DRAFT

Report #05-028

SAAS: Simulation Analysis of Aviation Security
Year One Report

Yao, K. & Kadam, S.

CREATE REPORT
Under FEMA Grant EMW-2004-GR-0112

November 2005

Center for Risk and Economic Analysis of Terrorism Events
University of Southern California
Los Angeles, California

This research was supported by the United States Department of Homeland Security through the Center for Risk and Economic Analysis of Terrorism Events (CREATE) under grant number N00014-05-0630. However, any opinions, findings, and conclusions or recommendations in this document are those of the authors and do not necessarily reflect views of the United States Department of Homeland Security.

SAAS: Simulation Analysis of Aviation Security

Year One Report

CREATE Report

November 2005

Ke-Thia Yao, Sandeep Kadam

Information Sciences Institute

University of Southern California

Abstract

This report describes the SAAS project's year one effort to implement an integrated simulation-based decision support tool framework to enable emergency personnel to rapidly respond to terrorists or to other emergencies on a regional or nation-wide basis. Using aviation security domain as a test bed, the SAAS tool framework integrates together multiple models and software components to enable these emergency personnel to explore possible responses to threats and to evaluate the risk and cost tradeoffs associated with the responses.

Acknowledgements

This research was supported by the United States Department of Homeland Security through the Center for Risk and Economic Analysis of Terrorism Events (CREATE), grant number EMW-2004-GR-0112. However, any opinions, findings, and conclusions or recommendations in this document are those of the author (s) and do not necessarily reflect views of the U.S. Department of Homeland Security.

1 Introduction

The SAAS project is developing an integrated simulation-based decision support tool framework to enable emergency personnel to rapidly respond to terrorists or to other emergencies on a regional or nation-wide basis. Using aviation security domain as a test bed, the SAAS tool framework integrates together multiple models and software components to enable these personnel to evaluate the range of risks and economic costs that accompany aviation threat scenarios requiring rerouting of commercial aircrafts.

Within CREATE, and within the wider aviation security community, researchers are developing models of different aspects of external threats to aviation security. These models include aircraft and airspace simulations; aircraft attack scenarios (e.g., weapon type, attack patterns); counter-measure and response strategies (e.g., aircraft engineering modifications, fire reduction, pilot protection, pilot training, airport fortification); risk models (aircraft hit probability models); economic models (aircraft diversion costs); and consequence models (structural aircraft survivability models). The goal of SAAS is to integrate these individual models to provide a holistic framework 1) to exercise these models and 2) to use them to assess the relative benefits of possible responses to threat, and then to recommend appropriate strategies. There are a number of possible uses of this integrated model framework:

- Experiential knowledge. The framework provides a means for experimenting with and learning from response scenarios, similar to the way, for example, groups of people practice emergency response scenarios (e.g., for cities) to discover what works and what doesn't. SAAS scenarios can involve hundreds or thousands of aircraft, so computer simulation is an essential component of these scenarios. Simulation is probably the only way to get a comprehensive grasp on the kinds of problems and solutions that would occur under terrorist threat conditions.
- Training aid. Emergency planners can use SAAS interactively to run against realistic scenarios, refining their own ability to respond quickly and confidently.
- Emergency assistant. In a true emergency, we envision SAAS, or something like it, as an adviser than can quickly provide a semi-optimal schedule for rerouting aircraft, that planners could use as a starting point for directing aircraft to alternate destinations.
- Cost/benefit optimizer. Fortified airports offer a possible means for significantly increasing our ability to find safe havens for aircraft under certain terrorist threat scenarios. The SAAS simulator can be used to select the quantity, location, and degree of fortification of airports, assessing the risk and cost of large numbers of possible fortification strategies and aircraft diversion strategies.

Another goal of SAAS is related to software engineering. By building functional prototypes we hope understand the requirements and software components needed to build a general risk analysts workbench for domains other then aviation. Functionality of software components currently incorporated into the SAAS framework includes:

- Event-based simulation that model air traffic and air space behavior
- Simulation logger to capture the simulation events for analysis

- Data sources to appropriately initialize the simulator
- Scenario definition to describe the aviation threat, counter-measures and response Strategies
- Risk and cost metrics to evaluate the outcome
- Optimization to tune counter-measures and response strategies
- Visualization to aid in the understanding of the results.

2 External Aviation Threat Scenarios

A SAAS simulation begins with a typical commercial aviation scenario – approximately 3000 commercial aircraft are airborne flying between cities in the United States. Until a threat occurs, each of these aircraft follows the prescribed route specified in a daily schedule. For our test bed simulations, the simulator assumes that aircraft are proceeding at constant speed from source to destination as it updates the position of each aircraft. In a live simulation, SAAS would be using live data to track the position of each aircraft. Figure 1 below provides an illustration of a SAAS display showing the positions of a subset of the airborne aircraft.

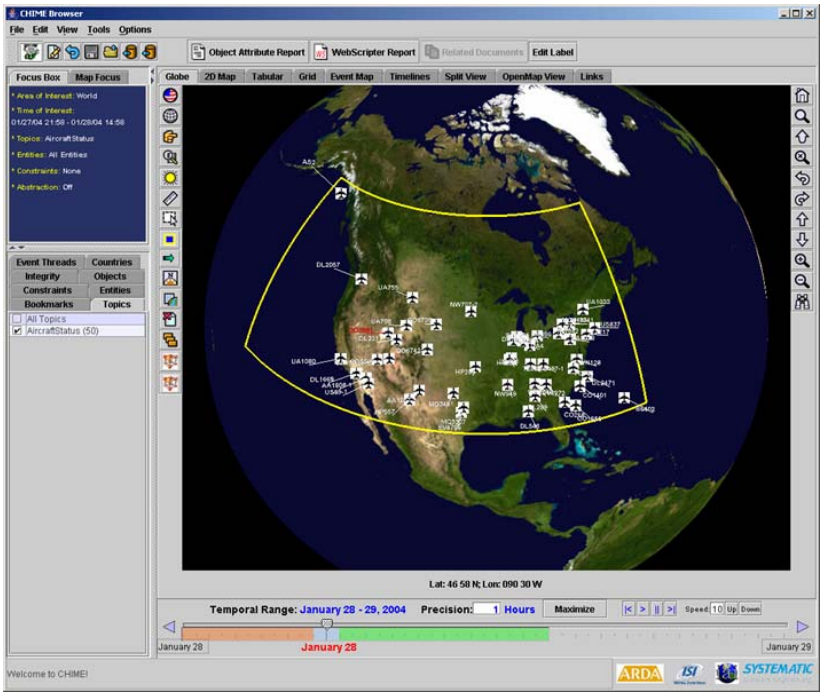

Figure 1 Current location of aircrafts

Based upon threat models developed by CREATE researchers Vicki Bier and Terry O'Sullivan, we parameterized the threat models along these dimensions:

