Proceedings of the 2010 Winter Simulation Conference
B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.

SIMULATION-BASED MANPOWER PLANNING WITH OPTIMIZED SCHEDULING IN A DISTRIBUTED MULTI-USER ENVIRONMENT

David Kalasky

IBM Corporation
2300 Dulles Station Blvd.
Herndon, VA 20171, USA

Michael Coffman

Transportation Security Administration
601 South 12th Street
Arlington, VA 22202, USA

Melanie De Grano

IBM Corporation
2300 Dulles Station Blvd.
Herndon, VA 20171, USA

Kevin Field

ProModel Corporation
556 East Technology Ave.
Orem, UT 84097, USA

ABSTRACT

The Transportation Security Administration (TSA) staffs and operates over 450 airports in the US. TSA has been using simulation to determine staffing requirements since 2005 and has recently completed a refresh of their manpower planning and scheduling system. The objectives of the effort were to replace the GPSS simulation engine, optimizer and user-interface (UI) to take advantage of current network-based systems technologies. The previous system was distributed to the 200+ users as a stand alone application. This presented maintenance, security and performance issues, especially during the annual budgeting process. This paper focuses on the creation and integration of the simulation engine which was required to replicate and improve on the existing GPSS model accuracy and performance. Additional considerations included providing TSA an easy-to-use simulation platform to maintain the simulation engine, make model and data edits and expand the use of simulation technology within TSA.

1 INTRODUCTION

1.1 Existing TSA System, User Environment and Motivation for Change

TSA is responsible for ensuring the security of commercial air travel in the United States with passenger and baggage screening operations at over 450 airports. Each of these airports presents unique screening equipment, layout configurations, and passenger travel characteristics. In order to make sure TSA resources are used in the most effective manner, a simulation model was developed shortly after the creation of TSA to model each airport's unique screening operations. This model proved to be an effective staffing tool and has gained the support of the airport administrations, airlines and budget oversight officials.

The initial simulation model was developed using the GPSS simulation language. GPSS is an older simulation language and TSA found that it was becoming increasingly difficult to work with compared to some of the commercial off-the-shelf (COTS) simulation packages. As TSA reached the end of its contract support of the existing tool, a decision was made to create the next generation of the staffing model

from scratch in a COTS simulation package rather than continue to develop the existing tool. In the next generation staffing model, TSA wanted to own all of the software source code to facilitate future software development.

TSA currently utilizes two separate applications to determine the staffing levels needed at each airport. The first application is the GPSS simulation model used to determine staffing requirements by job function in five-minute intervals. The second application is a schedule optimization software application which utilizes TSA's staffing guidelines to determine the optimal staffing schedule to cover the passenger and bag screening work. One of the goals of the next generation staffing model is to integrate these two software applications into one system which would provide the functionality of both. This integration should provide advantages in application usability, maintaining IT security, documentation and Help Desk support. The new application provides the following improvements over the initial tool:

- a web-based application utilizing an off-the-shelf software modeling package
- the ability to transition from a process of running one replication of a simulation model to the ability to run multiple replications and look at trends over the multiple replications
- integration of the functionality provided by two separate applications into one tool

By transitioning to a web-based application, TSA felt it would be easier to promote new versions of the software application to production and distribute the updates to the user base with little to no user involvement. The current system operates as a standalone application which requires updates to the software to be manually downloaded and installed by the user. Additionally, each month, new flight data becomes available through a subscription with OAG and a very large file that must be downloaded by the users in order to provide the most up-to-date data for analysis. A web-based application allows TSA to load new flight data on a centralized server and provides users the most current version of the application and data. The diagram in Figure 1 provides an overview of the proposed system's architecture.

