ( entity.isExist( VEHICLE_TYPE ) )
{
int vehicleType = (int)entity.getValue( VEHICLE_TYPE );

if ( vehicleType == 7 )
{
m_fixedWingList.add( entity );
processPeople( simEngine, tNow, msg );

/*************************
 * VEHICLE VERIFICATION
writeVehicle( tNow, entity );
*************************/
}
else
{
entity.setValue( TIME_IN_QUEUE, tNow );
m_vehList.add( entity );
processAllEvents( simEngine, tNow, msg );

/*************************
 * VEHICLE VERIFICATION
writeVehicle( tNow, entity );
*************************/
}
}
//----------------------------------
// Then must be Event
//----------------------------------
else
{
//System.out.println( "Event: " + tNow + ":" + entity.getValue( EVENT_TYPE ) );
m_eventQueue.addEntity( entity, tNow );
processAllEvents( simEngine, tNow, msg );
}
}

private void initList( double[] list )
{
for ( int n = 0; n < list.length; n++ ) list[n] = 0;
}

private void processPeople( SimulationEngine simEngine, double tNow, String msg )
throws Exception
{
// If the fixed wing vehicle list is empty...then we are done
// Only fixed wing vehicles will transfer people from local to remote sites
if ( m_fixedWingList.isEmpty() ) return;

while ( ( m_peopleCont.getNumberOfPeople() >= peopleThreshold ) &&
!m_fixedWingList.isEmpty() )
```

```
{
// Fill the next available fixed wing with people
Entity fixedWing = m_fixedWingList.get( 0 );
m_fixedWingList.remove( 0 );
int paxCapacity = (int)fixedWing.getValue( PAX_CAPACITY );

boolean done = false;
int numLoaded = 0;
int numAmb = 0;
int numNonAmb = 0;
double aveTCAmb = 0;
double aveTCNonAmb = 0;

while ( !done && ( m_peopleCont.getNumberOfPeople() > 0 ) )
{
PeopleGroup pg = m_peopleCont.getList().get( 0 );
int num = pg.getNumber();

// Did we reach the capacity of the fixed wing?
// Hence we did not move this entire group
if ( ( numLoaded + num ) > paxCapacity )
{
done = true;
num = paxCapacity - numLoaded;
pg.setNumber( pg.getNumber() - num );
}
// Otherwise we did move this entire group
// Hence we need to remove from queue
else
{
m_peopleCont.getList().remove( 0 );
}

numLoaded += num;

if ( pg.isAmbulatory() )
{
//System.out.println( num + " : " + pg.getTimeCreated() + " : " + tNow );
numAmb += num;
aveTCAmb += ( num * pg.getTimeCreated() );
}
else
{
numNonAmb += num;
aveTCNonAmb += ( num * pg.getTimeCreated() );
}
}

if ( numAmb > 0 ) aveTCAmb /= numAmb;
if ( numNonAmb > 0 ) aveTCNonAmb /= numNonAmb;
```

```
 fixedWing.setValue( EVENT_TYPE, 5 );
 fixedWing.setValue( DIST_TO_TRAVEL, distToRemote );
 fixedWing.setValue( PEOPLE_SAVED, (double)numLoaded );
 setPeopleAttributes( fixedWing, (double)numAmb, aveTCAmb, (double)numNonAmb,
aveTCNonAmb );
//System.out.println( "Sending fixed wing: " + numLoaded + " : " + aveTCAmb + " : " + ( tNow -
aveTCAmb ) );

 super.execute( simEngine, tNow, fixedWing, msg );
}
}

private void processAllEvents( SimulationEngine simEngine, double tNow, String msg )
throws Exception
{
// ###################################################################
// BEGIN: This section of code is to verify the event ordering
// ###################################################################
/*****
if ( m_eventQueue.getList().size() == 20 )
{
for ( int n = 0; n < m_eventQueue.getList().size(); n++ )
{
 TimedObject tObject = (TimedObject)m_eventQueue.getList().get( n );
 Entity event = (Entity)tObject.getObject();
 int type = (int)event.getValue( EVENT_TYPE );
 double tc = event.getValue( TIME_CREATED );
 System.out.println( "EVENT: " + type + " created at " + tc );
}
}
*******/
// ###################################################################
// END: This section of code is to verify the event ordering
// ###################################################################

