Managed Readiness Simulator (MARS) V2

The Graphical Simulation Output System

Stephen Okazawa
Mike Ormrod
Force Readiness and Air Systems Section

Defence R&D Canada – CORA

Technical Memorandum
DRDC CORA TM 2012-038
February 2012

Principal Author

Original signed by Stephen Okazawa

Stephen Okazawa

Defence Scientist

Approved by

Original signed by D.M. Bergeron, Ph.D.

D.M. Bergeron, Ph.D.

Section Head Force Readiness and Air Systems

Approved for release by

Original signed by Paul Comeau

Paul Comeau

Chief Scientist

Defence R&D Canada – Centre for Operational Research and Analysis (CORA)

Abstract

The Managed Readiness Simulator (MARS) is a versatile program that allows the user to quickly simulate a wide range of Canadian Forces readiness scenarios to determine if the Resources of an Establishment are able to satisfy the requirements of a set of operational Tasks. The first version of MARS (V1) demonstrated the managed readiness modelling concept and was successfully applied to a preliminary analysis of the Army's plans to generate the forces for Task Force Afghanistan. A Graphical User Interface (GUI), developed under software development contracts, integrated scenario setup, execution and output analysis tools into a single package written in Visual Basic. Following a recent major redesign, MARS V2 now incorporates an advanced software architecture that integrates database technology into simulation execution and a more flexible managed readiness model that includes support for Establishment dynamics. MARS V2 also implemented a new system for generating graphical simulation output. In practice, the need to customize existing outputs or create new outputs occurs frequently, and the expected turn-around time for these changes is short. The new MARS V2 output system allows the analyst to quickly mock up new simulation outputs using standard tools (MS Access and Excel) and then to automate the generation of those outputs so that they can be included in the standard set of outputs available in the MARS application. This paper documents the implementation of the new MARS V2 output system.

Résumé

Le programme de simulation de gestion de la disponibilité opérationnelle (programme MARS) est un programme polyvalent qui permet à l'utilisateur de rapidement simuler une vaste gamme de scénarios de disponibilité opérationnelle des Forces canadiennes afin de déterminer si les ressources d'un établissement peuvent répondre aux besoins propres à un ensemble de tâches opérationnelles. La première version du programme MARS (V1) a démontré le concept du modèle de gestion de la disponibilité opérationnelle. Elle a été utilisée avec succès lors d'une analyse préliminaire des plans de l'Armée visant à mettre sur pied les forces pour constituer la Force opérationnelle Afghanistan. Une interface graphique (GUI), développée dans le cadre de contrats de développement de logiciel, a permis d'intégrer la mise en place et l'exécution d'un scénario ainsi que des outils d'analyse de données dans un seul ensemble rédigé en Visual Basic. Après une importante restructuration, le programme MARS V2 incorpore désormais une architecture logicielle plus sophistiquée qui intègre une technologie de traitement de bases de données dans l'exécution de la simulation, ainsi qu'un modèle de gestion de la disponibilité opérationnelle plus souple qui comprend du soutien au niveau de la dynamique de l'établissement. Un nouveau système a également été mis en place pour générer des résultats de simulation graphique. En pratique, il arrive fréquemment qu'il faille modifier des données existantes ou en créer d'autres. Les délais pour effectuer ces changements sont alors très courts. Le nouveau système de données du MARS V2 permet à l'analyste de faire un échantillonnage rapide des nouvelles données de simulation à l'aide des outils standard (MS Access et Excel) et d'automatiser la production de ces données afin qu'elles puissent être ajoutées à l'ensemble des données normalisé dans le programme MARS. Le présent document porte sur la mise en œuvre du nouveau système de données de MARS V2.

This page intentionally left blank.

Executive summary

Managed Readiness Simulator (MARS) V2: The Graphical Simulation Output System

Stephen Okazawa; Mike Ormrod; DRDC CORA TM 2012-038; Defence R&D Canada – CORA; February 2012.

Background: The Managed Readiness Simulator (MARS) is a software application being developed at Defence Research & Development Canada - Centre for Operational Research & Analysis as an Applied Research Program managed by the Land Force Operational Research Team. MARS is designed to quickly simulate a wide range of readiness scenarios to determine if the resources of an Establishment are able to satisfy the requirements of planned operations.

The first version of MARS (termed V1) successfully conducted a preliminary analysis of the Army's plans to generate forces for Task Force Afghanistan. In the process of conducting this analysis, several aspects of the MARS V1 architecture and design were identified as limiting the potential of MARS to address future problems. In particular, modelling the dynamics of the Establishment (the creation, advancement and release of Establishment Resources), was not considered feasible within MARS V1. Certain design aspects of MARS V1 were also restrictive in terms of the types and complexity of scenarios that could be represented. Additionally, while the MARS V1 output system was powerful, it was also rigid. The outputs were implemented as part of the graphical user interface (GUI) which was written in Visual Basic code and maintained by contracted programming support. This meant that customizing existing outputs and developing new outputs required by clients was a slow process as the analyst did not have direct access to or control over the MARS output system.

Results: Prior research produced a new technical platform for a new version of MARS (termed V2) which greatly expanded its potential to address more complex scenarios. This platform, called the Simulation Runtime Database (SRDB) approach, was exploited to develop more advanced modelling capabilities including Establishment dynamics. This involved the full reimplementation of MARS using the techniques and capabilities available in the new platform. The reimplementation included a redesign of the MARS output system. Largely facilitated by the new SRDB architecture, the new MARS V2 output system enables the analyst to create customized graphical outputs using standard tools: MS Access and Excel. Then a special instruction set was designed that allows the analyst to automate the process of generating that output. Once the process is automated, the output can then be incorporated into the set of outputs available within the MARS application.

Significance: This paper documents the operation of the new MARS V2 output system. The principal achievement of the new system is that the analyst can quickly create customized outputs that can be readily incorporated into the MARS application. This allows the MARS analyst to rapidly respond to client requirements to look at simulation results in a certain way. As no two problems are alike, the information that is relevant to the client can differ greatly from one simulation scenario to the next. Therefore, the ability to rapidly create new graphical outputs for the MARS application has a significant impact on the analyst's ability to produce informative deliverables for clients in a timely manner.

Sommaire

Managed Readiness Simulator (MARS) V2: The Graphical Simulation Output System

Stephen Okazawa; Mike Ormrod; DRDC CORA TM 2012-038; R & D pour la défense Canada – CORA; Février 2012.

Contexte: Le programme de simulation de gestion de la disponibilité opérationnelle (MARS) est une application logicielle en cours de développement au Centre d'analyse et de recherche opérationnelle de Recherche et développement pour la défense Canada (RDDC). Il s'agit d'un programme de recherche appliquée géré par l'Équipe de recherche opérationnelle de la Force terrestre. Le programme MARS est conçu pour rapidement simuler une vaste gamme de scénarios de gestion de la disponibilité opérationnelle dans le but de déterminer si les ressources d'un établissement sont en mesure de répondre aux besoins des opérations planifiées.

La première version de MARS (nommée V1) a permis d'effectuer avec succès une analyse préliminaire des plans de l'Armée visant à mettre sur pied des forces pour la Force opérationnelle Afghanistan. Dans le cadre de cette analyse, plusieurs aspects de l'architecture et de la conception de MARS V1 ont été identifiés comme étant des facteurs limitant la capacité de MARS à traiter les problèmes à venir. En particulier, la modélisation de la dynamique de l'établissement (la création, le développement et la publication des ressources de l'établissement) n'était pas considérée possible avec le programme MARS V1. Certains aspects de la conception de MARS V1 étaient également restrictifs en termes de types et de complexité de scénarios pouvant être représentés. En outre, le système de données de MARS V1 était puissant, mais limitatif. Les données étaient intégrées dans l'interface graphique (GUI) rédigée en Visual Basic et maintenues par un soutien à la programmation en sous-traitance. Ainsi, il était long de modifier des données existantes ou d'en créer de nouvelles pour les clients, car l'analyste n'avait pas directement accès au système de données de MARS, ni même le contrôle.

Résultats: Des recherches précédentes ont produit une plateforme technique pour une nouvelle version de MARS (nommée V2) qui a grandement augmenté sa capacité à traiter les scénarios plus complexes. Cette plateforme, appelée approche de la base de données d'exécution de simulation (Simulation Runtime Database (SRDB)), a été utilisée pour élaborer des capacités de modélisation plus sophistiquées, y compris la dynamique de l'établissement. Cela impliquait une nouvelle mise en œuvre complète de MARS en utilisant les techniques et les capacités disponibles dans la nouvelle plateforme. La nouvelle mise en œuvre comprenait une restructuration du système de données du MARS. La nouvelle architecture de l'approche SRDB facilite grandement le travail de l'analyste, car il lui permet de modifier ou de créer des données graphiques dans le système du MARS V2 à l'aide d'outils standards : MS Access et Excel. Des directives spécifiques ont été établies afin de permettre à l'analyste d'automatiser la création de ces données. Une fois le processus automatisé, les données peuvent être intégrées dans l'application MARS.