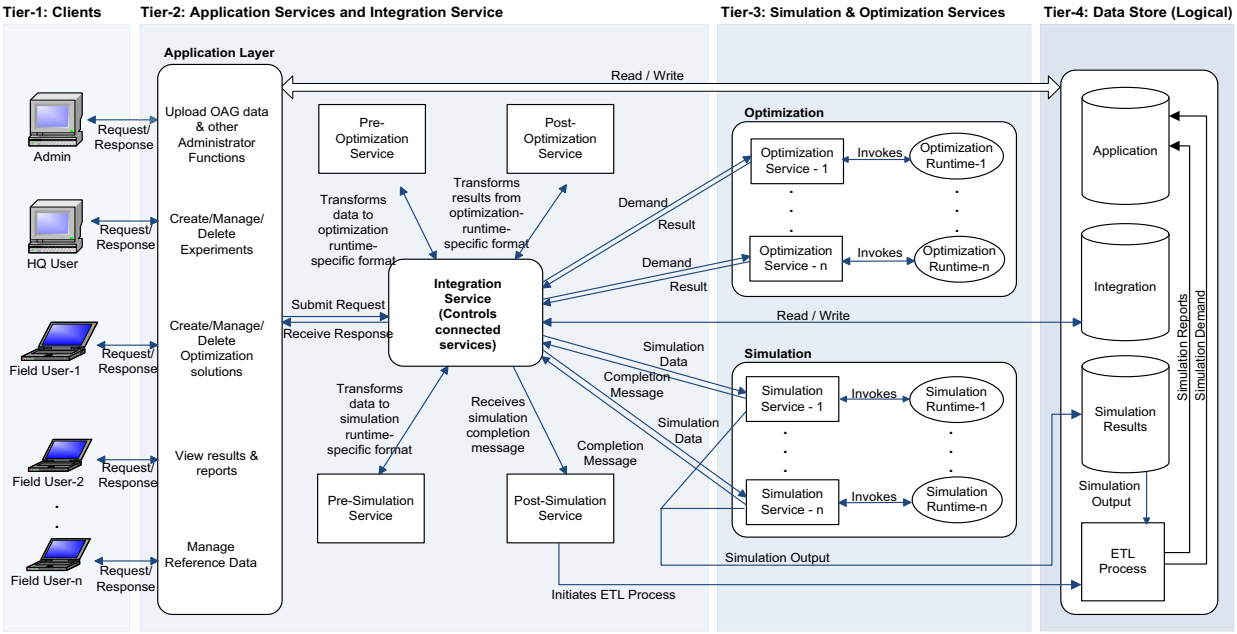


Figure 1. Proposed system architecture

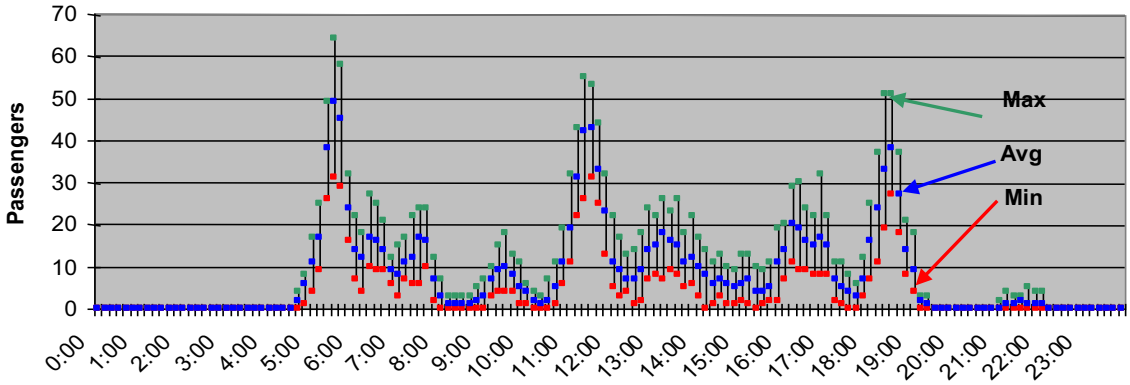
The architecture is structured into four tiers: Tier 1 comprises clients who are connecting to the application using the web browser, including headquarters (HQ) users and field users, Tier 2 contains the web application, integration service, and pre- and post- processes for simulation and optimization, Tier 3 consists of the simulation and optimization run-time instances as well as processes that manage the inte-

raction with the run-time instances, and Tier 4 has the application database with logical partitions as well as database processes such as Extract, Transfer, and Load (ETL). Designing the model architecture in this way allows longer-running services such as simulation and optimization computations to be executed in the background. In addition, pre- and post- execution transformations of the simulation and optimization model can be performed in a different tier than model execution. These design considerations minimize the overall system run time and provide adequate capacity for users engaged in all stages of planning.

1.2 Current Model Variability

The simulation model provides the required staffing levels needed every five minutes for both passenger checkpoints and bag zones. Once the staffing requirements are generated, they are used as input into the staff plan optimizer to determine optimum manpower plans with full-time equivalents (FTEs) and part-time (PT) personnel based on local scheduling constraints. Experimentation has shown that the variation in the simulation results at the five-minute interval level is dampened through the staff plan optimizer. Optimized schedules from different simulation model runs resulted in very similar calculations of total FTEs.

The following passenger arrival pattern in Figure 2 was obtained from 10 replications of an existing sample simulation model for a one-week period, using different seed values. While the total number of passengers is the same across simulation replications, altering the seed value affects the arrival pattern at the 5-minute level, which in turn affects the required staffing levels. Although the difference between the minimum and maximum passenger arrivals in this example varied up to ±20%, the variance was reduced to less than 2% by the staff plan optimizer.