1. Types of attack
   - Single attack on an individual airport
   - Attacks at multiple airports
2. Types of weapon with varying range capabilities
   - MANPADS SA-7B with maximum effective range of 5500m

- Barrett M82 HP Rifle with maximum effective range of 1800m
- AK-47 Assault Rifle with maximum effective range of 300m

3. Types of launch site
   - Uniformly distributed around the airport
   - Normally distributed around the airport (clustered nearer the airport)
   - Normally distributed around the flight path
4. Presence of countermeasures
   - Fortified airports and/or other countermeasures that reduce hit probability
   - Without countermeasures

Response strategies to threat include:

1. Prevent all aircrafts from taking off
2. Ground all airborne aircrafts
3. Divert airborne aircrafts away from threatened airports

Metrics to evaluate the response strategies include:

1. Number aircraft crashes due to running out of fuel
2. Time to land aircrafts
3. Risk (hit probability) of landing the aircrafts
4. Cost of diverting the aircrafts

When a threat occurs, some subset of the aircrafts needs to be rerouted to land at alternate, safer destinations. Figure 2 illustrates a sample wide-area regional attacked scenario. The rectangular region over southern California is restricted due to attack threat. The airports in this rectangular region are depicted in the color red. The safer fortified airports are depicted in green, and the unfortified airports are depicted in blue.

The six maps in Figure 2 plot the flight path of a single aircraft to illustrate the various diversion strategies and potential risks and costs. From the top to bottom and then from left to right, the first map shows the original great circle flight path of the aircraft from Denver to Los Angeles. The second map shows the aircraft diverted to the nearest airport when the diversion directive was given. The third map shows the aircraft diverted to the nearest *fortified* airport when the diversion directive was given. The fourth map shows the aircraft diverted to the nearest airport from the original destination. The fifth map shows the aircraft diverted to the nearest *fortified* airport from the original destination. Finally, the aircraft avoid the airspace above the restricted Las Vegas airport to minimize risk.

Below each map, the metric measuring the economic cost and the time taken to diverted the aircraft. The economic cost is a rough estimate of the direct cost of diversion. The formula used for this economic metric is described in Section 6.3.2.

Figure 2 ArcView maps illustrating the various diversion strategies

The direct cost of diverting a single aircraft as shown in Figure 2 is relatively low. However, the direct cost of diverting can be very quickly escalate reaching into millions of dollars, if we are to divert planes away from away from a major metropolitan region, like Los Angeles. Diversion maybe considered a strategy of last resort, but it has to be part of comprehensive strategy. Diversion is needed when there is solid evidence of eminent attack, or if an attack has already taken place.

3 Technology Background

The SAAS project draws upon ISI's experience from two areas of expertise: building and analyzing large scale simulations, and software engineering with respect to integration and interoperability.

3.1 Simulation

The SAAS simulator combines a multi-aircraft flight simulator with a specialized optimizer built to reason with the kinds of models relevant to the aviation security scenario. USC/ISI's simulator group has extensive experience with constructing ultra-scale discrete simulators. On-going work on simulation in ISI's Joint Experimentation on Scalable Parallel Processors (JESPP) scaled the message communication infrastructure of the Joint Semi-Automated Forces (JSAF) simulator to run on Linux supercomputer clusters (Maui High Performance Computing Center and Wright-Patterson Aeronautical Systems Center). See Figure 3. This enabled the simulator to simulate entities with more complex behaviors (such as, experimental sensors and their platforms), as well as increase the sheer number of entities. JSAF was able to simulate more than one million entities, which broke the old record by a factor of ten. In the recently concluded Phase I Joint Urban Resolve experiments, the JESPP logger recorded close to two Terabytes of simulation data. The Phase I experiment focused on evaluating potential candidate sensor technologies to be developed by the year 2015 to help US forces operated in complex urban terrain. The next experiment will focus on how to prevent mortar fire attack in urban terrain using current military technologies.

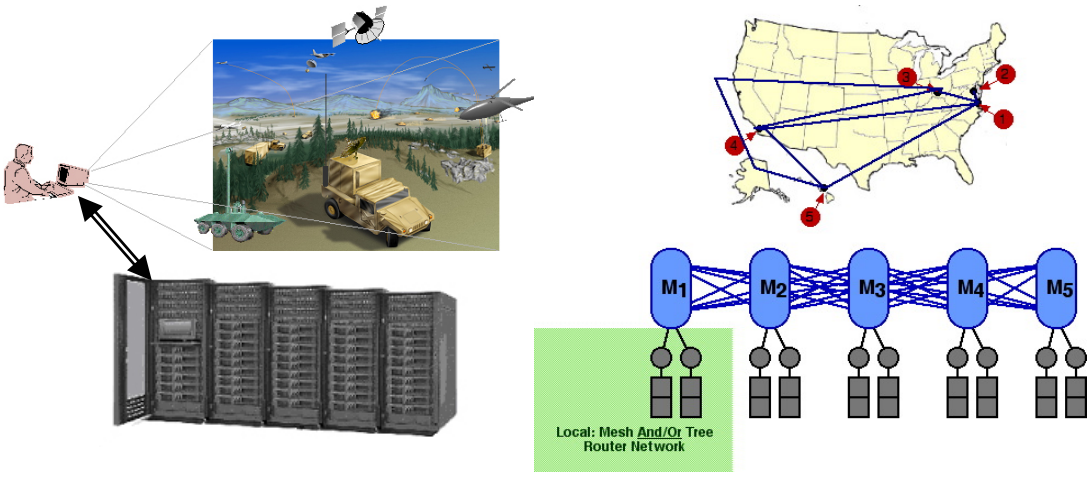

Figure 3 Distributed JSAF simulation runs on multiple Linux supercomputer clusters using JESPP's communication infrastructure.

For the SAAS application our experience enabled us to rapidly fabricate a custom simulator adapted to the requirements of the aviation security scenario. We include in our system a ported version of the JESSP logger. Usually, a logger is used for after-action review of simulation runs, but for SAAS, the logger maybe repeatedly referenced within the simulation cycle.

3.2 Software Engineering

SIM-TBASSCO (Semantic Interoperability Measures: Template-Based Assurance of Semantic interoperability in Software COmposition) addresses a critical problem in the longstanding software engineering goal of assembling software from components: adaptive composition sensitive to quality concerns. Conventional approaches support composition up to a point, but cannot handle qualitative considerations in composition, such as implementation effort, performance, resource requirements, or reliability. SIM-TBASSCO helps software developers engage in guided, efficient searches and evaluations of the set of alternative system implementations that can be built with the components available to them. It will let developers evaluate components' functional and data equivalence compatibility, find pertinent data conversion mappings, and predict performance (time, space, network) of a component architecture under specific usage situations and hardware/networking environments.