Portée: Le présent document porte sur le fonctionnement du nouveau système de données MARS V2. La principale réalisation du nouveau système est que l'analyste peut rapidement créer des données personnalisées pouvant déjà être intégrées dans l'application MARS. Ainsi, l'analyste

peut répondre rapidement aux demandes des clients qui désirent obtenir des résultats de simulation d’une certaine manière. Les problèmes sont tous différents, et l’information pertinente pour un client peut différer considérablement d’un scénario de simulation à l’autre. De ce fait, la capacité à créer rapidement de nouvelles données graphiques pour l’application MARS a une grande incidence sur la capacité de l’analyste à fournir de l’information aux clients en temps opportun.

This page intentionally left blank.

Table of contents

List of figures

List of tables

This page intentionally left blank.

1 Introduction

The Managed Readiness Simulator (MARS) is a versatile program that allows the user to quickly simulate a wide range of Canadian Forces (CF) readiness Scenarios to determine if the Resources of an Establishment are able to satisfy the requirements of a set of planned operational Tasks. The flexibility of MARS allows diverse operational tasks to be defined as processes composed of activities that place specific resource demands on the Establishment. The software also provides a graphical user interface (GUI) that facilitates the creation and execution of simulation scenarios and the analysis of simulation output. Ultimately, MARS is intended to be used as a decision support tool for senior commanders of the CF. It provides them with forecasts of the impact of proposed changes to lines of operation, the Establishment, the readiness plan, CF policy, and other factors that may affect the CF's ability to satisfy operational demands and to maintain the health of the Establishment. A more detailed description of the motivation behind the development of MARS and its potential applications can be found in [1].

MARS is being developed by Defence Research & Development Canada - Centre for Operational Research & Analysis (DRDC CORA) as an Applied Research Program managed by the Land Force Operational Research Team (LFORT). The first version of MARS (termed MARS V1) was developed in Rockwell's Arena simulation software [2] to demonstrate the managed readiness modelling concept. The GUI, developed under software development contracts, integrated scenario setup, execution and output analysis tools into a single package written in Visual Basic. Previous publications document the design and implementation of MARS V1 [3,4]. Completed in 2007, MARS V1 was successfully applied to a preliminary analysis of the Army's plans to generate the forces required for Task Force Afghanistan [5].

The development of a second version of MARS (termed MARS V2) was begun in 2008 in order to advance its capabilities in specific areas, in particular: the ability to scale to larger scenarios and to model the dynamics of the Establishment. Achieving these design goals involved developing a new Arena simulation architecture that integrated database technology into simulation execution [6] and a new managed readiness model with more advanced feature set able to support Establishment dynamics [7,8].

Also a part of MARS V2's enhancements was a new approach to visualizing simulation output. In MARS V1, pre-programmed graphical output was incorporated into the graphical user interface (GUI) which was programmed in Visual Basic (VB). This approach is powerful in that virtually any graphical output and supporting interface can be programmatically created. But it is also inflexible because the GUI's VB code is not readily accessible to the analyst. Thus modifications to existing output and the creation of new output must be communicated to and implemented by the contractor(s) responsible for the MARS GUI code. In practice, the need to customize existing outputs or create new outputs as required by MARS clients occurs frequently, and the expected turn-around time for these changes is short. However, the development time required to implement new graphical output in the MARS V1 GUI was long. Therefore, a more streamlined approach to creating graphical output and incorporating it into MARS was desired.

Rather than building pre-programmed output into the GUI, MARS V2 allows the analyst to design graphical output using Microsoft (MS) Access queries to extract data and MS Excel spreadsheets to visualize the data. The processes of querying simulation data and visualizing output are then automated using a simple set of instructions entered into an Excel spreadsheet. These instructions are executed by an interpreter which was programmed in Excel Visual Basic for Applications (VBA). The Excel file containing the instructions and visual output for a given analysis can then be linked to the MARS V2 GUI so that it can be accessed by MARS users. When the user requests a certain graphical output, the GUI launches an Excel window which then automatically executes the appropriate Access queries, generating the required data, and then displays the result in chart or tabular form. Because the result is presented in an Excel window, the full functionality of Excel is available to the analyst to further customize the output and to move it to other formats for reporting or presentation. This approach to incorporating graphical output into MARS V2 allows for rapid response to client requirements using a widely-available and accessible toolset.

The purpose of this paper is to document the implementation of the MARS V2 output system. Sections 2 and 3 first provide a review of MARS concepts, terminology and components. Section 4 provides an overview of the underlying methodology and features of the MARS V2 output system. Section 5 describes the implementation of the output system in detail and will be of primary interest to MARS developers in the event that the functionality of the output system needs to be modified or enhanced. Section 6 provides a detailed example of creating a new output file. Section 7 provides a brief description of each of the outputs currently available in MARS V2. Finally, Section 8 provides concluding remarks.

2 MARS concepts and terminology

The MARS program is designed to simulate a given readiness *Scenario* by forecasting the ability of an *Establishment* to generate the *Resources* required to satisfy a set of *Tasks* occurring over time under a given set of conditions. The program also records the state of every Resource throughout the simulation; therefore the results can be used to determine the utilization level of a unit within the Establishment or of a specific group of Resources.

The program currently models three types of Resources: Personnel, Equipment and Facilities. Every Resource occupies a *Slot* in an *Organization* as shown in Figure 1. In general, the *Attributes* of each Slot define a particular *Resource Requirement* of the Organization and the Slot can only be occupied by a single Resource that satisfies the requirement. For example a Slot may assert that it can only be filled by a Resource that satisfies the criteria: *rank == Captain AND occupation == Infantry*. However, special *Overflow* Slots can also be built into the Establishment that are allowed to contain many Resources. These Slots are sometimes needed to store extra Resources that do not satisfy the Resource Requirement of any Slot or that cannot be assigned to a Slot because eligible Slots are currently occupied.

An Organization consists of a group of Slots that define the Resource Requirements of a unit. They are typically arranged in a hierarchical tree structure where only the terminal nodes of the tree contain Slots. There are two types of Organizations. Establishment Organizations define the units that contain the Resources available in the Scenario. Theatre Organizations define templates of the units required by the Tasks being modelled and do not contain any Resources. During the simulation, the program uses these templates to create a group of Slots that will be filled by Establishment Resources selected to perform a Task.

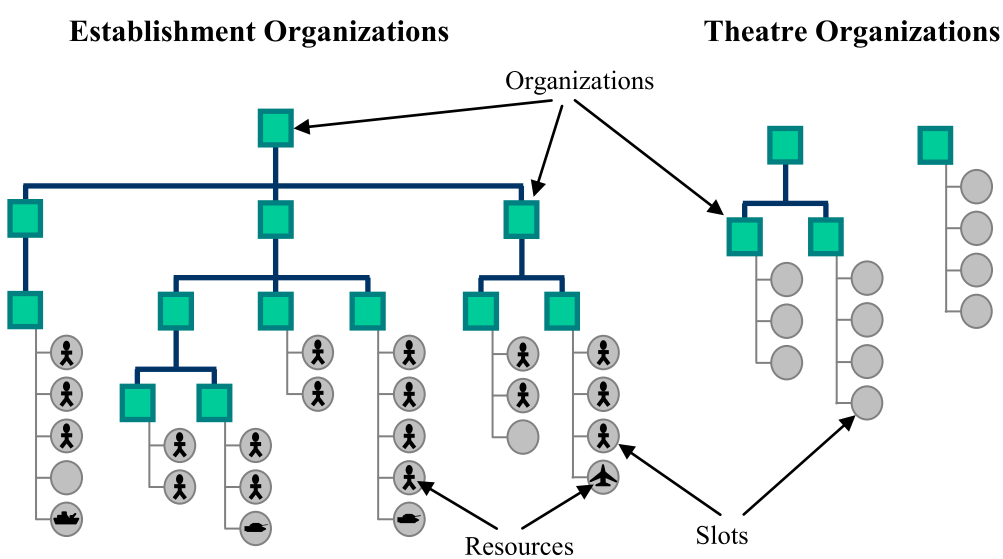


Figure 1: Establishment Organizations and Theatre Organizations showing Slots and Resources

Each Resource has Attributes that define its current state. Attributes store the information that determines whether a Resource can be chosen for a particular Task. Examples of Resource Attributes include a person's rank and qualifications. Other Attributes indicate whether the Resource is currently busy and whether there are any restrictions on what the Resource is allowed to do. In general, Attributes define the capability and availability of a Resource. These Attributes along with the Organization to which the Resource is attached are used to determine the eligibility of the Resource to be used by a given Task.

The operations and events being simulated in a MARS Scenario are represented by *Tasks* within the model. Each Task is broken down into *Activities* which are scheduled within the Task so that they will occur in a specified order. Activities are responsible for assembling the Resources they require. Activities can also be linked together to pass Resources from one Activity to another or to enforce dependencies. There are two types of Activities in the model:

1. A Process Activity temporarily employs Resources for a certain period of time and may alter their state upon commencement and completion.

2. An Event Activity changes the state of the selected Resources at a single point in time.

Each Activity is triggered in the simulation according to timing and Resource constraints. Activities simulate everything from training and operational Tasks to recruitment and retirement events or, if referring to equipment, acquisition and disposal events. The Activity construct is illustrated below in Figure 2.