Varying passenger arrivals generate different staffing requirements, which are converted to schedules through the staff plan optimizer.

Seed	Eff	FTE	FT	PT
100	60.83%	45.5	36	20
200	63.35%	43.7	33	19
300	61.25%	44.8	35	19
400	60.08%	45.9	36	20
500	62.50%	44.5	34	19
600	58.99%	46.6	38	20
700	61.45%	45.1	35	19
800	61.49%	44.9	35	19
900	61.01%	45.7	36	20
1000	61.22%	45.3	35	19

Staff plan optimizer results show a very small variation in the number of FTE's required across the 10 simulation runs.

Category	Average	Min	Max	stdev
Eff	61.22%	58.99%	63.35%	0.01
FTE	45.2	43.7	46.6	0.80
FT	35.3	33	38	1.34
PT	19.4	19	20	0.52

Figure 2. Variance reduction by use of the staff plan optimizer

1.3 Current Model and System Issues and Drivers for Platform Selection

The current manpower planning system had evolved over the past five years and included a GPSS simulation model and a CPLEX-based scheduling optimizer. The GPSS simulation engine is compiled code within a PC-based UI and reporting system. The combination of not owning the simulation code and support requirements for several hundred stand-alone PC versions of the tool were strong drivers for a system overhaul. The scheduling optimizer was network-based and coupled to the simulation model via a rough-cut capacity requirements plan. The plan was output as a CSV file stating the manpower requirements in five-minute intervals for a simulated work week. Running the model for one week was appropriate given the repeat nature of the weekly flight schedules and the runtimes experienced in running both the staff model (simulation for rough-cut manpower requirements) and staff plan (optimized, constraint-based scheduling).

One of the main goals of the next generation staffing model was to provide an increase in model execution speed and the ability to execute multiple iterations at the same time. Though many of the available simulation packages could successfully handle the modeling needs of the new application, ProModel was selected as it allows multiple simulations to be run on the same machine in a networked environment.

2 MODELING APPROACH –DEVELOPMENT AND VALIDATION

2.1 Prototype Development

The functional and technical requirements for the simulation were derived from the latest version of the current system. While most of the functional requirements were documented, several of the technical requirements were either not implemented or not easily discerned due to not having the GPSS model source code. This required a thorough analysis of the current system to understand demonstrable inputs, outputs, and functionality. The agreed upon approach was to replicate the current system functionality and results, while providing improvements in ease-of-use, run-time and the user interface for experimentation and results generation. The prototype model was structured in three functional areas: concourse, checkpoint and bag zone. These are the core processes in every airport and provided a structure for modeling that could simulate any airport configuration. The data and modeling for each functional area are now discussed.

2.2 Concourse Modeling

The concourse provides a location for passengers to arrive at the airport and check baggage. One of the key data sets and a major source of variation in the model are passenger arrivals at the airport and subsequently to the checkpoints for TSA screening. The current model used a fairly intense data development schema to model passenger arrivals at the airport. Since the model focus was on passenger and baggage screening, we were only interested in originating passengers and their luggage. The passenger arrival process begins with the OAG (Official Airline Guide) for flight departures and aircraft capacity. This information is coupled with data for flight load and origination factors. The following equation is used to calculate the expected number of passengers requiring checkpoint screening for each departing flight:

$$\textit{expected number of passengers} = \textit{aircraft capacity} * \textit{origination factor} * \textit{load factor}$$

While this provides a good estimator of the expected number of passengers, TSA maintains empirical data and resulting statistical distributions for modeling arrivals and screening processes of passengers and luggage. The first empirical distributions used in the model were to model the arrival patterns of passengers prior to their flight's estimated departure time (EDT). These empirical distributions were a function of time of day, peak vs. off-peak and domestic vs. international. The distributions were applied to individual flights and modeled as random draws for each passenger beginning at the earliest possible arrival

time. In the arrival data example shown in Figure 3, the earliest passengers arrive 170-180 minutes before the flight departure, and the latest passengers show up 50-60 minutes before the flight departure.

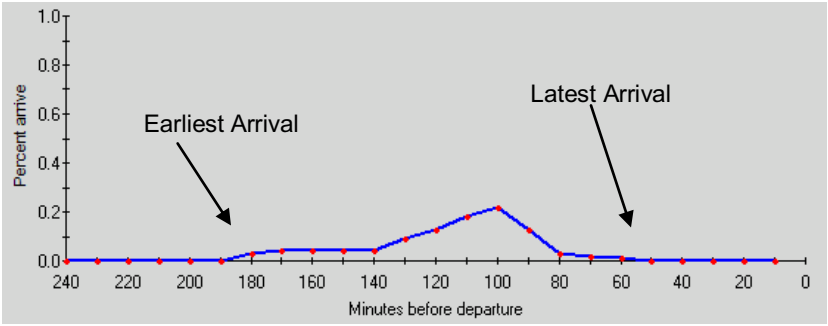


Figure 3. Example arrival distribution for peak domestic flight. Screenshot taken from Regal Decision Systems, Inc. (2009).