// If the vehicle list is empty...then we are done
if ( m_vehList.isEmpty() ) return;

// ----------------------------------------
// Keep generating sub-events while we have
// assets to meet the requirements.
// We need to process all events everytime.
// Higher priority events may not get
// processed because too far or does not
// meet SAR thresholds.
// ----------------------------------------
for ( int n = 0; n < m_eventQueue.getList().size(); n++ )
{
 TimedObject tObject = (TimedObject)m_eventQueue.getList().get( n );

 // Was this event requirements met?
```

```
if ( processEvent( simEngine, tNow, msg, tObject ) )
{
 m_eventQueue.getList().remove( n );
 n--;
}
}
}

private boolean processEvent( SimulationEngine simEngine, double tNow, String msg,
TimedObject tObject )
 throws Exception
{
 checkRemoteVehicles( tNow );

 Entity event = (Entity)tObject.getObject();

 //-------------------------------------------------
 // Common initialization for all event types
 // 1. Obtain event type and time created
 // 2. Initialize the comparators with
 //    - Range local to event
 //    - Range remote to event
 // 3. Determine minimum distance home (local)
 //-------------------------------------------------
 int type = (int)event.getValue( EVENT_TYPE );
 double tc = event.getValue( TIME_CREATED );
//System.out.println( "processEvent: " + type );

 double localToEvent = event.getValue( LOC_TO_EVT );
 double remoteToEvent = event.getValue( REM_TO_EVT );
 double rangeHome = event.getValue( EVT_TO_LOC );

 // Initialize static variables
 VehicleRangeComparator.setLocalToEvent( localToEvent );
 VehicleRangeComparator.setRemoteToEvent( remoteToEvent );
 VehicleRangeComparator vrc = new VehicleRangeComparator();

 VehicleTOSComparator.setLocalToEvent( localToEvent );
 VehicleTOSComparator.setRemoteToEvent( remoteToEvent );
 VehicleTOSComparator.setRangeHome( rangeHome );
 VehicleTOSComparator.setRefuelRange( maxLocalRange );
 VehicleTOSComparator vtc = new VehicleTOSComparator();

 // ---------------------------------------------------------------
 // SAR event (3)
 // ---------------------------------------------------------------
 // Find search and rescue vehicle that meets the following requirements:
 // 1. Satisfies minimum pax capacity
 // 2. Satisfies minimum TOS
 // ---------------------------------------------------------------
```

```
if ( type == 3 )
{
// Get list of all vehicles which meet the SAR capacity threshold
List<Entity> capReducedList =
VehicleCapacityComparator.reduceListByThreshold( m_vehList, sarCapacityThreshold );

//System.out.println( "SAR list: " + capReducedList.size() );
if ( capReducedList.isEmpty() ) return false;

// Sort list by mission range remaining
//Collections.sort( capReducedList, vtc );
//Instead of sort find maximum

double tos = 0;
Entity veh = capReducedList.get( 0 );

for ( Entity ent : capReducedList )
{
double t = VehicleTOSComparator.determineTOS( ent );
if ( t > tos )
{
tos = t;
veh = ent;
}
}

// Does this vehicle meet the minimum TOS threshold
if ( tos < sarRangeThreshold ) return false;

Entity entity = initializeAndCloneVehicle( simEngine, event, veh, localToEvent, remoteToEvent
);

entity.setValue( MAXIMUM_SR, tos );
// If people to be saved is greater than the capacity
// then we leave them behind and do not account for them
double pax = veh.getValue( PAX_CAPACITY );
double ps = event.getValue( PEOPLE_SAVED );
entity.setValue( PERCENT_CAPACITY, ps / pax );

setPeopleAttributes( entity, 0, 0, 0, 0 );

super.execute( simEngine, tNow, entity, msg );

return true;
}

// -----------------------------------------------------------------
// Evacuation (1) and Medevac (2) event
// -----------------------------------------------------------------
String vehAttribute = PAX_CAPACITY;
String evtAttribute = PEOPLE_SAVED;
```