4 Design Requirements

The following are the key SAAS design requirements:

- Ability to simulate and match historical aviation data, such as the taking-off time/landing time data collected by the Bureau of Transportation Statistics (BTS).
- Ability to programmatically make dynamic changes to the simulation scenario to simulate intelligent planned responses to terrorist events.
- Modular design that facilitates the integration of computational risk/cost assessment models and optimizers as system components.

5 Existing State of the Art

5.1 Decision Support Systems

Decision support tools, such Analytica and @Risk, typically model at a relatively coarse-grained, aggregate level. A prototypical problem model by these tools is the *harvesting problem* (Bhargava, 1999). In this problem, a farmer has to decide either to harvest now or to harvest later. If he harvest now, then he immediately makes a profit. If he waits, then he can make a larger profit. But, if a frost occurs, then he makes less. This type of decision analysis can be modeled using influence diagrams and decision trees.

One problem with this type of modeling is how to assign the probabilities. What is probability of frost occurring? How will frost affect the quality and yield of the product? How is profit affected? At this level of modeling, these numbers are inputs to the model. Potentially more detailed models may be used to derive these input parameters. For example, we maybe able to use historical frost data to derive a statistical model of frost probability. Or, maybe we can use a weather simulator.

In the context of aviation, we may choose to model airport traffic delays using queuing theory. In queuing theory models, planes arrive at airports with certain probability distributions. Airports have finite landing capacities. So, using queuing theory we are able to calculate expected landing delays. However, if we model at a finer-grained level,

then we can turn this stochastic process of plane arrivals into a more deterministic process. By modeling individual plans at the entity level (take-off time, flight path and speed), we can determine with much greater confidence the plane arrival times.

Even with this more detailed model, we may not be able to determine the exact plane arrival time. There maybe delays due to weather or due to mechanical problems. We need to judiciously choose to prune away model details not needed to answer the decision problem at hand. However, by already modeling at the entity level we argue that we are able to model decision problems with less need to use potentially ad hoc input parameters.

The downside of modeling at the detailed entity level is the need for much greater computational power. If needed, we plan to overcome this by using massively parallel programming techniques.

5.2 Aviation Simulators

In general, a drawback to the use of an existing simulator is that each one provides much more functionality than SAAS requires along some dimensions, and not enough functionality along others. The commercial aircraft simulated in SAAS are not pushing physical limits, as, for example, a jet fighter would do. Instead, the aircraft fly at predetermined speeds, along straightforward routes. The current SAAS scenario bases its calculation of how long it takes an aircraft to land primarily on the estimated degree of congestion at its destination airport. If we were to simulate an aircraft trying to avoid a MANPAD missile, we would need a high-end physical flight simulator, but our current approach can substitute mathematical estimations of the likelihood of casualty during a landing.

On the flip side, we do need the ability to simulate the location of aircraft distributed across the continental United States to a reasonable degree of realism. If we were to use mathematical models that stochastically positioned aircraft across the same area, we would have no confidence that our results represent an accurate model of what would happen in real situations.

After surveying the available flight simulators, we chose to construct a new simulator for SAAS, rather than to base SAAS on a pre-existing one. For the SAAS system we found it necessary to write a custom aviation simulator because the parameters of greatest interest to us in terms of risk/cost assessment are not found in either military or other flight simulators, or in commercial simulators. The existing simulators tend to focus on the physics of flight, detailed interaction among planes (e.g., planes in aerial combat) and/or internal characteristics of particular airplanes. The SAAS simulator has crude approximations of the airplanes and their physical behavior, but it has more detailed modeling of airports (such as capacity, plane landing queuing and in the future fortification) and modeling of flight schedules to allow easy implementation of various response scenarios. In addition, the SAAS simulator provides us the flexibility to

augment the airport/plane models as needed to be compatible with risk/cost assessment models.

Alternative aviation simulators that we have examined include Microsoft Flight Simulator, ISI's STEAM, JSAF, and TAAM. The Microsoft Flight Simulator focuses on individual pilots with hyper-realistic cockpit environment and high precision flight physics. The SAAS is more interested in collective flight patterns and behaviors of multiple aircrafts. ISI's STEAM uses cooperating intelligent agents that work together as a coherent team. For example, STEAM is used to implement synthetic helicopter entities in coordinated attacks and transport scenarios. Such intricate cooperative behaviors are not necessary for SAAS scenarios. Moreover, they maybe difficult to scale to SAAS-like scenarios with thousands of entities covering national geographical extents.

The JSAF simulator is certainly capable of scaling up to very large number of entities. It provides a general purpose simulation framework incorporating many different types of entities, such tanks, airplanes, ships, sensors, buildings, infantrymen, and civilians. However, it is geared toward simulation of military entities, and out-of-the-box it does not have models needed for civil aviation simulation, such models of commercial airports (location, capacity) and airplanes. Because of its general purpose nature JSAF is very large and complex. It consists of several hundred library modules and takes a couple of hours to compile even on relatively fast CPUs. However, most of the machinery provided by JSAF is not needed by SAAS. SAAS does not need simulate realistic combat interactions with weapons fire, damage assessment, logistics, and command and control. JSAF machinery should be able to implement the BTS history matching capability, but it probably would overkill. But, more importantly the complex nature JSAF makes it difficult to integrate JSAF into the "simulation loop" with integrated risk/cost assessment models and optimizers.

The TAAM simulator is designed to simulate gate-to-gate air traffic. For example, it is capable of modeling detailed aircraft ground movement (gate allocation, pushback, taxiing, towing, runway allocation, and de-icing) and terminal maneuvering area (departure and arrival sequencing, standard instrument departure and arrival procedures, holding stack management, radar vectoring on approach, and airborne conflict detection). The TAAM does provide the airport capacity information needed for SAAS scenarios. However, such detailed simulation is intended for fine-tuning airport/airspace performance and resource utilization, but not needed by SAAS. Currently, our SAAS simulator simply characterizes airports capacity by the number of planes that can takeoff/land per hour. Using TAAM to simulate SAAS scenarios would generate unneeded simulation details, and waste computing resources. But, more importantly TAAM lacks the BTS history matching capability, which is part of the SAAS design requirement. Also, it lacks programmatic interfaces to dynamically change simulation scenarios, and to integrate risk/cost assessment models and optimizers.