The simulation model implements this variability in passenger arrivals by generating all of the expected passengers for a given flight at the earliest passenger arrival time into the concourse location. Each passenger is then delayed according to a random draw from the appropriate arrival distribution. Thus, the arrival distributions are treated in the model as time delays with the earliest passengers having zero delay time. The peak domestic distribution is converted to a delay distribution in ProModel, as shown in Table 1.

Table 1. Peak domestic arrival pattern converted to a delay distribution

Percentage	Value
0	0
3	10
4	20
4	30
4	40
4	50
9	60
13	70
18	80
22	90
13	100
3	110
2	120
1	130

The first nonzero entry in the table means that 3% of the passengers are delayed between 0 and 10 minutes once they have been generated (the distribution is continuous). Once in the system, each of the passengers would wait according to a random draw from this peak domestic distribution before processing in the concourse. While TSA maintains several standard passenger arrival data sets, each user has the ability to model their unique arrival patterns for situations such as connecting, military and charter flights.

The next step in the process is to account for check-in and baggage check. The number of checked bags is modeled as a discrete distribution ranging from zero to six bags. The distribution is unique to each flight and like passenger arrival patterns is a function of the flight duration and destination (international vs. domestic). Likewise, the check-in factor is unique for each flight and results in a delay in the concourse location. If the generated passengers have no bags and are not checking in, then they are routed directly to a checkpoint for passenger screening. Bag zones can be located at curbside, lobby or in a bag room. Once baggage and check-in delays are accounted for, passengers are routed to an available checkpoint. When multiple checkpoints are available for a given flight, passengers are routed based on most capacity available.

2.3 Checkpoint Modeling

The goal of the simulation model is to determine staffing requirements for every five minute interval. At the checkpoints, the staffing is based on the number of lanes open. Thus the model logic for opening and closing checkpoint lanes had to be developed before staffing levels could be determined. The checkpoints operate in a single queue, multiple server mode.

Checkpoints are broken up into at most five different lane types, called priority groups, which were created to model lanes that have different processing rates for passenger screening. A snapshot of the checkpoint queue is taken every five minutes to determine how many checkpoint lanes are needed to process the passengers in queue within the measure of effectiveness (MOE), or maximum wait time goal. The model performs an iterative calculation within a subroutine to determine the appropriate number of lanes to open. The time to clear the passengers out of the queue is determined by:

$$\textit{time to clear queue} = \textit{number in queue}/\textit{processing rate}$$

The number of lanes needed to guarantee passengers are processed within the MOE is calculated by:

$$\textit{number of lanes} = \textit{time to clear queue}/\textit{MOE}$$

The result is rounded up to the nearest integer. Once the total number of lanes is found, it is broken up into the number of lanes per priority group. If the MOE cannot be achieved with the number of priority 1 lanes available, the model will determine how many more lanes are needed at the next priority group rate, up to priority group 5. The different rates of each priority group are taken into account when determining the total number of lanes needed for that interval. If there are no passengers waiting in line, the checkpoint lanes are closed for the next five minutes. This is done so that airports that have long intervals between flight departures can make best use of their staff. An additional demand for personnel screening at the checkpoints is created in the form of non-passengers which are generated in two ways: as a percentage of passenger volume and fixed or specified volume levels.

2.4 Bag Zone Modeling

Bag screening employs both automated Explosion Detection Systems (EDS) and manual Explosive Trace Detection (ETD) using a wide variety of equipment and configurations across all airports. Bags are screened at rates with variation depending on the bag zone configuration. Where EDS machines are used for primary screening, a secondary screening queue is available for alarmed bags which do not pass primary screening. Secondary processing is accomplished by re-screening, on-screen resolution or through manual ETD searches. The opening and closing of bag screening equipment is handled in the model in the same fashion as checkpoints to open and close equipment to maintain a MOE threshold.

2.5 Prototype Validation

Sample airports were used to develop and verify the initial Enhanced Staffing Model (ESM) prototypes. The prototype models were built as stand-alone applications so that it could easily be compared to the current Staffing Allocation Model (SAM) system. While the prototypes did not have all the functionality of the final model, they include the following:

- read actual flight schedule information
- create individual passenger arrivals according to user-defined distributions prior to their estimated departure time
- create individual non-passenger arrivals according to a defined percentage of passengers or based on hourly arrivals
- model check-in and associated delays to account for passenger ticketing
- create passenger baggage according to user-defined distributions
- model checkpoint lane and bag screening equipment opening and closing

- model and reports queue activities for checkpoints and bag screening equipment
- utilize schedules for checkpoint and bag zones open and closed hours

The goal was to match current system statistics within +/- 5%. Table 2 shows the selected statistics used for validating the model. Following the model design of generic submodels for the concourse, checkpoint and bag zones, data arrays were created to manage inputs and outputs from multiple submodel instances.