```
if ( type == 2 )
{
vehAttribute = LITTER_CAPACITY;
}

// ---------------------------------------------------------------------
// Transport (4) event
// ---------------------------------------------------------------------
if ( type == 4 )
{
vehAttribute = PAYLOAD;
evtAttribute = TRANSPORT;
}

List<Entity> rangeReducedList = null;
boolean remoteList = false;

// If evacuation first see if any air vehicles can go to the remote
if ( type == 1 )
{
double rangeRemote = event.getValue( EVT_TO_REM );
rangeReducedList = createSortedMissionList( vehAttribute, rangeRemote );
if ( !rangeReducedList.isEmpty() ) remoteList = true;
}

if ( !remoteList )
{
rangeReducedList = createSortedMissionList( vehAttribute, rangeHome );
}

// People to be saved or supplies to be transported
double evtRequirement = event.getValue( evtAttribute );

//System.out.println( "Available AC: " + rangeReducedList.size() + " " + evtAttribute + ": " +
evtRequirement );
// create sub-events
boolean done = false;

while ( !done && !rangeReducedList.isEmpty() )
{
Entity veh = getOptimalVehicle( rangeReducedList, vehAttribute, evtRequirement );
rangeReducedList.remove( veh );

Entity entity = initializeAndCloneVehicle( simEngine, event, veh, localToEvent, remoteToEvent
);

// Assume we can take all the people left in this event
double vehCapacity = veh.getValue( vehAttribute );
double num = evtRequirement;
```

```
//System.out.println( (int)veh.getValue( TAIL_NUMBER ) + " : " + vehAttribute + ": " +
vehCapacity  );

 // If there are too many left then we can only take
 // up to our capacity and we decrement the number of
 // people left
 if ( vehCapacity < evtRequirement )
 {
 num = vehCapacity;
 evtRequirement -= num;
 }
 else
 {
 num = evtRequirement;
 evtRequirement = 0;
 done = true;
 }

 entity.setValue( evtAttribute, num );
 entity.setValue( PERCENT_CAPACITY, num / vehCapacity );

 if ( type == 2 ) // medevac
 {
 setPeopleAttributes( entity, 0, 0, num, tc );
 }
 else if ( ( type == 1 ) || ( type == 3 ) ) // evac and sar  (actually type 3 handled above)
 {
 setPeopleAttributes( entity, num, tc, 0, 0 );
 }

 super.execute( simEngine, tNow, entity, msg );

 // If we used all the vehicles that can take people to remote
 // and still need more, then generate list to local site
 if ( remoteList && rangeReducedList.isEmpty() )
 {
 rangeReducedList = createSortedMissionList( vehAttribute, rangeHome );
 remoteList = false;
 }
 }

 // Ran out of vehicles before meeting requirements for the event
 // reset the requirement
 if ( !done )
 {
 event.setValue( evtAttribute, evtRequirement );
 return false;
 }

 return true;
 }
```

```
private void checkRemoteVehicles( double tNow )
throws Exception
{
// If we just checked vehicles at remote, skip this check
if ( ( tNow - m_lastTimeCheckedRemote ) < 1 ) return;

m_lastTimeCheckedRemote = tNow;

// check to see if a vehicle has been stuck at a remote site
// if so, move to local base
for ( Entity veh : m_vehList )
{
int loc = (int)veh.getValue( CURRENT_LOCATION );
double waitTime = tNow - veh.getValue( TIME_IN_QUEUE );

if ( ( loc == REMOTE_SITE ) && ( waitTime > 4 ) )
{
veh.setValue( CURRENT_LOCATION, (double)LOCAL_BASE );
}
}
}

private List<Entity> createSortedMissionList( String vehAttribute, double rangeHome )
{
// Reduce list of vehicles to those that can perform the mission
List<Entity> rangeReducedList = VehicleRangeComparator.performMissionList( m_vehList,
rangeHome );

// Sort the remaining vehicle list in order by pax/litter capacity or pay load
// Descending order highest capacity to lowest capacity
VehicleCapacityComparator vcc = new VehicleCapacityComparator( vehAttribute );
Collections.sort( rangeReducedList, vcc );

return rangeReducedList;
}

private Entity initializeAndCloneVehicle( SimulationEngine simEngine, Entity event, Entity veh,
double localToEvent, double remoteToEvent )
throws Exception
{
// Remove vehicle as we are sending on mission
m_vehList.remove( veh );