6 SAAS System Architecture

To date, work on SAAS has created the infrastructure for a robust simulation tool, and implemented some basic simulation scenarios that have already yielded early information on the kinds of problems faced during threats to aviation security. SAAS includes

- A simulator that can track the path of large numbers (typically around 3000) of airborne aircraft during both routine and emergency flight. SAAS does not target detailed "last mile" simulations of aircraft behavior in close proximity to airports. It handles issues such as an airport's capacity for handling significant numbers of incoming aircraft, but assumes alternate simulators would be used if greater detail were desired.
- A database that includes knowledge of individual aircraft, airport locations and capacities, typical commercial flight schedules, passenger destination profiles, and other data essential to running a realistic simulation
- A logging facility that records all events that occur during a simulation. Modeled after USC's JESPP logger, the SAAS logger can be used to search or replay a simulation, enabling researchers to perform post hoc analyses of simulation runs.
- A graphic display facility that tracks aircraft moving on a 2.5D display of the continental United States. SAAS also can show a timeline view of a scenario.
- Metrics used to evaluate the risk and cost associated with particular response strategies and with particular simulator outcomes. Some of these preliminary metrics developed with the help of CREATE researcher. More sophisticated metrics maybe build and used, if need.
- An optimizer that searches for optimal strategies for rerouting of aircraft for a given threat scenario.

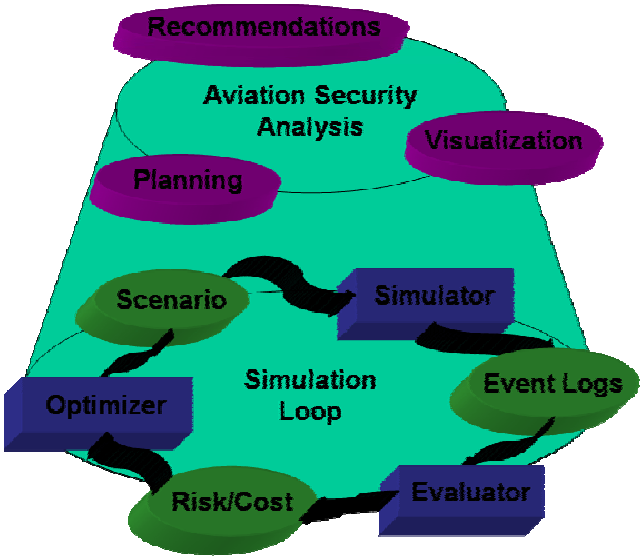

Figure 4 The SAAS Architecture. The "Simulation Loop" allows emergence planners to understand potential risk and cost of complex disaster scenarios by providing detailed entity-based simulations, evaluations of the simulation results with respect to risk/cost, and ways to tune the scenario improve response.

6.1 Scenarios and data sets

The simulation loop begins with the Scenario. In our context, we partition the scenario into three parts: neutral, red and blue.

The neutral part defines the background context in which the scenario operates. In our case, it is the flight schedules and air space flight patterns that would have taken place if no terrorist threat occurred. Bureau of Transportation Statistics (BTS) tracks the on-time performance of domestic flights operated by large air carriers. It provides statistics on the number of on-time, delayed, cancelled and diverted flights (BTS, 2005). Currently, we are able to digest BTS's raw Airline On-Time Performance database to initialize our simulation with actual historical flight information. The database records take-off and landing times using the local time zone. To make this time information useful to the simulator, we had to create a lookup table containing the major airports and their respective time zones.

Federal Aviation Regulation (FAR) requires minimum fuel requirements (FAA, 2005). For Visual Flight Rules (VFR) FAR Section 91.151 specifies that the aircraft must have enough fuel to fly to the first point of intended landing plus either an additional 30 minutes during the day, or an additional 45 minutes at night. For Instrumented Flight Rules (IFR) FAR Section 19.167 specifies that the aircraft must have enough fuel 1) to fly to the first point of intended landing, 2) fly from that point to an alternate airport and 3) fly another 30 minutes under normal cruising speeds. Under appropriate weather conditions and using standard instrumented, or specially approved, approaches, part 2 of the rule about flying to an alternate airport maybe waived. To simplify our scenario, we assume all aircrafts have enough fuel to fly 60 minutes after reaching their first intended point of landing.

The red part of the scenario definition specifies how the terrorist plan to attack. Elements of the attack plan that need to be specified include:

1. Types of attack
    - Single attack on an individual airport
    - Attacks at multiple airports
2. Types of weapon
    - MANPADS SA-7B with maximum effective range of 5500m
    - Barrett M82 HP Rifle with maximum effective range of 1800m
    - AK-47 Assault Rifle with maximum effective range of 300m
3. Types of launch site
    - Uniformly distributed around the airport
    - Normally distributed around the airport (clustered nearer the airport)
    - Normally distributed around the flight path

When specifying the types of attack, the airport(s) to be attacked is also specified. The types of weapon and launch site, influences risk metrics on landing the aircrafts. CREATE research Vicki Bier's sensitivity studies indicate the longer the range of the weapon the higher the hit probability. Also, the studies indicate that the normally

distributed launch sites (nearer airport or near flight path) have higher hit probability than uniformly distributed launched site around the airports.

The blue part of the scenario definition specifies the counter measures and response strategies to threats. Blue elements that need to be specified include:

1. The airports to be fortified and the perimeter of fortification
2. Other counter measures that reduces hit probabilities
3. Aircraft diversion strategies

By specifying the perimeter we can determine the probabilities of an aircraft being hit by an attack. Again from Vicki Bier's sensitivity studies the hit probabilities decreases as the radius of secured perimeter increases. This decrease is the most dramatic for uniformly distributed around the airport type of launch site. See Section 6.3.1 for more details. Also, we can potentially model other types of counter measures, such as on-board counter measures and pilot training. These counter measures all reduce the hit probabilities.

The aircraft diversion strategies are discussed in Section 2.

6.2 Simulation and simulation logging

Using the date specified in the scenario, the simulator reads in the historical BTS on-time database for that date, and defines a list of flight plans for all aircrafts. The simulator starts the simulation at midnight the previous day. As the simulation clock advances, it simulations the current position of the all the aircrafts currently in the air. If the scenario does not include any threat events, then simulation proceeds according the historical BTS data. Aircrafts take-off and land according to the historical time. The simulator approximates the position in between take-off and landing using interpolation along the great Earth circle. If emergence response events were injected into the simulation, then behavior of the aircraft entities change according to the response plan specified. Depending on the response plan, the optimization component may be invoked, see Section 6.4.

The landing capacities of airports are difficult to obtain. We used historical data to determine the maximum number of aircrafts that landed in any one hour period in the month of January 2004. Then, we added a fudge factor to that number and used that as airport landing capacity. The simulator uses this landing capacity to regulate approaching aircrafts forcing them to wait in line if needed. If more accurate data became available we could easily substitute in the correct airport capacity number.

With each clock tick of the simulation clock, each simulated entity emits an event indicating its current status. All these emitted event messages are intercepted and stored in event logs. The information in event logs is stored in relational database tables. Currently, we have chosen MySQL, a popular open source database, to implement the event logs. See the Appendix section for a list of current relational tables.

This relational interface over the event logs provides a good clean way to hide the choice of individual simulators and to hide how messages are intercepted and stored. As we have learned in our JESSP project, many analysts are comfortable working with relational data and can readily develop performance metrics based on the relational logs.