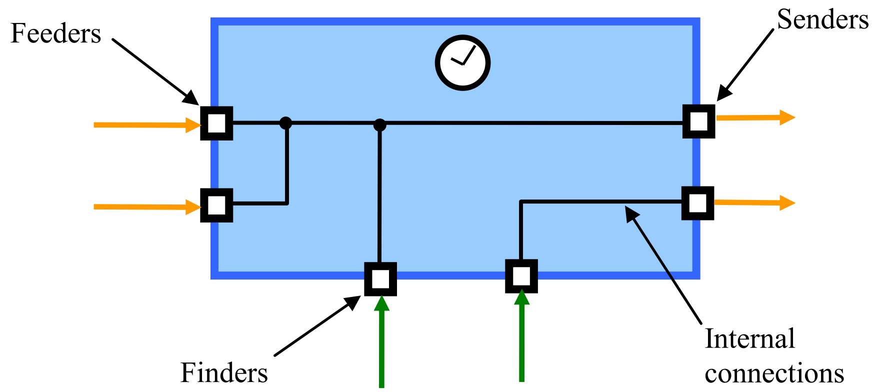


Figure 2: The Activity construct showing Feeders, Finders and Senders and internal connections.

When an Activity is triggered and starts processing, it must acquire the Resources it needs to carry out its function. Resources enter an Activity as part of a resource group (*ResGrp*) through either a *Feeder* or *Finder* node and exit through a *Sender* node. An Activity may have multiple Feeders, Finders and Senders, and must have at least one Feeder or one Finder in order to act on at least a single ResGrp. A ResGrp is a set of Resources and Slots where each Resource occupies a single Slot and each Slot is either empty or occupied by at most one Resource. ResGrps are created by Finders which select Resources from the Establishment to participate in the Activity. Figure 3 illustrates the steps carried out by the Finders.

First, the Finder identifies the Theatre Organizations that contain the Slots that define the *Resource Requirement* for the Activity. For example, a Disaster Assistance Response Team Triage Unit might contain Slots for a medical officer, a medical technician, and a nurse.

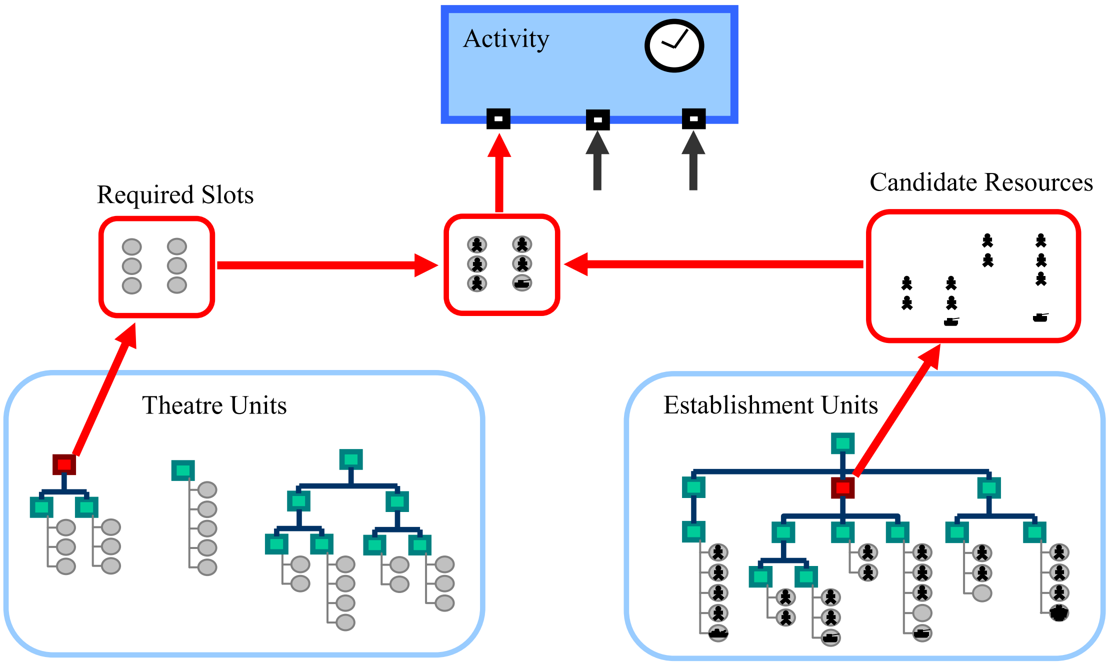


Figure 3: Resource selection process used by an Activity Finder.

Next, the Finder specifies a prioritized list of Establishment Organizations that may be searched to find Resources to participate in the Activity. This prioritized list is subject to constraints that can be used to limit the number of Resources taken from a given Organization and to filter for Resources with certain Attributes. The Finder also verifies that these Resources have not already been assigned to a conflicting Activity. This produces a list of *Candidate Resources*.

The Finder then attempts to fill each Required Slot with one of the Candidate Resources by comparing the Attributes of the Resources to the requirements of each Slot. If a suitable match is found, the Resource is assigned to the Slot and becomes part of the ResGrp being created by the Finder. To maximize the number of successful matches, the Finder attempts to assign the least qualified Candidate that meets the requirements of each Slot. For example, a Slot that can be filled by either a Major or Lieutenant Colonel will be preferentially assigned a Major because Lieutenant Colonels, being of higher rank, are most often in shorter supply and may be required by other Slots with more stringent requirements. This process is repeated for all Finders, with each Finder creating a ResGrp.

Note that the process of selecting Establishment Resources and matching them to Activity Slots is not deterministic. Resources having equivalent qualifications with respect to a given set of requirements are randomly sorted. Thus multiple iterations of the same simulation scenario can produce slightly different results. However, the user has option to set the random number generation seed value if there is a need to exactly reproduce the events of a given run.

Activities also acquire Resources through Feeders which receive ResGrps that were created by a preceding Activity and passed on through one of that Activity's Senders. After acquiring its Resources through its Finders and Feeders, the Activity verifies that a specified minimum number of the required Slots have been filled. If this minimum requirement is not met, the Activity fails and the Resources are released. If sufficient Resources are found, the Activity takes control of the selected Resources, altering their Attributes to reflect the nature of the Activity and employing them for the duration of the Activity.

Each Feeder and Finder is connected internally to a Sender. Upon Activity completion, each ResGrp is passed to a Sender which alters the Attributes of the Resources within the ResGrp to reflect the completion of the Activity. Each Sender is then responsible for either passing the ResGrps to the Feeder node of a follow-on Activity or for releasing the Resources within the ResGrp back to their Establishment unit. Senders and Feeders are the connection nodes that allow Activities to be linked together within a larger Task. When the Activity's processing time has finished and each ResGrp has exited through a Sender, the Activity is complete. Similarly, when all of the Activities of a Task are finished, the Task is complete.

To allow Tasks to be reused within a Scenario and to control when they begin, *Task Generators* are used to assign a start time to a Task. Multiple instances of a Task can be generated on a *Rotation* schedule to model the repetition of a Task such as a cycle of deployments that make up a continuous operation. When all of the Task Generators have been processed, and all of their associated Tasks are finished, the MARS Scenario is complete and the simulation stops.

From the outputs generated by the MARS simulation, the extent to which the Establishment was able to supply the Resources required for all the Tasks being modelled can be measured. More specifically, the output is analyzed to determine how successful each Finder was in creating its ResGrp from the Establishment. By aggregating the results from the Finders, the user can determine how successful the Establishment was in generating the required Resources for each Activity, Task, Task Generator, and the entire Scenario. Similarly, the states of the Resources within the Establishment can be tracked over time. These results can be combined to plot the state over time of a selected group of Resources, a unit or group of units, or the entire Establishment.

MARS is a versatile tool with many potential applications. Its strengths are its generic constructs that allow users to quickly simulate virtually any force generation Scenario and its outputs that provide users with the ability to aggregate and drill down into the simulation results to identify the causes of a particular outcome. The ability to forecast how successfully an Establishment can generate the forces required to satisfy both operational and sustainment demands will provide decision makers with invaluable information that currently is unavailable.

3 MARS application components

The MARS application consists of three major components, shown in Figure 4: a Scenario Database, a Managed Readiness Model, and a Graphical User Interface (GUI). The Scenario Database stores all the data that define a specific Scenario. The Managed Readiness Model is the discrete event representation of the process of selecting and employing Resources. The GUI allows the user to interact with the database and the model by performing three management functions. As the Input Manager, it is responsible for facilitating the transfer of Scenario data into the Scenario database. The input data consist of the Tasks, the Establishment, plans, and policies to be modelled in the proposed Scenario. The data may be input directly through the GUI input screens or imported from an external source such as a Corporate database or spreadsheet. As the Simulation Manager, the GUI controls the execution of the Simulation Scenario. Finally, as the Output Manager, it allows a user to analyze the simulation results and to generate output reports.