// Set distance to travel attribute
int loc = (int)veh.getValue( CURRENT_LOCATION );

if ( loc == REMOTE_SITE )
{
veh.setValue( DIST_TO_TRAVEL, remoteToEvent );
}
```

```
else
{
veh.setValue( DIST_TO_TRAVEL, localToEvent );
}

// Clone the veh entity and merge with event attributes
Entity entity = (Entity)veh.clone();
Enumeration e = event.keys();
while ( e.hasMoreElements() )
{
String key = (String)e.nextElement();
entity.setValue( key, event.getValue( key ) );
}

// If the air vehicle is currently at the single location site
// then ensure the event to local is the same as local to event
if ( loc == LOCAL_SITE )
{
entity.setValue( EVT_TO_LOC, entity.getValue( LOC_TO_EVT ) );
}

// ----------------------------------------------------------------
// Finally, if starting from local base, ensure that the local
// distances do not cause a triangle inequality.
// Also, make sure distances total to at least the minimum distance
// between local site and local base
// ----------------------------------------------------------------
if ( loc == LOCAL_BASE )
{
double eventToLocal = entity.getValue( EVT_TO_LOC );

if ( ( localToEvent - eventToLocal ) > maxLocalRange )
{
entity.setValue( EVT_TO_LOC, localToEvent - maxLocalRange );
}

if ( ( localToEvent + eventToLocal ) < minLocalRange )
{
entity.setValue( EVT_TO_LOC, minLocalRange - localToEvent );
}
}

return entity;
}

private void setPeopleAttributes( Entity entity, double numAmb, double tcAmb, double
numNonAmb, double tcNonAmb )
throws Exception
{
entity.setValue( AMB_NUM, numAmb );
entity.setValue( AMB_TC, tcAmb );
```

```
entity.setValue( NON_AMB_NUM, numNonAmb );
entity.setValue( NON_AMB_TC, tcNonAmb );
}

private Entity getOptimalVehicle( List<Entity> list, String vehAttribute, double evtRequirement )
throws Exception
{
Entity optVehicle = list.get( 0 );

for ( Entity veh : list )
{
double val = veh.getValue( vehAttribute );
//System.out.println( val );
// If this vehicle meets the requirement then send in place
// of vehicle with more capacity.
if ( val >= evtRequirement )
{
//System.out.println( "Optimal: " + veh.getValue( TAIL_NUMBER ) );
optVehicle = veh;
}
else
{
return optVehicle;
}
}

return optVehicle;
}

/*************************
 * VEHICLE VERIFICATION
*************************/
private void writeVehicle( double tNow, Entity veh )
throws Exception
{
if ( tNow > 1 ) return;

int vt = (int)veh.getValue( VEHICLE_TYPE );
m_vehBuf.append( pfd4.sprintf( (int)veh.getValue( TAIL_NUMBER ) ) );
m_vehBuf.append( pfd3.sprintf( vt ) );
m_vehBuf.append( pfd3.sprintf( (int)veh.getValue( PAX_CAPACITY ) ) );
if ( vt != 7 ) m_vehBuf.append( pfd3.sprintf( (int)veh.getValue( LITTER_CAPACITY ) ) );
else m_vehBuf.append( "   " );
if ( vt != 7 ) m_vehBuf.append( pff5.sprintf( veh.getValue( PAYLOAD ) ) );
else m_vehBuf.append( "     " );
m_vehBuf.append( pff5.sprintf( veh.getValue( SPEED ) ) );
if ( vt != 7 ) m_vehBuf.append( pff5.sprintf( veh.getValue( MISSION_RANGE ) ) );
else m_vehBuf.append( "     " );
m_vehBuf.append( pff10.sprintf( tNow ) );
m_vehBuf.append( EOL );
}
```

```
// Used by CheckFuelNode
public void checkFuel( SimulationEngine simEngine, Entity veh, double tNow )
 throws Exception
{
   // Roll die for distance to travel
 double range = simEngine.getRandomGenerator().getRandom( minLocalRange,
maxLocalRange );

 veh.setValue( DIST_TO_TRAVEL, range );
 veh.setValue( REFUEL_FLAG, 1 ); // Cause refuel
}