Table 2. Model output statistics

	Reported Statistic
Checkpoint	Sum of passengers processed
	Sum of non-passengers processed
	Max processing time for all Checkpoints (minutes)
	Max passengers in queue for all Checkpoints
	Average processing time (Average Queue Time) in Minutes
	Total Checkpoint Utilization
	% of passengers waiting over 10 minutes
	Total # of lanes open for all priority groups
Bag Zone	Count of bag entered across all Bag Zones
	Count of bags processed across all Bag Zones
	Count of alarmed bags across all Bag Zones
	Count of alarms resolved across all Bag Zones
	Max processing time for all Bag Zones in Minutes
	Max bags in queue for all Bag Zones
	Average bag TAT (average delay) in minutes
	% of bags with TAT greater than 10 minutes
	Utilization of EDS primary systems
	Utilization of ETD alarm resolvers

2.6 Model Virtualization

To facilitate modeling of all the various airport configurations, arrays were used to store input data, perform calculations and report output values. The following is an example of such an array used for the Hartsfield-Jackson Atlanta International Airport (ATL) airport. The processing rates in the table are in terms of passengers per hour, or PPH.

Table 3. Example checkpoint configuration array for ATL

Checkpoint	Num Priority 1 lanes	Processing Rate 1 (PPH)	Num Priority 2 lanes	Processing Rate 2 (PPH)	Num Priority 3 lanes	Processing Rate 3 (PPH)	Num Priority 4 lanes	Processing Rate 4 (PPH)	Num Priority 5 lanes	Processing Rate 5 (PPH)	MOE
1	16	180	2	150	4	150	0	0	0	0	10
2	2	180	4	180	0	0	0	0	0	0	10
3	2	180	2	180	0	0	0	0	0	0	10

After verifying prototype results, all model elements were structured so the model could be "virtualized." All data was handled within ProModel in Arrays, User Distributions and Shift files. A single subroutine is called at initialization of the model to read each of the files using READ statements within ProModel to populate the three main data arrays for Flight Data, Checkpoint Configuration and Bag Zone Configuration.

2.7 Submodels: Zero Level Model (ZLM), Concourse, Checkpoint and Bag Zone

The zero level model contains all the attributes, global variables, data arrays, subroutines, and statistical distributions inherent in every ESM model. The zero level model contains all the data management constructs to model any airport regardless of size or configuration.

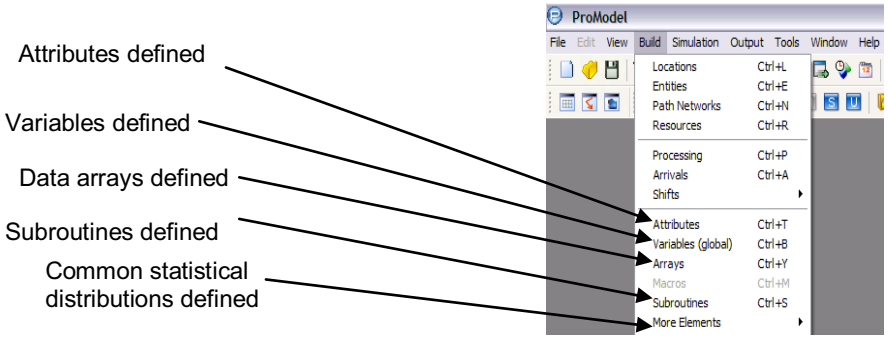


Figure 4. Zero Level Model build constructs

The basic concourse submodel is a single ProModel location with infinite capacity containing logic for processing passengers. While the prototype model was hard coded with checkpoints, bag zones and routing logic, the virtual model is built by merging submodels and programmatically writing the final aspects of the model via the ProModel application programming interface (API). The concourse submodel has two processing records: the first for passengers (arrival delay, check-in, bag divesting) and the second for bags (generate quantity, determine standard or oversize, routing delay to bag zone).