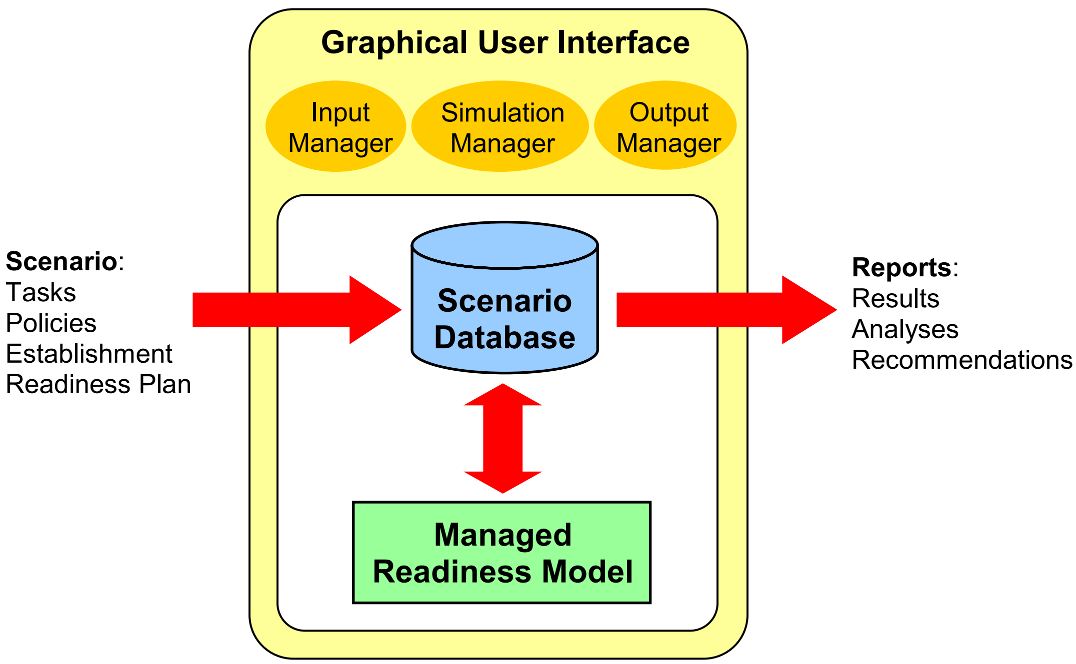


Figure 4: Components of the MARS Application with data transfer shown as red arrows.

4 MARS V2 output system

During a MARS V2 run, a large amount of data is captured that can be analyzed to make an assessment of a given simulation scenario. The MARS V2 architecture tightly integrates an Arena discrete event model with an Access database [6]. The database manages all of the data that represents the initial conditions of the simulation, the current state of the simulation at any point during a run, and the record of simulation events. The Arena discrete event model continuously updates the database as events occur in the simulation. Consequently, for most simulation objects such as Resources and Activities, a complete history of their actions during a run is present in the database. The analysis of this data allows the analyst to determine in detail what happened during the run, which parts of the scenario performed as planned, which parts encountered problems, and what was the nature and cause of these problems.

MARS data generally falls into one of two categories that are both relevant in assessing a given readiness scenario. The first category of data relates to measuring success in conducting operations. This includes data on how successfully Activity Finders acquired Resources from the Establishment, whether Activities ran as scheduled, the number of successfully completed Activities in a Task, and the number of successfully completed Tasks in a scenario. The second category of data relates to measuring the health of the Establishment as a result of conducting operations. This includes data on the states of all resources and slots in the Establishment, when Resources were busy, which Organizations were busiest, which Organizations were under-utilized, which Establishment slots were filled and which were empty.

These data allow several types of analyses to be performed to generate useful outputs. Aggregate outputs summarize data from the whole simulation, for example, displaying the number of successfully completed Activities in each Task. Time-based outputs show the state of simulation objects over time, for example, plotting the number of deployed Resources in an Organization over time. Snapshot outputs show the state of the simulation at a specific time, for example, displaying the number of Resources that meet certain criteria on a particular date.

The concept that underlies the MARS V2 output system is to allow the analyst to generate useful graphical output from MARS data using standard tools and then to automate the procedure so that the output can be incorporated into the MARS V2 GUI and accessed again with a few clicks. The advantage of this output system is that new outputs or customized outputs can be built and incorporated immediately into the MARS V2 GUI by the analyst. In the previous version of MARS, new outputs had to be programmed by the contractor(s) responsible for the development of the MARS GUI code which resulted in a much longer turn-around time when responding to clients' requests to view simulation output data in a particular way.

4.1 Structure of MARS V2 outputs

In developing a system that automates the generation of graphical output, a general structure and toolset for conducting post-simulation analysis was assumed. This structure and toolset, shown in Table 1, was chosen to be as flexible as possible while still allowing the process to be easily automated and incorporated into MARS.

Table 1: Structure and toolset for MARS V2 output generation.

Step	Description	Tool
1	Narrow the scope of the analysis to an area of interest	Access SQL queries
2	Extract the relevant data, process it, and format it	Access SQL queries
3	Visualize the data in chart or tabular form	Excel

For the first step, above, the analyst isolates the data relating to a particular area of interest, such as a group of Activities or Resources or a particular time during the simulation that warrants closer examination. Since the data produced by a MARS V2 run is resident in an Access database, this step is conveniently performed using Access Structured Query Language (SQL) queries to isolate the data relating to the subject of the analysis.

In the second step, the raw data from Step 1 is processed to generate results in a form that can be readily visualized as a chart or table. Again, because the data MARS produces is already resident in an Access database, Access queries are the most powerful and convenient tool for this type of data processing and preparation. Since MARS V2 is able to handle very large datasets and produces a large amount of data during a run, SQL queries are often the only practical tool able to process MARS simulation data. Very complex operations can be performed in this step by using multiple queries, nested queries, and intermediate tables.

In the third step, the results that were prepared in Step 2 are imported to another tool designed for data visualization. In most cases the most appropriate tool for this purpose is MS Excel. Though other tools could be used for this step such as MatLab, it was assumed that MS Excel is the more practical choice because of its wider install base and compatibility with other MS Office applications making dissemination of outputs straightforward. However, it is acknowledged that Excel lacks some capabilities available in other tools.

It is assumed that the vast majority of practical analyses of MARS V2 data can be conveniently carried out using this structure and toolset. Once an analysis producing a desired output has been developed in the form defined above, the procedure can be automated and incorporated into the set of standard outputs available in the MARS V2 GUI.

4.2 Automation of MARS V2 outputs

The automation of the process of generating a MARS V2 output is achieved using an MS Excel file with special features built into it. The file is structured in a standard way that the MARS V2 GUI can interact with. The graphical output in Step 3 of the output generation process is built in this Excel file. The main feature of the file is that it contains an interpreter programmed in Visual Basic for Applications (VBA) that executes sequences of instructions entered into certain Excel sheets. The instructions specify actions to be carried out for Steps 1 and 2 of the output generation process. The actions include executing SQL queries on the MARS database and storing query results in Excel sheets. Once the appropriate data has been retrieved from the database, Excel's data visualization features, such as charts and pivot tables, will respond to the updated data and display the output.

The instruction sequences that the interpreter executes are stored in sheets formatted as shown in Table 2 and are processed row by row from top to bottom. Each instruction takes the form of the example shown. First, a row with an entry in the Action column specifies the action to be taken, other columns in this row remain blank. The rows following the action row specify parameters that adjust what the action will do. In the example shown, the action is to execute an SQL query, the first parameter is the SQL query to be executed, and the second parameter is the name of the Excel sheet that the query result should be stored in.

The comment field is ignored by the interpreter but may be used by the analyst to document a sequence of instructions. The debug field gives the analyst feedback from the interpreter in the event that an error interrupts execution. If an error occurs, the interpreter will write an error message in the debug column at the line that produced the error.

Table 2: Excel sheet format for interpreter instructions.

Action	**Parameter**	**Value**	**Comment**	**Debug**
SQL_query				
	sql statement	select * from Table		
	sheet name	OutputData		

The set of instructions that the interpreter understands is shown in Table 3 including the parameters required, example parameter values and a description. The interpreter expects the parameters to be supplied in the order specified in this table. Errors are likely to occur if parameters are supplied in a different order. In most cases, the interpreter only reads the contents of the value column and ignores the parameter column, however, for the Obj_exe and Obj_query actions the contents of the parameter column are important as noted in the comments.

Table 3: Output automation instruction set.

Action Name	Parameters	Example Values	Description
Database	database name database path	Scenario.mdb D:\Data\Scenarios	Opens a connection to the specified database. This action must be the first instruction in any sequence of instructions.
Import	sheet name table name	DataSheet DataTable	Imports data from the specified database table to the specified Excel sheet.
Export	sheet name table name	DataSheet DataTable	Exports data from the specified Excel sheet to the specified database table.
Import_time			Imports the SelectedOutputTime field from the Scenario Information table in the database to the Sel Output Time field on the FRONT sheet in the Excel file.
Export_time			Exports the Sel Output Time value on the FRONT sheet in the Excel file to the SelectedOutputTime field in the Scenario Information table in the database.
SQL_exe	sql statement	delete from Table	Executes an SQL query.
SQL_query	sql statement sheet name	select * from Table OutputData	Executes an SQL query that returns data and stores the result in the specified Excel sheet.
Obj_exe	obj name param1 param2 …	SaveTempTable 2 4 …	Executes an Access query object. The "obj name" parameter specifies the name of the query object. The remaining parameters and values are passed as query parameters to the query object. For the query parameters, the names in the parameter column must match the name of the parameter defined in the Access query object. The number of query parameters that can be specified after the "obj name" is unlimited.