// Used by CTRRand
public double getRandom( double min, double max )
{
 return min + m_random.nextDouble() * ( max - min );
}
}
```

DisasterControlNodeCanvas

```
import com.bht.engr.mac.m3s.ui.canvas.GoOnNodeCanvas;

public class DisasterControlNodeCanvas extends GoOnNodeCanvas
{
}
```

DisasterControlNodeEditDbx

```
import com.bht.engr.mac.m3s.ui.edit.GoOnNodeEditDbx;

public class DisasterControlNodeEditDbx extends GoOnNodeEditDbx
{
private final static String TITLE = "Disaster Control Node";

public DisasterControlNodeEditDbx()
{
super( TITLE );
}
}
```

DisasterStencil

```
import com.bht.engr.mac.m3s.ui.stencil.NetworkStencil;

public final class DisasterStencil extends NetworkStencil
{
public Object createNetworkObject()
 throws Exception
{
 DisasterControlNode node = new DisasterControlNode();
 // This needs to be unique in the network
 node.setName( DisasterControlNode.NODE_NAME );

 getFacade().getNetwork().add( node );

 return node;
}
}
```

EventNextTime

```
import com.bht.engr.mac.m3s.input.*;
import com.bht.engr.mac.m3s.input.computeValue.ComputeValueFactory;
import com.bht.engr.mac.math.condition.SimpleExpression;
import com.bht.engr.mac.util.MethodException;
import static com.bht.engr.mac.util.XMLCreateHelper.*;
import static com.bht.engr.mac.util.XMLTopHandler.*;
import org.xml.sax.SAXException;
import org.xml.sax.Attributes;
import java.util.ArrayList;
import java.util.List;

public final class EventNextTime extends IntervalNextTime implements NextEntity
{
 private final static String MINIMUM = "MinimumQuantity";
 private final static String MAXIMUM = "MaximumQuantity";
 private final static String CREATE_LEN = "CreationLength";
 private final static String EVT_RQT = "EventRequirement";
 private final static String DEFAULT = "0";

 private DisasterControlNode m_masterNode = null;
 private List<EventInfo> m_list = new ArrayList<EventInfo>();
 private SimpleExpression m_minExpr = new SimpleExpression( DEFAULT );
 private SimpleExpression m_maxExpr = new SimpleExpression( DEFAULT );
 private SimpleExpression m_createExpr = new SimpleExpression( DEFAULT );
 private double m_creationLength;
 private double m_min;
 private double m_max;
 private double deltaT;
 private double eventDelayTime;

 public void initialize( SimulationEngine simEngine )
 throws Exception
 {

 super.initialize( simEngine );

 //-------------------------------------
 // Need to re-evaluate the expression
 //-------------------------------------
 m_minExpr.initialize( simEngine.getComputeValueFactory() );
 m_maxExpr.initialize( simEngine.getComputeValueFactory() );
 m_createExpr.initialize( simEngine.getComputeValueFactory() );

 m_masterNode = (DisasterControlNode)simEngine.getNetwork().getNode(
DisasterControlNode.NODE_NAME );

 m_creationLength = getCreationLengthValue( null );
 m_min = getMinimumRequirementValue( null );
```

```
m_max = getMaximumRequirementValue( null );
deltaT = getTimeBetweenCreationsValue( null );
// Delay creating events until this time
eventDelayTime = getTimeOfFirstCreationValue( null );
}

public void init()
{
m_list.clear();

int total = getMaximumCreationsValue();
double eventTime = m_creationLength - eventDelayTime;
double b = 2 * (double)total / eventTime;
double slope = -b / eventTime;
double time = deltaT / 2;

while ( time < eventTime )
{
 double y = time * slope + b;
 int num = (int)( y * deltaT );
 createEvents( num, eventDelayTime + time - deltaT/2, deltaT );
 time += deltaT;
}
}

public double getNextTime( SimulationEngine simEngine, Entity entity )
throws Exception
{
if ( !isMore() )
{
 throw new MethodException( "EventNextTime.getNextTime",
 "Reached maximum number of creations - No more" );
}