The basic checkpoint submodel contains an infinite capacity queue and a multiple capacity processing location bounded by the number of parallel screening lanes. The routing of passengers to the appropriate checkpoints is written in after the submodels are merged using the ProModel API. Non-passengers are also modeled within the checkpoint submodel. Three processing blocks are used at the checkpoint queues for the processing of passengers, rate-based non-passengers and hourly quantity-based non-passengers. All entities (passengers and non-passengers) are processed at the main checkpoint processing location. Similarly, the bag zone submodel contains a single queue with infinite capacity and a multiple capacity processing location bounded by the number of machines for primary processing. However, the bag zone submodel also has another queue with infinite capacity and multiple server locations for secondary processing of the bags. There is also a capability to utilize secondary processing to help maintain bag processing MOE and promote higher equipment and manpower utilization. This option is called the "Hybrid" mode.

3 PROGRAMMATIC MODEL BUILD, EXECUTION, AND POST PROCESSING REPORTS

3.1 Merged Model Build (MMB)

While it would be possible to write the entire model via the ProModel API, the use of submodels utilizes the common elements shared in the ZLM and leverages the submodel logic and naming features. The process begins by loading the ZLM which contains all the attributes, global variables, data arrays, subroutines, and statistical distributions inherent in every ESM airport model. The overall ESM system database has a master file with configuration information for each airport such as concourse, checkpoint, and bag zone counts and other global information. This meta-data file is used to merge submodels (concourses, checkpoints and bag zones) to model the subject airport. When submodels are merged within ProModel, a suffix can be added to the submodel location names along with a number in order to differentiate among multiple submodels with the same name. Using this schema, an airport with one concourse, one check-

point and one bag zone would generate a Locations table as shown in Figure 5 after merging all submodels with the ZLM.

Locations

Icon	Name	Cap.	Units
	concourse_CC_1	INFINITE	1
	checkpoint_q_CP_1	INFINITE	1
	checkpoint_CP_1	1	1
	bagzone_q_BZ_1	INFINITE	1
	bagzone_BZ_1	1	1
	bagzone_sq_BZ_1	INFINITE	1
	bagzone_s_BZ_1	1	1

Figure 5. Example locations table after merging submodels

After merging all necessary submodels, the ProModel application programming interfaces (APIs) are used to populate the remaining configuration and operational data including secondary bag screening equipment counts and related screening rates. Next, the shifts, variables, user distributions and random number streams are added programmatically to the model following the structure of the ProModel user interface shown in Figure 6.

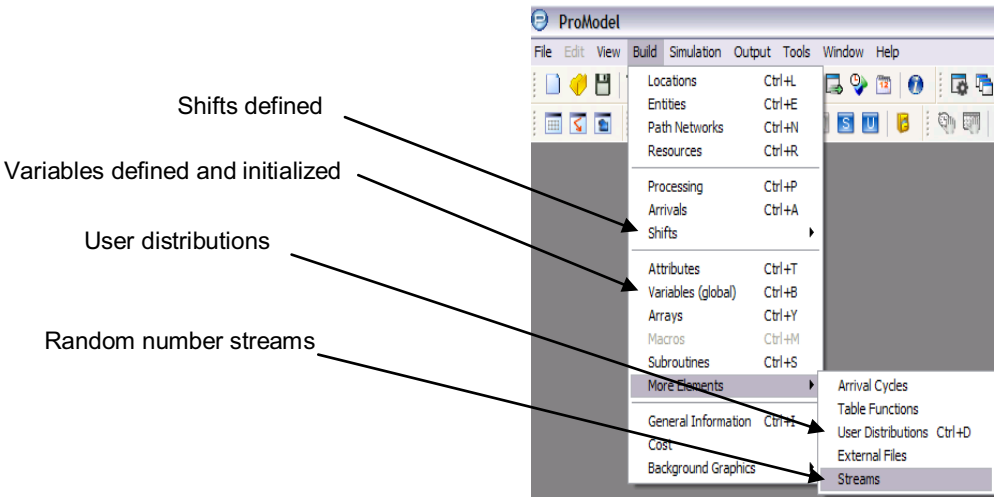


Figure 6. Data and operational constructs added programmatically

3.2 Executing the Merged Model

Once the submodels are merged and all data elements are in place, the model is run for 168 hours of simulation time (one week). The models are executed without animation for maximum run-time performance. While ProModel generates an output summary report, the ESM model writes out a custom system status report with entries for every five-minute interval (2016 total five-minute intervals). Example statistics include average waiting time, average queue length, number of checkpoint lanes open, and number of primary and secondary bag screening equipment deployed. The conversion to manpower requirements and formatting of reports for presentation via the UI are done in post-simulation processing, again to maximize run-time performance and take advantage of the multi-tier architecture shown in Figure 1.