Obj_query	obj name sheet name param1 param2 …	GetOrgSummary OutputData 2 4 …	Executes an Access query object that returns data. The operation is identical to the Obj_exe action except that a sheet name is specified indicating where the query result will be stored to.

For Step 1 of the output generation process, the user specifies simulation objects and/or a snapshot time that the analysis should focus on. These user selections are stored in both the database and spreadsheet and they must be kept synchronized. The Import, Export, Import_time and Export_time actions are used to maintain this synchronization. This will be discussed in more detail in the next section.

The Obj_exe and Obj_query actions execute stored query objects. These are query objects that have been created and saved in Access. In most cases, this is the recommended way of creating queries for MARS outputs because the queries can be built using the Access visual query designer. This makes building and modifying complex queries relatively straightforward, whereas writing pure SQL strings becomes unmanageable beyond simple queries.

4.3 Organization of the Excel output file

The MARS output Excel file contains three instruction sequences separated into different sheets called SYNC_FromDB, SYNC_ToDB, and RUN. Each of these sheets has columns formatted as in Table 2, above. The SYNC_FromDB sheet contains import instructions that synchronize the Excel spreadsheet with the Access database. This overwrites the Excel data specified in the instructions with the data from the table in the database. The SYNC_ToDB sheet contains export instructions that synchronize the database with the spreadsheet. This overwrites the database data specified in the instructions with the Excel data. The RUN sheet contains the set of instructions that processes and retrieves the output data so that it can be visualized in an Excel chart or table.

The synchronization sheets, SYNC_FromDB and SYNC_ToDB, allow data to be passed back and forth between Excel and Access. This is used to manage settings that specify the scope of the analysis for Step 1 of the output generation process. For example, the user may wish to enter a list of Activities that they want to view in an output. These Activities can be entered in an Excel sheet and then copied to a database table by calling the Export action in the SYNC_ToDB instructions. This table can then be used by the queries executed in RUN instructions to restrict the output to display only the Activities specified by the user. The SYNC_FromDB instructions will typically perform the opposite actions of SYNC_ToDB, allowing scope data in the database to be imported into the Excel file.

In most cases, the passing of data back and forth between the database and the Excel file is managed by the GUI. The GUI provides several dedicated interfaces that allow users to define the scope of their analysis. There are currently four such dedicated interfaces:

- An interface to select Task Generators, Tasks, Activities and ResGrps;
- An interface to select Organizations;
- An interface to define filters that restrict which Organizations, Slots and Resources will be included based on their attributes; and
- An interface to select the time to be used in a snapshot output.

The filter interface can only be used in conjunction with the Organization selection interface. When an Organization is selected, the analysis performed in the RUN instructions typically searches within the Organization to determine its sub-Organizations, the Slots belonging to these Organizations and the Resources occupying these Slots. Organizations, Slots and Resources all have attributes and the filter interface can be used to restrict the output to Organizations, Slots and Resources that have certain attributes. For example, the user may wish to refine an output to display information only on Slots that are vacant, or on Resources that have a certain qualification.

As the user makes his/her selections using these interfaces, the GUI stores this information in database tables. The queries executed in the RUN instructions can then use this information to restrict the results generated for the output. However, these database tables will be continually overwritten as the user views different outputs. In order to retain this information so the user knows which simulation data were used in a given output, the GUI runs the SYNC_FromDB instructions to store this information in the Excel file. Therefore, the SYNC_FromDB instructions should contain actions that import to the Excel file any scope data generated by the four interfaces above. Table 4 shows where the data for each interface is stored in the database and where the SYNC_FromDB instructions should store this data in the Excel file.

Table 4: Location of data storage for output scope selection interfaces.

Interface	**Access Database Location**	**Excel Spreadsheet Location**
Task Generator, Task, Activity, ResGroup selection	Output_Selected ResGrps table	SelectedResGrps sheet
Organization selection	Output_SelectedOrgs table	SelectedOrgs sheet
Filter selection	OUTPUT_Filters table	Filter sheet
Snapshot time selection	SelectedOutputTime field in Scenario Information table	Sel Output Time field on FRONT sheet (FRONT!B8)

Note that, for the first three interfaces shown in Table 4, the Excel sheet names where the scope data are stored do not need to be exactly as shown. Any sheet name can be chosen as long as the

SYNC_FromDB and SYNC_ToDB instructions correctly synchronize the sheets with the appropriate database tables using Import and Export actions respectively. For the snapshot time selection, the location in the Excel file is fixed to cell B8 on the FRONT sheet. This was done because the snapshot time is a single value rather than a table which may have arbitrary dimensions. To synchronize the selected snapshot time the Import_time and Export_time actions must be called from the SYNC_FromDB and SYNC_ToDB instructions respectively.

Storing the scope settings of an analysis in the Excel file, along with the instructions that were carried out to produce the output, provides a complete record of what the output data represents. This allows the user to review an output at a later date and to know exactly how the results were created. This also allows the GUI to restore a past analysis, retrieving its scope settings from the Excel file by running SYNC_ToDB. This loads the scope data back into the database and the GUI can then refresh the appropriate interfaces. The user can then alter the scope and repeat the same analysis and compare the results.

The Excel template file contains two other standard sheets called FRONT and CHART. The FRONT sheet contains basic information about the output and three buttons allowing the user to manually execute the SYNC_FromDB, SYNC_ToDB, and RUN instructions. This sheet is primarily used by the MARS V2 GUI to manage the output Excel files. The use of each of the basic information fields is described in Table 5.

Table 5: Output information stored in the FRONT sheet

Field Name	**Example Value**	**Description**
Database File	Scenario.mdb	The file name of the MARS database that contains the data for this analysis.
Database Path	D:\Data\Scenarios	The folder path containing the database file.
Output Name	Resource Utilization vs Time	A name given to the output.
Template Source	OUTPUT_RULvsTIME.xls	The name of the Excel template file that the output is based on. Each new output is created from a certain template file.
Output Description	Plots the overall resource utilization level over time for the selected Organizations	A description of what information the output produces.

Field Name	Example Value	Description
TreeView Index	1	Specifies the interface that the MARS GUI should display to allow the user to define the scope of the analysis. Possible values are: 0 : no interface to be displayed 1 : display Organization selection interface 2 : display Task Generator, Task, Activity, ResGroup selection interface
Filter Index	1	Specifies whether the MARS GUI should display an interface that allows the user to filter simulation objects based on their attributes. This allows the user to further refine the data that is included in the analysis. Filters can be applied to Organizations, Slots, and Resources. Possible values are: 0 : no interface to be displayed 1 : display filter selection interface
Sel Output Time	365	For snapshot outputs, specifies whether the MARS GUI should display an output time selection interface. The value also determines the simulation time that the output should be based on. Any integer value, X, may be entered in this field with the following meaning: $X \geq 0$: display time selection interface, X specifies the time (in days since simulation start) that the output should be based on $X = -1$: display time selection interface, indicates that the output should be based on the simulation initial conditions $X < -1$: no time selection interface to be displayed

The CHART sheet is the standard location where the final graphical output is displayed. However, this is a convention, not a requirement. The output can be displayed in any sheet, and additional outputs may be displayed in different sheets in the same Excel file. Because the whole Excel window is displayed when a given output is selected from the MARS V2 GUI, the user can switch between sheets to view the different outputs.

Beyond the five standard sheets described above, the analyst can create any number of additional sheets to support the analysis. In most cases, additional sheets are required to store the scope information of the analysis as shown in Table 4. The synchronization instructions must then be programmed to keep these sheets synchronized with the corresponding tables in the database. Additional sheets are also often required to perform calculations and to prepare the data so it can be plotted in a chart or summarized in a pivot table.

4.4 Running MARS V2 outputs from the GUI

To include a new output in the MARS V2 GUI, the Excel file for the output is simply placed in the outputs folder in the main MARS directory on the analyst's PC. The analyst must also ensure that any Access query objects and synchronization tables that the output uses are present in the scenario database file being analyzed. Compatibility issues should also be considered because some outputs may make use of simulation data or queries that are only present in more recent versions of the tool.

Once an output has been included in MARS, the user can select it from the list of available Excel output files in the MARS V2 GUI. Depending on the TreeView Index, Filter Index and Sel Output time fields on the FRONT sheet of the chosen Excel file, the GUI will display the appropriate interfaces that allow the user to narrow the scope of the analysis.

If the TreeView Index for the selected output is 0, no interface is required to select specific simulation objects to be included in the analysis. If the TreeView Index is 1, a tree interface is displayed that allows the user to select specific organizations to be included in the analysis. Each organization can be expanded to view its sub-organizations. A checkbox next to each organization's name allows the user to select it. The ID of each Organization that the user selects is stored in a table in the MARS database called Output_SelectedOrgs. An Import action must be added to SYNC_FromDB to save the Output_SelectedOrgs table to a specified sheet. Likewise, an Export action must be added to SYNC_ToDB in order for the MARS V2 GUI to be able to retrieve the selected Organizations from the Excel file.