EventInfo ei = m_list.get( 0 );
return ei.getTime();
}

public Entity getNextEntity()
{
EventInfo ei = m_list.remove( 0 );
Entity entity = new Entity();
entity.setValue( EVT_RQT, (double)ei.getNumber() );

return entity;
}

// Override CountNextTime.isMore()
public boolean isMore()
{
return !m_list.isEmpty();
```

```
}

private void createEvents( int num, double time, double deltaT )
{
if ( num == 0 ) return;

List<Integer> tmp = new ArrayList<Integer>();

while ( num > 0 )
{
int val = (int)m_masterNode.getRandom( m_min, m_max );

if ( val > num ) val = num;

// Do not create an event with a requirement of zero
if ( val > 0 ) tmp.add( new Integer( val ) );

num -= val;
}

double tbe = deltaT / tmp.size();

for ( int n = 0; n < tmp.size(); n++ )
{
Integer i = tmp.get( n );
double t = time + n * tbe;
m_list.add( new EventInfo( t, i.intValue() ) );
}
}

public void startElement( String namespaceURI,
String localName, String qName, Attributes attr )
throws SAXException
{
super.startElement( namespaceURI, localName, qName, attr );

try
{
if ( localName.equals( MINIMUM ) )
{
String val = getValue( attr, M3SInputHelper.VALUE );

if ( val != null )
{
setMinimumRequirement( val );
}
}
else if ( localName.equals( MAXIMUM ) )
{
String val = getValue( attr, M3SInputHelper.VALUE );
```

```
if ( val != null )
{
 setMaximumRequirement( val );
}
}
else if ( localName.equals( CREATE_LEN ) )
{
 String val = getValue( attr, M3SInputHelper.VALUE );

 if ( val != null )
 {
  setCreationLength( val );
 }
}

} catch ( Exception e ) {
 throw new SAXException( e.getMessage() );
}
}

public void toXML( StringBuffer buf )
{
 super.toXML( buf );

 createEntryWithAttribute( buf, MINIMUM, null,
 createAttribute( M3SInputHelper.VALUE, getMinimumRequirement() ) );

 createEntryWithAttribute( buf, MAXIMUM, null,
 createAttribute( M3SInputHelper.VALUE, getMaximumRequirement() ) );

 createEntryWithAttribute( buf, CREATE_LEN, null,
 createAttribute( M3SInputHelper.VALUE, getCreationLength() ) );
}

public String toString()
{
 return "Event Next Time";
}

//----------------------
// Accessor Methods
//----------------------
public double getCreationLengthValue( Entity entity )
 throws Exception
{
 return m_createExpr.compute( entity );
}

public String getCreationLength()
{
 return m_createExpr.getExpression();
```

```
}

public void setCreationLength( String val )
{
m_createExpr.setExpression( val );
dirty();
}

public double getMinimumRequirementValue( Entity entity )
throws Exception
{
return m_minExpr.compute( entity );
}

public String getMinimumRequirement()
{
return m_minExpr.getExpression();
}

public void setMinimumRequirement( String val )
{
m_minExpr.setExpression( val );
dirty();
}

public double getMaximumRequirementValue( Entity entity )
throws Exception
{
return m_maxExpr.compute( entity );
}

public String getMaximumRequirement()
{
return m_maxExpr.getExpression();
}

public void setMaximumRequirement( String val )
{
m_maxExpr.setExpression( val );
dirty();
}

class EventInfo
{
private double m_time;
private int m_num;

public EventInfo( double time, int num )
{
m_time = time;
m_num = num;
```

```
}

 public double getTime() { return m_time; }
 public double getNumber() { return m_num; }
}
}
```

EventNextTimeEditDbx

```
import com.bht.engr.mac.m3s.ui.edit.*;
import com.bht.engr.mac.utilx.TextFieldPanel;
import javax.swing.BoxLayout;
import javax.swing.JPanel;
import javax.swing.BorderFactory;

public final class EventNextTimeEditDbx extends IntervalNextTimeEditDbx
{
private final static boolean[] s_tfEditable = { true, true, true };
private TextFieldPanel m_evtTF = null;

public EventNextTimeEditDbx()
{
String[] tfLabel = {
"Total Creation Length (hrs)",
"Minimum Requirement",
"Maximum Requirement" };