6.3 The Evaluator and metrics

The evaluator employs various *metrics* to evaluate the results of simulation runs and the decision made within the simulations. In turn these metrics access the relational event logs to compute their risk and cost assessments. Currently, we have implemented four metrics:

1. Number aircraft crashes due to running out of fuel
2. Time to land aircrafts after the divert signal event was given
3. Risk (hit probability) of landing the aircrafts at airports under threat
4. Cost of diverting the aircrafts

We provide more details on the implementation of metrics 3 and 4 below.

6.3.1 Risk of landing an aircraft

Our risk metric of landing an aircraft at an airport under attack is based on sensitivity studies perform by CREATE researchers Vicki Bier and her student Uche Okpara. Figure 5 shows two sample sensitivity graphs they generated. The graphs plot the expected hit probability with respect to the radius of secure region. The three curves within each graph represent different types to attack distribution: uniform around the airport, clustered near the center of the airport, and normal around the flight path. For example, if we were to form a 2000m secure region, then under Barrett rifle and uniform attack distribution conditions the expected hit probability on a single aircraft is approximately 0.18. If somehow, we were to land 5 aircrafts this airport, then the expected probability that none of the 5 aircrafts were hit is $0.82^5 = 0.37$.

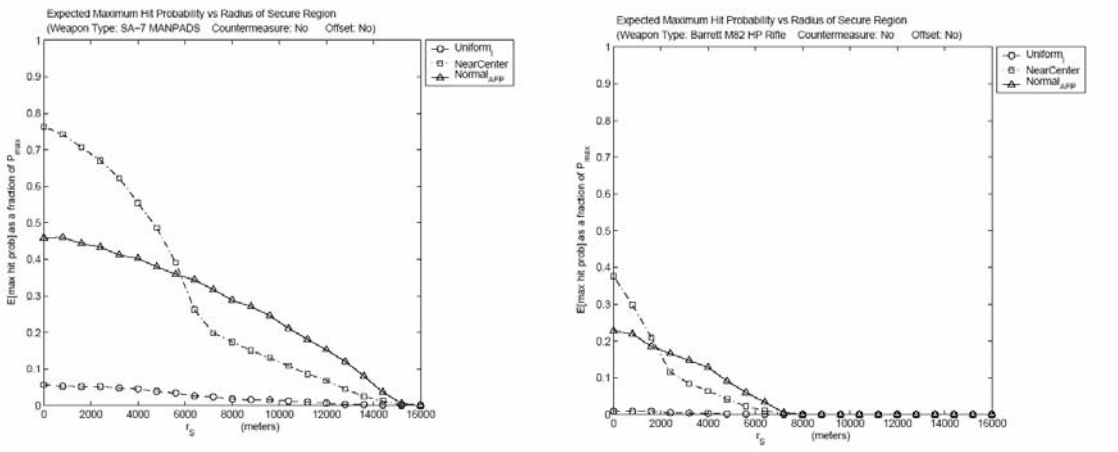

Figure 5 Sensitivity analysis for r_s (radius of secure region) generated by CREATE researchers Vicki Bier and Uche Okpara. The left graph plots the sensitivity of a MANPADS SA-7 weapon, and the right graphs plots the sensitivity of a large caliber Barrett M82HP rifle.

6.3.2 Calculation of cost required to divert an aircraft

The cost of diverting an aircraft may be divided into direct cost and indirect cost. Direct cost is the immediate cost incurred due to the flight diversion. The direct cost formula in Figure 6 provides a rough estimate of this cost. This formula is based on preliminary discussion with CREATE researcher with Vicki Bier and Lester Lave. The assumed parameter values used in the formula can easily be changed.

The indirect cost relates to the additional flight delays and rescheduling attributed to the diverted flight. Warren Qualley of American Airlines estimates the number of flight affected may range anywhere from 2 to 50 depending of the particular flight (Qaulley, 1997). Qualley states that sometimes indirect costs can eclipse the direct cost. However, currently we decided not include indirect cost into our diversion cost calculation. This is because of the large variability of number of affected flights and because we do not model the flight schedule of individual airlines. We may include indirect cost at a later time.

Diversion Cost ($) = Passenger Relocation Cost + Total Passenger Time Value
+ Crew Delay Cost + Aircraft Reposition Cost

Where,

Passenger Relocation Cost ($) = Cost of Each Alternate Means ($)
* Number of Alternate Means Required

Number of Alternate Means Required = Number of Passengers
/ Capacity of Each Alternate Means

Total Passenger Time Value ($) = Value of Each Passenger's Time ($/Hr)
* Number of Passengers
* Time Required to Relocate (Hrs)

Time Required To Relocate (Hrs) = Relocation Distance (miles) / Speed of Traveling
+ Boarding Delay (Hrs)

Assumptions:
Value of Each Passenger's Time = 20 ($/Hr)
Cost of Each Alternate Means = 500 ($/Day)
Capacity of Alternate Means = 50
Crew Delay Cost = 1000 ($)
Aircraft Reposition Cost = 2000 ($)
Speed of Traveling = 100 (miles/Hr)
Boarding Delays = 1 (Hr)

Figure 6 Estimation of the direct cost diverting an aircraft.

6.4 Optimization

SAAS combines a discrete simulator with an optimizer, so that they work in tandem. Most simulations introduce degrees of complexity both in terms of physical detail and programming complexity that precludes the incorporation of optimizers within the simulation loop. Instead, when humans are working with a simulator, they normally perform any optimizations manually, figuring out solutions on-the-fly, and then entering commands to influence the simulation based on the results of their mental calculations. The reason that we can support this combination in SAAS stems from the fact that the inherent complexity lies not in the physical parameters of the scenario, but in the risk and cost models referenced by the optimizer.

Currently, in we have two different optimizers with SAAS. The first is a simple greedy optimizer that we developed. Advantage of this optimizer is that is very fast and relatively easy to customize to explore different landing strategies. For example, we were able to easily add a nested optimization loop on top on the greed optimizer to explore how many fortified airports were needed in a national threat scenario where all aircrafts have to land fortified airports.

Also, SAAS is able to invoke in an off-line fashion the COmputational INfrastructure for Operations Research (COIN-OR) SYMPHONY solver (COIN-OR, 2005). COIN-OR is an open source effort to provide the operation research community with cutting-edge research optimization software. The next section describes our efforts in mapping the SAAS landing problem to COIN-OR solvers. The SYMPHONY is a mixed integer linear programming solver within COIN-OR.

6.4.1 COIN-OR Solver

A faithful mapping of the aircraft landing problem to mathematical programming results in high time complexity problems. Previous work in aircraft landing that used mixed integer linear programming modeled 3 landing strips with about 50 aircrafts (Beasley et al, 2000). In our case, with a national threat scenario we are potentially dealing with thousands of aircrafts with hundreds of airports.