If the TreeView Index is 2, a tree interface is displayed that allows the user to select specific Task Generators, Tasks, Activities and ResGrps. Task Generators can be expanded to view their Tasks, Tasks can be expanded to view their Activities, and Activities can be expanded to view their ResGrps. A checkbox next to each item allows the user to select it. The user's selections are stored in a table called Output_SelectedResGrps. This table has five columns: Option, TGID, RotoSeq, TskActIndx and ResGrpID. The Option field specifies whether the row identifies a Task Generator, Task, Activity or ResGrp with the values 1, 2, 3 or 4 respectively. The latter four fields store the Task Generator ID, Rotation sequence number which identifies the specific Task, the Activity Index within the Task, and the ResGrp ID as needed to identify the user's selections. Import and Export instructions must be added to the Excel file to synchronize the Output_SelectedResGrps table with a specified sheet.

If the Filter Index for the selected output is 0, no filter interface is required for the analysis. If the Filter Index is 1, an interface is provided that allows the user to define a series of filters to refine which simulation objects are included in the analysis. Filters can be applied to Organizations, Slots and Resources based on their attributes. Typically, the filter interface is only required in conjunction with the Organization selection interface. In this case, the filter would be applied to the user-selected Organizations, the Slots within those Organizations, and the Resources occupying those Slots. To create a filter, the user specifies allowable values for given Attributes. This information is stored in a database table called OUTPUT_Filters. Each filter record in the table has a FilterLevel field that specifies whether the filter applies to an Organization, a Slot, or a Resource using the values 1, 2 or 3 respectively. A fourth filter level is used in some outputs that also plot data using a subset of the filtered set. For example, an output could plot the number of Resources with a certain rank in a selected Organization over time, and it could also plot the subset of those Resources that are deployed. Import and Export actions must be added to the Excel file to synchronize the OUTPUT_Filters table with a specified sheet.

If the Sel Output Time for the selected output is less than -1, no snapshot time selection interface is required for the analysis. If the Sel Output Time is greater than or equal to -1, an interface is displayed that allows the user to select a snapshot time that the output should focus on. This interface allows the user to set the snapshot time to the simulation initial condition, the simulation end state, or an arbitrary time between these two states. If the user selects the simulation initial condition, a value of -1 is stored in the SelectedOutputTime field of the Scenario Information table in the database. If the user selects the simulation end state, the simulation duration in days is stored in the database. If the user selects an arbitrary time between simulation start and end, this time in days since simulation start is stored in the database. Note that the value -1 rather than 0 was chosen to correspond to the simulation initial condition because actions may occur in the simulation at time 0. In general, the snapshot time, X, corresponds to the state of the simulation just after the last event occurring at time X. The Import_time and Export_time actions must be added to SYNC_FromDB and SYNC_ToDB respectively to synchronize the selected output time between the database and the Excel file.

When the user has finished entering his/her selections in these interfaces, he/she can choose to run the analysis to produce the output. The MARS V2 GUI then tells the Excel file to execute the SYNC_FromDB instructions to save the user’s selections to the specified sheets in the Excel file. The GUI then tells the Excel file to execute the RUN instructions that will run the necessary queries to process and retrieve the output data from the database, storing the results to specified sheets in the Excel file.

When the Excel file is asked to execute the RUN instructions, it performs some additional steps before beginning the list of actions on the RUN sheet. The first step is to execute the SYNC_ToDB instructions. While this seems like a wasted step because in many cases the GUI just ran SYNC_FromDB, it is necessary because the user is able to make scope selection changes directly in the Excel file and these changes must be copied back to the database in order to be used by the RUN instructions. The second step is to call a function called update_current_data(). This function checks the Sel Output Time field on the FRONT sheet to determine if the output relies on snapshot data at a particular time during the run. If the output requires snapshot data, the function executes a sequence of SQL queries that update the current state tables for Organizations, Slots and Resources. These tables are SO_OrgAttrCurrent, SO_SlotAttrCurrent and SO_ResAttrCurrent. The update searches in various tables that record attribute changes to Organizations, Slots and Resources over the course of the simulation and determines what their states were at the time selected by the user. This information is stored in the current state tables which can then be used in the RUN instructions to produce the output. Once this step is complete, the RUN instructions are executed which processes and then retrieves the output data from the database, storing it in sheets in the Excel file.

The last step in generating the output is to ensure that Excel detects the new data correctly before it redraws the chart, pivot table, or other graphical display. When the interpreter finishes executing the RUN instructions, it calls an Excel VBA function called chart_update(). This function normally performs two tasks. First, it readjusts the data ranges used by charts to capture the updated data, and second, it activates the sheet that contains the final output, usually the CHART sheet. Depending on the output, the chart_update() function may also customize the appearance of the chart, for example, to set the chart's title or to adjust its axes. These tasks are typically accomplished in just a few lines of code, but this code must be custom-written for each output by the analyst. Typically, the easiest way to generate this code is to start the Excel macro recorder and perform the actions by hand.

Once, the RUN instructions and chart_update() function complete, the user will see the output in the Excel window. The user can further customize the output using any of the standard Excel functionality. The user can also readjust the scope of the analysis using the GUI interfaces or by directly editing the data in Excel and then rerunning the output. The MARS V2 GUI provides the option to save the output which stores a copy of the Excel file with the scenario database that the GUI can retrieve later if the user requests it.

5 MARS output example

This section will describe one MARS output called OUTPUT_OrgTasks in detail. This will familiarize the reader with the process of creating a new MARS output and demonstrate how the features discussed in the previous sections operate in practice.

The OUTPUT_OrgTasks output allows the user to select a group of Organizations and produces an area plot of the Tasks that these Organizations participated in over time. A sample plot is shown in Figure 5. Each coloured region on the plot corresponds to a Task. The height of the region shows the number of Resources that participated in the Task and the width of the region shows the duration of the Task along the horizontal time axis. These regions are stacked so the total height of the coloured regions corresponds to the total number of Resources committed to the Tasks at any time. The grey background region shows the total number of Resources in the selected Organizations, so the analyst can use this output to view the overall level of commitment of selected Organizations relative to their total resource strength. The output also supports filters which can be used to restrict which Resources are included in the counts based on their attributes. Note that a legend is not shown because the number of Tasks that appear depends on the scenario being run and this number could be very large. However, the analyst can mouse-over the coloured regions in Excel to display a pop-up text box that shows the name of the corresponding Task.

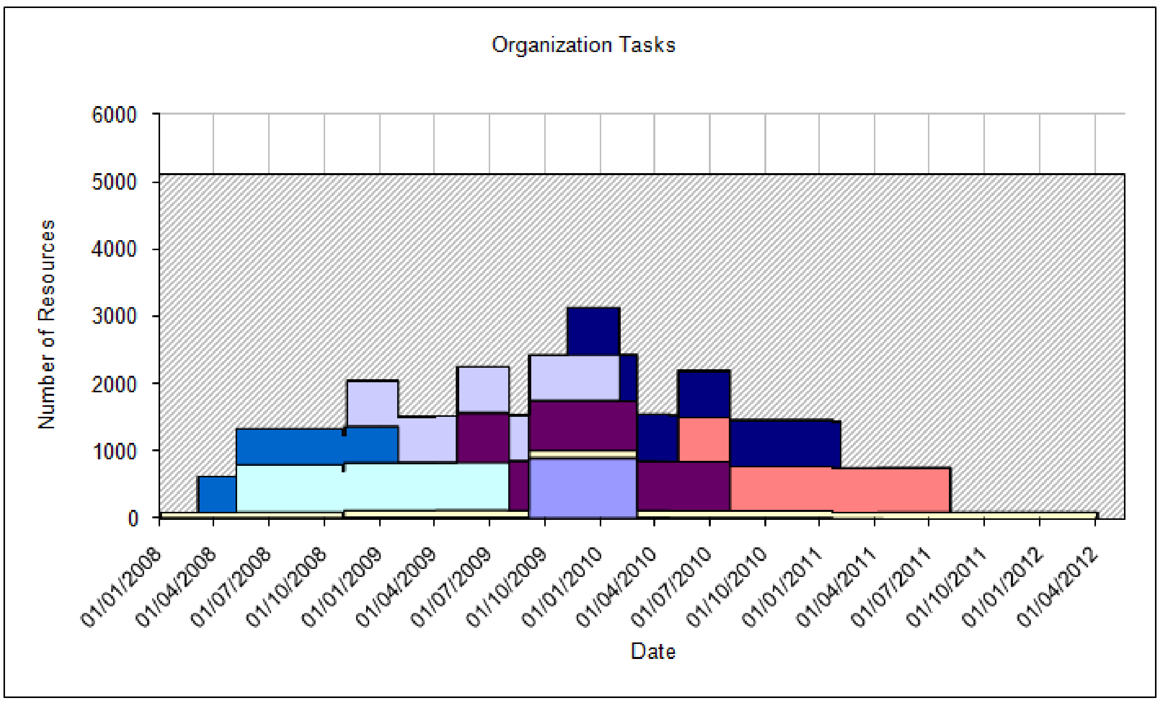


Figure 5: Organization Tasks output showing the number of resources committed to various tasks over time.