3.3 Use of Generated Output File to Perform Post-processing Calculations and Reporting

The generated output file is used for two purposes. The first is to report back to the user the performance of the modeled system. Given the number of statistics reported, as shown in Table 2, and the reporting intervals, every five minutes, the sample user outputs will not be presented in this discussion. The second use of the output statistics is to determine the staffing requirements for passenger and bag screening. The staff calculation for checkpoints is calculated using lane group staffing constants and the number of lanes from that priority group open during each five-minute period. The following equation represents the staffing calculation for priority groups 1-5:

$$\textit{staff for Priority Group } i = (\#Lanes_i * C1)\ \textit{(unrounded)} + (\#Lanes_i * C2)\ \textit{(rounded up)} + (\#Lanes_i * C3)\ \textit{(rounded down)},$$

where C1, C2, and C3 are staffing constants which represent the combination of regular staff and additional staff needed for checkpoint screening processes. The equation is applied to the number of lanes required (open) to calculate checkpoint staffing requirements for each five-minute interval.

A similar equation for calculating bag zone staff in post processing is determined from EDS and ETD counts. The EDS constants are based on the type of EDS system, and whether the system has on screen resolution (OSR) capability if the type is inline. ETD constants are grouped together for primary ETD systems and alarm resolver positions. The staff equation for bag zones is found below:

$$\textit{staff for Bag Zone } i = (\#EDS*C1)\textit{(not inline)} + (\#ETD*C2)\textit{(primary and alarm resolver positions)} + (\#EDS*C3)\textit{(inline without OSR)} + (\#EDS*C4)\textit{(inline with OSR)}$$

where C1, C2, C3 and C4 are staffing constants which represent staff types for different bag screening equipment and processes. The four types of bag zone equipment configurations include EDS primary, ETD primary, inline and hybrid. The equation is applied to the number of bag screening equipment required (open) to calculate bag zone staffing requirements for each five minute interval. The resulting staff requirement file is then stored for subsequent scheduling through the staff plan optimizer.

4 TECHNICAL CHALLENGES FACED AND SOLUTIONS FOR DATA-DRIVEN MODELS

The challenge was not only to model processes with considerable amounts of data inputs and outputs, but also be a predictive model which provides rough-cut manpower requirements. While many simulation models and most simulation packages support data-driven modeling, they are generally operated by the modeler for a number of experimental scenarios. For this project, the team had the challenge of creating a modeling approach where a model can be built dynamically which represents any airport configuration. In determining the appropriate approach for this type of modeling, it was discussed whether a simulation *language* or simulation *package* would be more appropriate. Due to the large volume of data and the fact that the model was producing capacity planning results as opposed to operational results, it originally appeared that a simulation language would have been more appropriate. However, a simulation language would not have provided a user-friendly simulation platform to maintain the simulation engine, make model and data edits, and expand the use of simulation technology within TSA.

The prototype phase employed many of the simulation software package features such as animation, spreadsheet file interchanges, built-in statistical distributions and pre-configured output report and graphics. However, the final network model had to work without most of these "features" as the end users would not have access to the ProModel UI. An example of where this occurs is in the shift files. The shift files are normally defined using a drag-and-drop interface within ProModel. In the network version, shift files had to be converted from a drag-and-drop UI to a text file and programmatically added after the model merge. The users will only see a dialog box which allows them to add in the hours of operation.

During model development, the team encountered issues which influenced the final simulation design:

- **Challenges of the data interchange between ESM and ProModel**. To overcome data security and access issues, we were relegated to using CSV files for data inputs and outputs. Several of our files contained mixed data types including integers, strings and real numbers. For example the flight data included aircraft capacity (integer), flight identifier (string) and departure time (real number). As ProModel cannot input/output strings via CSV files, we were able to implement a standard naming convention that utilized the concatenate function to serialize the various mixed mode string names.
- **Location of global variables in a merged model**. If global variables are kept in the submodels as opposed to the zero-level model, the merging process will add a suffix for concourse (CC_n), checkpoint (CP_n) and bag zone (BZ_n) to the variable in the merging process. All global variables are defined in the zero-level model and inherited by all of the submodels.
- **Challenges in virtualization.** Since the models are "generic" it is crucial that model locations are placed in a known order in the ProModel Locations table, and the position in the table must be referenced properly. This presented a challenge since each airport could have different numbers of concourses, checkpoints, and bag zones, and thus locations will not be in the exact same position in the table among different airports. Since each submodel has a set number of locations, the position of the table had to be generated as a function in order to refer to it during execution of the subroutines.
- **Reading input files into arrays.** ProModel has methods for reading input files into arrays via an interchange with Excel and through database calls. However, since this particular environment did not have access to Excel nor were we permitted to query the secure databases, input files are programmatically written into the external files tables. We utilize the READ statement in ProModel, which reads in data one record at a time, to import data from the external CSV files to internal data arrays. This method has the potential to be time consuming and error prone with large irregular data sets.