In our mapping, we simplified the problem in order to scale up the size of the problem that can be solved. One key simplification is that instead of enforcing the minimum time interval between to two successive aircraft landings, we divide up time into fixed length time slots. The length of the time slot may differ from airport to airport, based on the landing capacity of the airport. This simplification turns the aircraft landing problem into an assignment problem.

Initially, we setup the constraint that only one aircraft may land at one time slot. With this one-to-one mapping, the solution we obtain is a strict upper bound of the problem formulation using minimum time intervals between aircrafts. Latter to reduce the problem size further, we lengthen the time slots and increased the number of aircraft that can land per time slot. Without this one-to-one mapping, the solution is no longer a strict upper bound, but the time to solve problems is dramatically reduced.

One sample instance that we ran of the national threat scenario diverted 3383 aircrafts to land at 254 airports. The one aircraft to one airportSlot formulation took ~1.6GB to encode using the standard MPS encoding. This problem size was too big to fit into main memory and to run efficiently. Merging four slots into one and setting M to 4, the problem encoding size reduced to ~400MB. It took COIN-OR SYMPHONY ~2 hours to solve this problem on 3GHz Pentium 4. Using a variable merging scheme, where we tried to get slot sizes of ~10 minutes, we reduce the problem encoding size down to ~350 MB. This problem took ~20 minutes to solve.

Comparing with our simple greedy algorithm of diverting aircrafts to the airport nearest their current location, the COIN-OR solver is able to find diversion plans that are ~10 minutes faster on average per plane. This indicates COIN-OR was able to find alternate less congested airports to land the aircrafts. With 3383 planes to land, saving ~10 minutes per plane equates to ~560 hours of fuel savings.

Below is our formulation of the aircraft landing problem:

d_{ij} = 1, if $plane_i$ lands $airportSlot_j$
c_{ij} = cost of landing $plane_i$ at $airportSlot_j$

Objective function:

$$\min \sum_i \sum_j c_{ij} d_{ij}$$

Constraints

$\forall i \sum_j d_{ij} = 1$, a *plane* must land at one and only one *airportSlot*

$\forall j \sum_i d_{ij} \le M$, limit the number of *planes* per *aiportSlot*

The matrix $C = (c_{ij})$ can be manipulated to represent different cost concerns. Currently, we use the matrix C to represent the time it takes for $plane_i$ to land at $airportSlot_j$. But, potentially the cost may represent a combination of different concerns, including

- Time to land $plane_i$ at $airportSlot_j$
- Cost to divert $plane_i$ to $airportSlot_j$
- Risk of landing $plane_i$ at $airportSlot_j$

In addition, we can potentially add more constraints to include other aspect of the problem. For example, by constraining appropriate d_{ij} variables to 0, we exclude the optimizer from considering airportSlots that are too far away or that are too far into the future. This way we can account for limited fuel capacity of the aircrafts.

Some airport fortification strategies are dynamic and may take time to setup. For example, mobilizing the police to setup a temporary perimeter will take time. In these scenarios, we can constrain appropriate d_{ij} variables to 0 to force the optimizer to consider other airports or to delay the aircraft arrival.

7 Visualization

The SAAS architecture provides a Visualization component to allow users to enable users to interact with the simulator and to explore the results of the simulations. In SAAS simulation logs are stored in relation tables, which provide a convenient API for access, but it is often difficult for average users to grasp the information stored within those tables. Visualization provides alternative means for viewing the state (and projected states) of a simulation.

We have incorporated two visualization components into SAAS. One is ISI' CHIME, a Semantic Web-based tool that can directly access the SAAS relational simulation log (MacGregor, 2003). Using semantic mappings, it correctly interprets time and location attributes to automatically generate visualizations. Figure 1 is a CHIME map (based on OpenMap) that show the location of aircrafts at a point in time. Figure 7 below shows an alternative means for viewing, plotting aircraft on a timeline that shows the intervals during which each aircraft is airborne. These two views show the projected airborne intervals before and after the occurrence of a threat. In the after view, the intervals are shortened because the aircraft have been instructed to land more quickly at alternative destinations.

We have also used the industry standard ArcView GIS to display map-based information, see Figure 2.

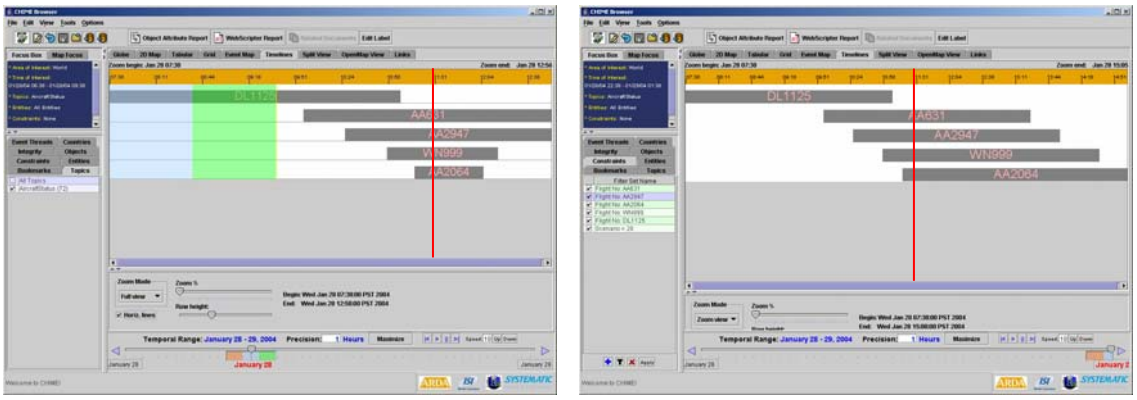

Figure 7 Flight times showing normal flight patterns and diverted flight patterns.

8 Experimental Results

We are currently in the process of exercising the SAAS system to run through various threat scenarios. The threat scenario we plan to investigate ranges from local threats that shut down individual airports to national threats that may force the diversion of all aircrafts. The metrics that we plan to use to measure the various response strategies include time to land the diverted aircrafts, the risk to land the aircrafts and the cost related to diversion.

Below we show initial results from a threat scenario were LAX is under attack. Aircrafts heading to ward LAX are ordered to divert. In addition, because the responsible terrorists have not been apprehended, aircrafts heading toward neighboring airports of Long Beach, Burbank, Santa Ana and Ontario are also ordered to divert. The weapons employed by the terrorist are readily transportable. The terrorist may easily relocate by car and attack the neighboring airports. In addition, some of the neighboring airports' landing flight paths intersect with LAX. The terrorist may be able to attack aircrafts landing at these neighboring airports using the same attack location.