The information on the FRONT sheet is shown in Table 6. The important fields to note are, first, the TreeView Index which is set to 1 telling the GUI to display the Organization selection interface, second, the Filter Index which is set to 1 telling the GUI to display the filter interface, and third, Sel Output Time is set to -2 telling the GUI that a snapshot time selection interface is not required. Other fields are self-explanatory and do not affect the behaviour of the output.

Table 6: FRONT sheet information in the Organization Tasks output.

	A	B
1	Database File	Scenario.mdb
2	Database Path	D:\Data\Scenarios
3	Output Name	Organization Task Activities
4	Template Source	OUTPUT_OrgTasks.xls
5	Output Description	Plots the number of resources in selected Organizations committed to Tasks over time.
6	TreeView Index	1
7	Filter Index	1
8	Sel Output Time	-2

Because the output makes use of the Organization selection interface and the filter selection interface, the Excel file has to synchronize with the Output_SelectedOrgs and OUTPUT_Filters tables in the database that hold the information about the user's selections. To store this information in the Excel file, there are two corresponding sheets named SelectedOrgs and Filter. The SYNC_FromDB instructions (shown in Table 7) and SYNC_ToDB instructions (shown in Table 8) contain the import and export instructions that keep the two database tables and the two Excel sheets synchronized.

Table 7: SYNC_FromDB sheet in the Organization Tasks output.

	A	B	C
1	**Action**	**Parameter**	**Value**
2	Database		
3		database name	Scenario.mdb
4		database path	D:\Data\Scenario
5	Import		
6		sheet name	SelectedOrgs
7		table name	Output_SelectedOrgs
8	Import		
9		sheet name	Filter
10		table name	OUTPUT_Filters

Table 8: SYNC_ToDB sheet in the Organization Tasks output.

	A	B	C
1	**Action**	**Parameter**	**Value**
2	Database		
3		database name	Scenario.mdb
4		database path	D:\Data\Scenario
5	Export		
6		sheet name	SelectedOrgs
7		table name	Output_SelectedOrgs
8	Export		
9		sheet name	Filter
10		table name	OUTPUT_Filters

Note that for all sheets that contain instructions, the first action is always the Database instruction. The two parameters in the Database instruction are the database name and path. The values provided for these two parameters are set to reference the database name and path information on the FRONT sheet ("FRONT!B1" for the database name and "FRONT!B2" for the database path). This means the database name and path only have to be set once on the FRONT sheet and then each of the three instruction sheets references these values when they open a connection to the database.

In order to generate the output data that will ultimately be plotted in Excel, several queries were created to process the simulation data in the scenario database. These queries were built using the Access query designer and saved in the database. Table 9 lists all of the queries involved in the analysis and a description of what each query does.

Table 9: Access query objects used by the Organization Tasks output.

Query Name	Description
OUTPUT_OrgTree	Takes the list of user-selected Organizations in the Output_SelectedOrgs table and retrieves the full list of Organizations which includes the user-selected Organizations and all their child Organizations.
OUTPUT_OrgTaskActivities1	Retrieves the list of Resources that belong to the Organizations found in the OUTPUT_OrgTree query.
OUTPUT_OrgTaskActivities1_FILTER	Determines which of the Resources found in the OUTPUT_OrgTaskActivities1 query should be removed because they do not match the user-selected filter criteria in the OUTPUT_Filters table.
OUTPUT_OrgTaskActivities1_FINAL	Removes the resources found in the OUTPUT_OrgTaskActivities1_FILTER query from the full list of Resources found in the OUTPUT_OrgTaskActivities1 query.
OUTPUT_OrgTaskActivities1_COUNT	Counts the number of Resources found in the OUTPUT_OrgTaskActivities1_FINAL query.
OUTPUT_OrgTaskActivities2	Takes the Resources found in OUTPUT_OrgTaskActivities1_FINAL and determines which ResGroups they were a part of, counts the number of Resources per ResGroup, and determines the periods for which those ResGroups were occupied by Activities.
OUTPUT_OrgTask3	Organizes the data found in OUTPUT_OrgTaskActivities2 by Task and calculates the number of Resources that entered (positive integer) or exited (negative integer) the Task over time.
OUTPUT_OrgTask4	Crosstab query that summarizes the data from OUTPUT_OrgTask3 with Tasks in the columns, dates in the rows, and values showing the number of Resources that entered or exited a given Task at a given time.

The OUTPUT_OrgTask4 query is the final query that generates the results that are then used in the Excel file to create the plot. The other queries that OUTPUT_OrgTask4 relies on are executed internally by Access. In addition to this data, the output needs the simulation start date and duration, and it needs the total number of Resources in the selected Organizations that meet the selected filter criteria which is the result of the OUTPUT_OrgTaskActivities1_COUNT query. Table 10 shows the RUN instructions that perform these actions.

Table 10: RUN sheet in the Organization Tasks output.

	A	B	C
1	**Action**	**Parameter**	**Value**
2	Database		
3		database name	Scenario.mdb
4		database path	D:\Data\Scenario
5	Obj_query		
6		obj name	OUTPUT_OrgTask4
7		sheet name	OUTPUT
8	SQL_query		
9		sql statement	SELECT [SimStartDate], [RunEndTime] FROM [Scenario Information]
10		sheet name	OUTPUT2
11	Obj_query		
12		obj name	OUTPUT_OrgTaskActivities1_COUNT
13		sheet name	OUTPUT3

The three query actions in the RUN instructions store their results to sheets OUTPUT, OUTPUT2 and OUTPUT3 respectively. Once these actions are executed, the OUTPUT sheet will contain the data on the numbers of Resources that participated in Tasks over time. The OUTPUT2 sheet will contain the result of the SQL query that retrieves the simulation start date and duration from the Scenario Information table in the database. And the OUTPUT3 sheet will contain the count of the total number of resources in the selected Organizations that meet the selected filter criteria.

Once these data are in the Excel file, several calculation sheets will refresh themselves using the new data in order to format it for plotting. The CALCS and CALCS2 sheets reorganize the Resource usage data from the OUTPUT sheet using the simulation start date and duration from OUTPUT2 and adding the total Resource count from OUTPUT3 so that the information can be plotted using an area plot along a time axis. The CALCS3 sheet counts the number of rows and columns returned by the query in the OUTPUT sheet, and it copies the total Resource count from the OUTPUT3 sheet.

The chart which appears in the CHART sheet is based on the numbers in the CALCS2 sheet. However, when the output runs, the number of rows of data that need to be plotted is unknown. Therefore, the data range for the chart has to be adjusted after each new run. This adjustment is programmed into the chart_update() function shown in Table 11. This function calculates how many rows of data from the CALCS2 sheet the chart should be using based on the row count in the CALCS3 sheet. It then updates the chart's data range based on the required number of rows. After doing this, the function activates the CHART sheet so the chart is displayed to the user as soon as all the processing completes.

Table 11: *The chart update function in the Organization Tasks output.*

```
Sub chart_update()

    Dim LastRow As Integer
    Dim LastCol As Integer
    Dim DataRange As String

    'calculate last row and last col of data that should be plotted
    LastRow = 2 + 2 * Sheets("CALCS3").Cells(1, 2) + 2
    LastCol = 204

    'create data range for chart
    DataRange = "C2:" & ColumnLetter(LastCol) & LastRow

    'activate the chart and set the new data range
    Sheets("CHART").Activate
    ActiveSheet.ChartObjects("Chart 13").Activate
    ActiveChart.SetSourceData Source:=Sheets("CALCS2").Range(DataRange),
                              PlotBy:=xlColumns

End Sub
```

This completes the steps required to automate the generation of this output. By placing the output excel file in the output folder of the MARS installation directory, the new output will become available within the MARS application.

6 MARS V2 standard outputs

The current version of MARS includes many standard outputs that were built using the method described in the previous section. These outputs can be used to investigate many aspects of a MARS simulation run. However, it is expected that this list will be added to over time as the intent of the MARS V2 output system was not to provide a complete set of outputs but to provide a system that makes creating new MARS outputs as easy as possible. Table 12 contains the current list of 18 standard MARS outputs including a description of the output, the name of the excel file, and a sample of the chart or table that the output produces.

Table 12: List of built-in MARS V2 outputs.

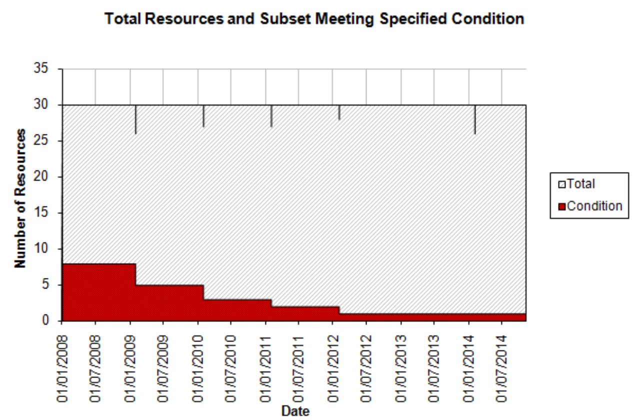
<table>
<tr><th>Output</th><th>Sample</th></tr>
<tr><td>

Title: Organization Resources Filter

Description: For selected organizations, displays the number of resources that meet the specified filter conditions over time. Gray hashed area shows the total number of resources in the organizations over time. Red area shows the number of those resources that meet the filter conditions.