String[] tfToolTip = {
"Enter the total creation length in hours.",
"Enter the minimum requirements.",
"Enter the maximum requirements." };

m_evtTF = new TextFieldPanel( null, tfLabel, null,
s_tfEditable, tfToolTip, TextFieldPanel.LABEL_LEFT );

m_evtTF.getTextField( 0 ).getDocument().addDocumentListener( this );
m_evtTF.getTextField( 1 ).getDocument().addDocumentListener( this );
m_evtTF.getTextField( 2 ).getDocument().addDocumentListener( this );

JPanel panel = new JPanel();
panel.setBorder( BorderFactory.createCompoundBorder(
BorderFactory.createTitledBorder( "Event Next Time" ),
BorderFactory.createEmptyBorder( 5, 5, 5, 5 ) ) );
panel.setLayout( new BoxLayout( panel, BoxLayout.X_AXIS ) );
panel.add( m_evtTF );

getMainPanel().add( panel );
}

public void apply()
throws Exception
{
super.apply();

EventNextTime nt = (EventNextTime)getBusinessObject();

nt.setCreationLength( m_evtTF.getText( 0 ) );
```

```
nt.setMinimumRequirement( m_evtTF.getText( 1 ) );
nt.setMaximumRequirement( m_evtTF.getText( 2 ) );
}

protected void update()
throws Exception
{
super.update();

EventNextTime nt = (EventNextTime)getBusinessObject();

m_evtTF.setText( 0, nt.getCreationLength() );
m_evtTF.setText( 1, nt.getMinimumRequirement() );
m_evtTF.setText( 2, nt.getMaximumRequirement() );
}
}
```

PeopleContainer

```
import java.util.ArrayList;
import java.util.List;

public final class PeopleContainer
{
private List<PeopleGroup> m_list = new ArrayList<PeopleGroup>();

public int getNumberOfPeople()
{
int val = 0;

for ( PeopleGroup grp : getList() )
{
 val += grp.getNumber();
}

return val;
}

public void init()
{
getList().clear();
}

public List<PeopleGroup> getList()
{
return m_list;
}
}
```

PeopleGroup

```
public final class PeopleGroup
{
private int m_number;
private double m_timeCreated;
private boolean m_isAmbulatory;

public PeopleGroup( int num, double tc, int amb )
{
setNumber( num );
setTimeCreated( tc );

if ( amb == 1 ) m_isAmbulatory = true;
else            m_isAmbulatory = false;
}

public int getNumber() { return m_number; }
public void setNumber( int val ) { m_number = val; }
public double getTimeCreated() { return m_timeCreated; }
public void setTimeCreated( double val ) { m_timeCreated = val; }
public boolean isAmbulatory() { return m_isAmbulatory; }
}
```

VehicleCapacityComparator

```
import com.bht.engr.mac.m3s.input.Entity;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;

public final class VehicleCapacityComparator implements Comparator<Entity>
{
private String m_attribute;

public VehicleCapacityComparator( String val )
{
m_attribute = val;
}

public int compare( Entity veh1, Entity veh2 )
{
double pax1 = determineCapacity( veh1, m_attribute );
double pax2 = determineCapacity( veh2, m_attribute );

return (int)(pax2 - pax1);
}

public boolean equals( Object obj )
{
return false;
}

public static List<Entity> reduceListByThreshold( List<Entity> vehs, double
sarCapacityThreshold )
{
List<Entity> list = new ArrayList<Entity>();

for ( Entity veh : vehs )
{
if ( determineCapacity( veh, DisasterControlNode.PAX_CAPACITY ) <= sarCapacityThreshold )
{
list.add( veh );
}
}

return list;
}

private static double determineCapacity( Entity veh, String attr )
{
try
{
return veh.getValue( attr );
```