Figure 8 shows an experimental result curve plot that trades off diversion cost against time to land the aircrafts. The three points on the trade-off curve represents three different response strategies 1) land as soon as possible by diverting to the airport nearest to their current location, 2) land at a safer fortified airport nearest to their original intended destination airport, and 3) land at any airport nearest to their original intended destination. The details of the three strategies are shown in Figure 9, Figure 10 and Figure 11.

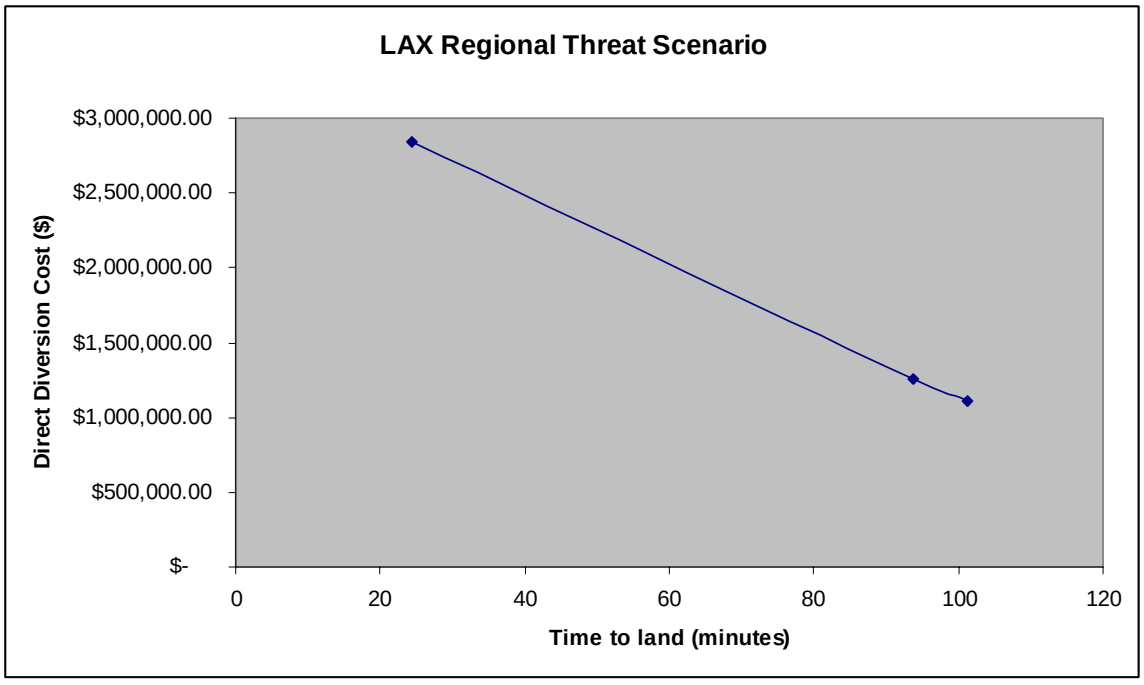

Figure 8 Tradeoff curve between diversion cost and time to land diverted aircraft.

SAAS Results when planes are diverted to fortified airport nearest to their current location

Airport under attack: LAX

Airports shut down:

1. Los Angeles, Los Angeles International Airport (LAX)
2. Long Beach, Long Beach Airport (LGB)
3. Burbank, Burbank-Glendale-Pasadena Airport (BUR)
4. Santa Ana, John Wayne Airport-Orange County Airport (SNA)
5. Ontario, Ontario International Airport (ONT)

Total Flights Diverted: 158
Total Flights Landed: 158
Total Flights Crashed: 0
Percentage Flights Landed: 100.0
Percentage Flights Crashed: 0.0

---------- TIME STATISTICS ----------

- Average time taken by a diverted aircraft to land after Ground command: 24.42 min(s).
- Standard Deviation in time taken by a diverted aircraft to land after Ground command: 25.32 min(s).

---------- DISTANCE STATISTICS ----------

- Average distance traveled by a diverted aircraft before landing and after the Ground command: 125.69 mile(s).
- Standard Deviation in distance traveled by a diverted aircraft before landing and after the Ground command: 191.32 mile(s).

---------- COST STATISTICS ----------

- Average Cost to divert one plane: $17,966
- Total Cost to divert the planes: $2,838,645

Figure 9 LAX regional threat scenario: land as soon as possible.

SAAS Results when planes are diverted to fortified airport nearest to original destination

Airport under attack: LAX

Airports shut down:

1. Los Angeles, Los Angeles International Airport (LAX)
2. Long Beach, Long Beach Airport (LGB)
3. Burbank, Burbank-Glendale-Pasadena Airport (BUR)
4. Santa Ana, John Wayne Airport-Orange County Airport (SNA)
5. Ontario, Ontario International Airport (ONT)

Total Flights Diverted: 158
Total Flights Landed: 158
Total Flights Crashed: 0
Percentage Flights Landed: 100.0
Percentage Flights Crashed: 0.0

---------- TIME STATISTICS ----------

- Average time taken by a diverted aircraft to land after Ground command: 93.68 min(s).
- Standard Deviation in time taken by a diverted aircraft to land after Ground command: 81.2 min(s).

---------- DISTANCE STATISTICS ----------

- Average distance traveled by a diverted aircraft before landing and after the Ground command: 568.4 mile(s).
- Standard Deviation in distance traveled by a diverted aircraft before landing and after the Ground command: 576.13 mile(s).

---------- COST STATISTICS ----------

- Average Cost to divert one plane: $7,981
- Total Cost to divert the planes: $1,261,026

Figure 10 LAX regional threat scenario: land at fortified airport nearest original intended destination.

SAAS Results when planes are diverted to airport (fortified or not) nearest to original destination

Airport under attack: LAX

Airports shut down:

1. Los Angeles, Los Angeles International Airport (LAX)
2. Long Beach, Long Beach Airport (LGB)
3. Burbank, Burbank-Glendale-Pasadena Airport (BUR)
4. Santa Ana, John Wayne Airport-Orange County Airport (SNA)
5. Ontario, Ontario International Airport (ONT)

Total Flights Diverted: 158
Total Flights Landed: 158
Total Flights Crashed: 0
Percentage Flights Landed: 100.0
Percentage Flights Crashed: 0.0

---------- TIME STATISTICS ----------

- Average time taken by a diverted aircraft to land after Ground command: 101.17 min(s).
- Standard Deviation in time taken by a diverted aircraft to land after Ground command: 86.48 min(s).

---------- DISTANCE STATISTICS ----------

- Average distance traveled by a diverted aircraft before landing and after the Ground command: 619.62 mile(s).
- Standard Deviation in distance traveled by a diverted aircraft before landing and after the Ground command: 607.46 mile(s).

---------- COST STATISTICS ----------

- Average Cost to divert one plane: $7,037
- Total Cost to divert the planes: $1,111,982

Figure 11 LAX regional threat scenario: land at the closest airport nearest to the original intended destination.