5 DISCUSSION AND CONCLUSIONS: PROGRAMATIC MODEL BUILDING AND BENEFITS

The ProModel APIs provided the capability to dynamically build airport models. Using the merge feature with a zero level model and submodels was helpful in that the processing logic was included within the submodels and the suffix kept track of which concourse, checkpoint, or bag zone was being used. While other software packages could have also handled the type of data used in ESM, ProModel's table structure was advantageous for mapping database information into ProModel tables.

The finished product is anticipated to have many benefits to TSA over the old system. Possibly most important is the ability of users to run multiple instances on multiple CPUs in a distributed environment. This creates a better user experience due to the ability to run multiple models at once, thus reducing simulation execution time and promoting better user productivity. In the previous PC-based system, users were unable to perform other work locally while performing simulation runs.

One of the benefits of migrating from stand-alone models to a networked model is the improvement of network efficiency over stand-alone operation. Instead of having 200 stand-alone versions, there is one repository for data and every user has the most current data available in the system. Having one repository for data aids in the maintainability of the system; data updates such as flight information from the OAG will be imported once, meaning overall data handling, storage and data recovery issues will be minimized. The networked model will eliminate at least 50% of help desk issues since most of the issues currently experienced are due to data migration, updating data, and transferring files from user to user. Further-

more, guaranteeing that users have the correct data improves overall model accuracy, which is important in determining staffing for budget runs.

The ESM model was originally built to replicate the functionality of TSA's current GPSS model. The team worked closely with TSA to understand their current requirements as well as develop ideas for future model enhancements. ESM was built to be flexible and could be extended in the future to not only provide capacity planning but potentially allow TSA to perform studies at an operational level such as examining different passenger routing rules, evaluating alternate checkpoint and concourse designs and planning for new screening technologies and processes.

ACKNOWLEDGMENTS

We would like to thank all of the TSA employees who helped provide operational inputs and reviews as we transitioned from SAM to ESM. The TSA SAM help desk was a constant source of assistance as well as the TSA IT operations group. TSA does not endorse any particular company or product and although Mr. Coffman's inputs are based on his TSA experiences, the views expressed do not necessarily reflect those of the agency. Lastly, the consultants at Procentrix did a remarkable job of building the data bases, execution environment and user interfaces.

REFERENCES

Regal Decision Systems, Inc. 2009. TSA Staffing Model Version 5.4.0 User Guide. Belcamp, MD: Regal Decision Systems, Inc.

AUTHOR BIOGRAPHIES

DAVID KALASKY is an IBM Senior Managing Consultant specializing in Business Analytics and Optimization (BAO). In addition to IBM, David has worked in consulting roles with Sun Microsystems, Motorola and Westinghouse. He also has extensive experience in simulation software, modeling and analysis working with Systems Modeling Corporation as Product Services Manager and as a Regional Manager for ProModel Corporation. He has several WSC proceedings publications and was the business chairman for the 1990 WSC. David has an BSIE degree from the University of Kentucky and an MSOR from Penn State University. His email address is <drkalasky@us.ibm.com>.

MICHAEL COFFMAN is the Program Manager for the Workforce Allocation Program at The Transportation Security Administration (TSA) / Department Of Homeland Security (DHS). He has managed the development and support of TSA's staffing model solutions since 2006 and worked as a contractor supporting the development and application of the simulation model preceding his employment with TSA. Previous to being involved with TSA, Michael spent five years working at FedEx as an Industrial Engineer working to improve productivity and the operational effectiveness of package delivery operations. Michael holds a degree in Manufacturing Engineering Technology from the University of Memphis. His email address is <michael.coffman@tsa.dhs.gov>.

MELANIE DE GRANO is a Senior Consultant in IBM's Business Analytics and Optimization (BAO) group. She applies significant knowledge in optimization and simulation modeling to implement business solutions and has experience in both manufacturing and service industries. Melanie has taught undergraduate courses in process quality engineering, engineering statistics, and engineering economics. She received her BS, MS, and PhD degrees in Industrial Engineering and Operations Research from Penn State University in 2002, 2004, and 2007, respectively. Her email address is <melanie.degrano@us.ibm.com>.

KEVIN FIELD is an Integrated Product Team Director at ProModel Corporation for the Process Simulator and ProModel suite of simulation products. He is responsible for the quality, support, training, development, and partnering initiatives of these products. Kevin also directs the Technical Support, Training, and IT departments and is responsible for ProModel's relationship and engagements with Microsoft. Kevin has been with ProModel for 17 years and has held multiple positions such as Quality Assurance Engineer, Technical Support Engineer, Software Engineer, Senior Applications Engineer, Senior Product Specialist, Consultant, and Director of Engineering Development. He has a Bachelor of Science degree in Mechanical Engineering from Brigham Young University. His email address is <kfield@promodel.com>.