```
} catch ( Exception e ) {
 System.out.println( "VehicleCapacityComparator.determineCapacity: " + e.getMessage() );
}

 return 0;
}
}
```

VehicleRangeComparator

```
import com.bht.engr.mac.m3s.input.Entity;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;

public final class VehicleRangeComparator implements Comparator<Entity>
{
private static double m_locToEvent;
private static double m_remToEvent;

public int compare( Entity veh1, Entity veh2 )
{
// Determine after flying to the event which vehicle has the most range left
double mrl1 = determineMissionRangeLeft( veh1 );
double mrl2 = determineMissionRangeLeft( veh2 );

return (int)(mrl2 - mrl1);
}

public boolean equals( Object obj )
{
return false;
}

public static List<Entity> performMissionList( List<Entity> vehs, double rangeHome )
{
List list = new ArrayList<Entity>();

for ( Entity veh: vehs )
{
double mrl = determineMissionRangeLeft( veh );

// If this vehicle can fly from current location to
// the event and at least to a local site then add
// to the list of vehicles which can perform task
if ( ( mrl - rangeHome ) > 0 )
{
list.add( veh );
}
}

return list;
}

public static double determineMissionRangeLeft( Entity veh )
{
try
{
```

```
// Current Location
int cl = (int)veh.getValue( DisasterControlNode.CURRENT_LOCATION );

// Mission Range
double mr = veh.getValue( DisasterControlNode.MISSION_RANGE_LEFT );

// Vehicle currently at remote site
if ( cl == DisasterControlNode.REMOTE_SITE )
{
 return mr - m_remToEvent;
}
// Vehicle currently at local site/base
else
{
 return mr - m_locToEvent;
}

} catch ( Exception e ) {
 System.out.println( "VehicleRangeComparator.determineMissionRangeLeft: " + e.getMessage()
);
}

return 0;
}

public static void setLocalToEvent( double val )
{
m_locToEvent = val;
}

public static void setRemoteToEvent( double val )
{
m_remToEvent = val;
}
}
```

VehicleTailComparator

```
import com.bht.engr.mac.m3s.input.Entity;
import java.util.Comparator;

public final class VehicleTailComparator implements Comparator<Entity>
{
public int compare( Entity veh1, Entity veh2 )
{
 try
 {
 // Order vehicles by tail number
 double tn1 = veh1.getValue( DisasterControlNode.TAIL_NUMBER );
 double tn2 = veh2.getValue( DisasterControlNode.TAIL_NUMBER );

 return (int)(tn1 - tn2);

 } catch ( Exception e ) {
 System.out.println( "VehicleTailComparator: " + e.getMessage() );
 }

 return 0;
 }

 public boolean equals( Object obj )
 {
 return false;
 }
}
```

VehicleTOSComparator

```
import com.bht.engr.mac.m3s.input.Entity;
import java.util.Comparator;

public final class VehicleTOSComparator implements Comparator<Entity>
{
private static double m_locToEvent;
private static double m_remToEvent;
private static double m_rangeHome;
private static double m_refuelRange;

public int compare( Entity veh1, Entity veh2 )
{
// Determine after flying to the event which vehicle has the most range left
double mrl1 = determineTOS( veh1 );
double mrl2 = determineTOS( veh2 );

return (int)(mrl2 - mrl1);
}

public boolean equals( Object obj )
{
return false;
}

public static double determineTOS( Entity veh )
{
try
{
// Current Location
int cl = (int)veh.getValue( DisasterControlNode.CURRENT_LOCATION );

// Mission Range Left
double mr = veh.getValue( DisasterControlNode.MISSION_RANGE_LEFT );

// Vehicle currently at remote site
if ( cl == DisasterControlNode.REMOTE_SITE )
{
return ( mr - m_remToEvent - m_rangeHome - m_refuelRange );
}
// Vehicle currently at local site/base
else
{
return ( mr - m_locToEvent - m_rangeHome - m_refuelRange );
}

} catch ( Exception e ) {
System.out.println( "VehicleTOSComparator.determineTOS: " + e.getMessage() );
}
```

```
return 0;
}

public static void setLocalToEvent( double val )
{
m_locToEvent = val;
}

public static void setRemoteToEvent( double val )
{
m_remToEvent = val;
}

public static void setRangeHome( double val )
{
m_rangeHome = val;
}

public static void setRefuelRange( double val )
{
m_refuelRange = val;
}
}
```