9 Summary

The SAAS project is developing a simulation-based decision support infrastructure and its corresponding tools to help emergency personnel to rapidly respond to terrorism events based on risk/cost models.

The SAAS simulation helps emergency responders understand complex and dynamically changing situations. It helps the responders discover interesting, and possibly unforeseen, behaviors that may result from actions of the terrorists and from the actions of the responders themselves. With integrated computational risk/cost assessment models it helps the responders evaluate the consequences of the behaviors. The SAAS simulator infrastructure makes paper risk models "come alive" (and vice-versa). It provides a repeatable platform to allow modelers fine tune parameters and to allow responders to adjust their response strategies accordingly.

The Simulator enables learning/training in advance of actual crises. The U.S. military uses a similar methodology. They use simulations to practice developing detailed emergency plans, called Time-phased Force Deployment Data (TPFDD), in response to Noncombatant Evacuation Operation (NEO) scenarios. These practices aid the human responders to assimilate the doctrine so that they react properly in time of an emergency. Moreover, these repeated practices may help the responders to discover new and improved doctrines (the right response to a given situation).

10 Reference

Beasley, J. E., M. Krishnamoorthy, Y. M. Sharaiha, D. Abramson. "Scheduling Aircraft Landings--- The Static Case." Transportation Science, Vol 34, No 2, May 2000.

Bhargava, Hemant K., Suresh Srihar and Craig Herrick. "Beyond Spreadsheets: Tools for Building Decision Support Systems." IEEE Computer, volume 23, issue 3, March 1999, pp 31-39.

Bureau of Transportation Statistics. "Airline On-Time Statistics and Delay Causes." June 1, 2005. < http://www.transtats.bts.gov/OT_Delay/OT_DelayCause1.asp>

Computational Infrastructure for Operations Research. June 1, 2005. < http://www.coin-or.org/>

Federal Aviation Adminstration. "Federal Aviation Regulation." June 1, 2005. http://www.airweb.faa.gov/Regulatory_and_Guidance_Library/rgFAR.nsf/MainFrame?OpenFrameSet

MacGregor, Bob, In-Young Ko. "Representing Contextualized Data Using Semantic Web Tools." In Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems, ISWC 2003. Sanibal Island, Florida, USA, October 2003.

Qualley, Warren L. “Impact of Weather on and use of Weather Information by Commercial Airline Operations.” Workshop on the Social and Economic Impact of Weather. Boulder, Colorado, USA, April 2-4 1997. http://sciencepolicy.colorado.edu/socasp/weather1/qualley.html

11 Appendix

Description of SAAS database

Tables:

1. Scenario – For each run of a single SAAS simulation, an entry is generated in this table that briefly describes the simulation. The table contains following fields
    1. Name – Name of the Scenario.
    2. Simulation Date – Date for which the simulation is executed.
    3. Time of Entry.
    4. Date of Entry.
    5. Ground Time – Specifies the time at which flights need to be diverted in case of emergency.

2. Airport – List of airports along with their attributes
    1. Iata Code – A unique 3 character code associated with an airport (example: ‘LAX’ for Los Angeles International Airport).
    2. Airport Name.
    3. City Name – City in which the airport is located.
    4. State.
    5. Lat – Latitude at which the airport is located.
    6. Lon – Longitude at which the airport is located.
    7. Alt – Altitude at which the airport is located.
    8. Time Zone – The time zone in which the airport is located.
    9. Busiest Flights per Hour – The maximum number of planes that can land on the airport in an hour.

3. Flight Plan – For each flight in the simulation an entry that describes the flight’s journey is entered in this table with the following attributes.
    1. Scenario – Id of the scenario with which the flight is associated.
    2. Carrier – A 2 character code of the flight carrier.
    3. Flight Number.
    4. Origin – Id of airport at which the flight took off.
    5. Take Off Time.
    6. Dest – Id of airport that the flight is destined to.
    7. Landing Time.
    8. Number of Passengers.
    9. Diverted Dest – Id of airport to which the flight is diverted in case of emergency.

10. New Landing Time – New landing time of the flight, if it is diverted to a new destination in case of emergency.
11. Dist From Original Dest – Distance of new destination airport from the original destination airport.
12. Diverted at Lat, Diverted at Lon – Location on the flight's path where the flight is given an emergency ground command.
13. Cost of Diversion – Amount in dollars required to divert the plane during emergency.

4. Aircraft Status – Each flight in the simulation, at regular interval of time, generates an entry that indicates its current location during the execution of the simulation. Each entry has the following field.
    1. Simulation Time – Simulation Time at which this entry was generated.
    2. Scenario – Id of the scenario with which the flight is associated.
    3. Flight Plan – Flight Plan Id of the aircraft for which this entry is generated.
    4. Lat, Lon – Location of the aircraft.
    5. Alt – Altitude of the aircraft.
    6. Azimuth – Azimuth of the aircraft that indicates the direction in which the plane is heading.
    7. ETA – Estimated time of Arrival of this aircraft to the airport it is destined to.
    8. Time To Fuel Empty – Specifies the time after which the aircraft would run out of fuel. In other words this value indicates the amount of time the plane can fly from this point on.

5. Flight Summary – For each flight in the simulation a brief report of its journey is entered in this table which contains the following fields.
    1. Scenario – Id of the scenario with which the flight is associated.
    2. Flight Plan – Flight Plan Id of the aircraft for which this report is generated.
    3. Simulation Time – Simulation Time at which this report was generated.
    4. Lat, Lon – Location of the flight when this report was generated.
    5. Flight Report – Entry that indicated whether the flight 'Landed' successfully or 'Crashed' due to lack of fuel.

6. Fortified Airport – A list of airports that are fortified with the following attributes.
    1. Airport Name – The unique 3 character code associated with an airport.
    2. Time to Fortify – Time required to fortify the airport (in minutes).
    3. Cost to Fortify – Economic cost required to fortify the airport. This value is on a scale of 1 to 10.
    4. Landing Capacity Per Hour – The maximum number of planes that can land on the airport in an hour.
    5. Capacity After Fortifying – The maximum number of planes that can land on the airport in an hour during emergency.
    6. Probability of Fortification – The degree of fortification of an airport. This value is on a scale of 0 to 1.

7. Unsafe Airports – The List of Airports that are under attack or that may be unsafe for an aircraft to land on.

1. Airport – The unique 3 character code associated with an airport.

8. Batch Ground Times – If the simulation is to be executed for multiple Simulations Dates rather than a single day, this can be achieved by specifying a list of emergency ground dates and times (one for each day). This table contains following fields.
    1. Ground Date – Simulation Date for which the simulation is supposed to be executed.
    2. Ground Time – Simulation Time at which emergency ground signal is to be generated.