File: OUTPUT_OrgFilterCondition

</td><td>



</td></tr>
<tr><td>

Title: Organization Resources

Description: For selected organizations, displays the breakdown of resources by occupation and rank.

File: OUTPUT_OrgRes

</td><td>

<table>
<tr><td>Sum of CountOfResID</td><td>Rank</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>MOSID</td><td>Unsp</td><td>Civ</td><td>PteR</td><td>PteB</td><td>PteT</td><td>Cpl</td><td>MCpl</td></tr>
<tr><td>AMMO TECH [169]</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>ARMD (178)</td><td></td><td>11</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>ARTY (179)</td><td></td><td></td><td></td><td></td><td>32</td><td></td><td></td></tr>
<tr><td>ARTYMN FD [8]</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>CBT ENGR [339]</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>CHAP (349)</td><td></td><td></td><td></td><td></td><td></td><td>6</td><td></td></tr>
<tr><td>CONST TECH [306]</td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td></tr>
<tr><td>COOK [164]</td><td></td><td></td><td></td><td>47</td><td></td><td></td><td></td></tr>
<tr><td>CRMN [5]</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>D MECH [225]</td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td></tr>
</table>

</td></tr>
<tr><td>

Title: Organization Resources Used

Description: For selected organizations, displays the number of resources that were used and unused broken down by occupation and rank.

File: OUTPUT_OrgRes_Duration

</td><td>

<table>
<tr><td>Count of ResID</td><td></td><td>Rank</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>MOSID</td><td>Used</td><td>Capt</td><td>Cpl</td><td>CWO</td><td>Lt</td><td>MCpl</td><td>MWO</td><td>PteR</td><td>PteT</td><td>Sgt</td><td>WO</td><td>Total</td></tr>
<tr><td>INF (180)</td><td>0</td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td></tr>
<tr><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>INF (180) Total</td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td></tr>
<tr><td>INFMN [10]</td><td>0</td><td>2</td><td>69</td><td>2</td><td>2</td><td>41</td><td>8</td><td></td><td>58</td><td>25</td><td>14</td><td>221</td></tr>
<tr><td></td><td>1</td><td></td><td>7</td><td></td><td></td><td>1</td><td></td><td>25</td><td>2</td><td>3</td><td>2</td><td>40</td></tr>
<tr><td>INFMN [10] Total</td><td></td><td>2</td><td>76</td><td>2</td><td>2</td><td>42</td><td>8</td><td>25</td><td>60</td><td>28</td><td>16</td><td>261</td></tr>
<tr><td>Total</td><td></td><td>3</td><td>76</td><td>2</td><td>2</td><td>42</td><td>8</td><td>25</td><td>60</td><td>28</td><td>16</td><td>262</td></tr>
</table>

</td></tr>
</table>

Title Organization Resources Unused

Description: For selected organizations, displays the number of resources that were unused broken down by occupation and rank.

File: OUTPUT_OrgRes_Unused

Count of ResID		Rank							
MOSID	Used	PteT	Cpl	MCpl	Sgt	WO	MWO	CWO	Total
⊟INF (180)	0	4	4	1	1	1	1	1	13
⊟INFMN [10]	0	6	6	3	3	3	3	3	27
Total		10	10	4	4	4	4	4	40

Title: Organization Resources Used RUL

Description: For selected organizations, displays the number of resources that were used and unused on Activities with a Resources Utilization Level of 1.

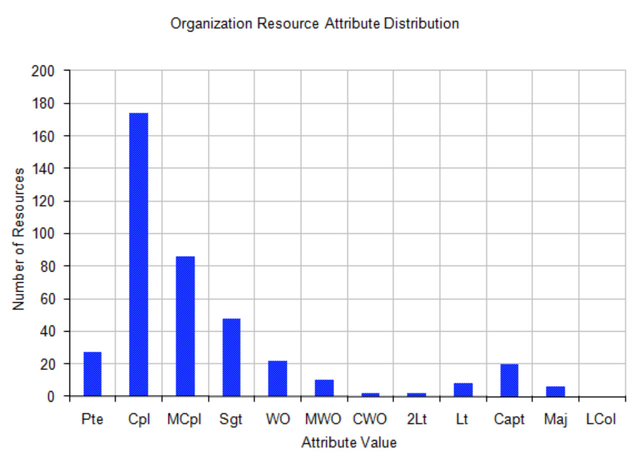
File: OUTPUT_OrgRes_Usage_RUL1

Count of ResID		Rank							
MOSID	Used	PteT	Cpl	MCpl	Sgt	WO	MWO	CWO	Grand Total
⊟INFMN [10]	0	10	10	4	4	4	4	4	40
	1								
INFMN [10] Total		10	10	4	4	4	4	4	40
Grand Total		10	10	4	4	4	4	4	40

Title: Organization Resources Attribute Distribution

Description: For selected organizations, displays the number of resources having each possible value of a chosen attribute. The attribute id is specified in the RUN sheet.

File: OUTPUT_OrgResAttrDist



Title: Organization Slots

Description: For selected organizations, displays the number of slots broken down by occupation requirement and rank requirement.

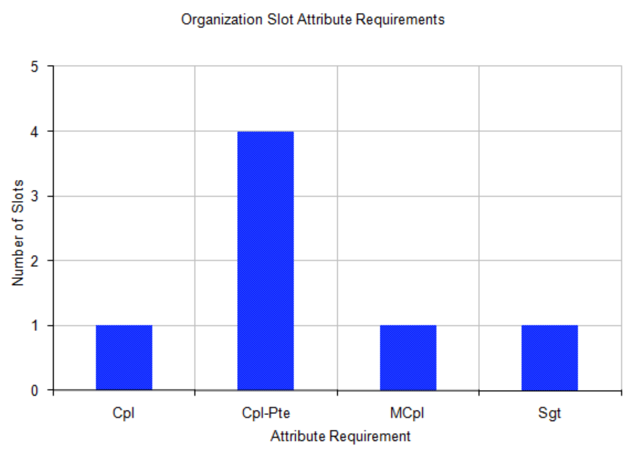
File: OUTPUT_OrgSlot

Count of SlotID	Rank Req							
MOSID Req	PteT	Cpl	MCpl	Sgt	WO	MWO	CWO	Total
INFMN [10]	1	1	1	1	1	1	1	7
Total	1	1	1	1	1	1	1	7

Title: Organization Slot Attribute Distribution

Description: For selected organizations, displays the number of slots having each possible value of a chosen attribute requirement. The attribute id is specified in the RUN sheet.

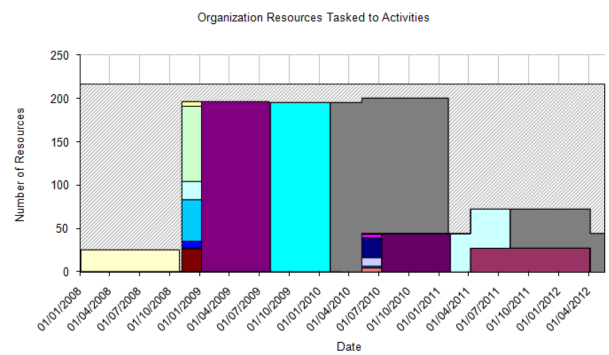
File: OUTPUT_OrgSlotAttrDist



Title: Organization Task Activities

Description: For selected organizations, displays the number of resources committed to activities over time. Filter can be applied to restrict the resources counted to those having specified attributes. Gray hashed area shows total number of resources available in the organization. Coloured areas show resources committed to activities.

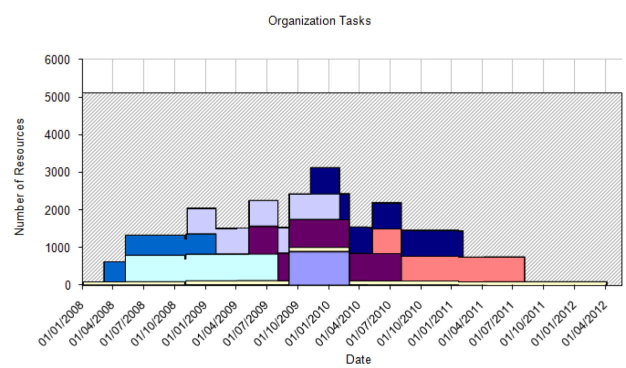
File: OUTPUT_OrgTaskActivities_V2



Title: Organization Tasks

Description: For selected organizations, displays the number of resources committed to tasks over time. A filter can be applied to restrict the resources counted to those having specified attributes. Gray hashed area shows total number of resources available in the organization. Coloured areas show resources committed to tasks.

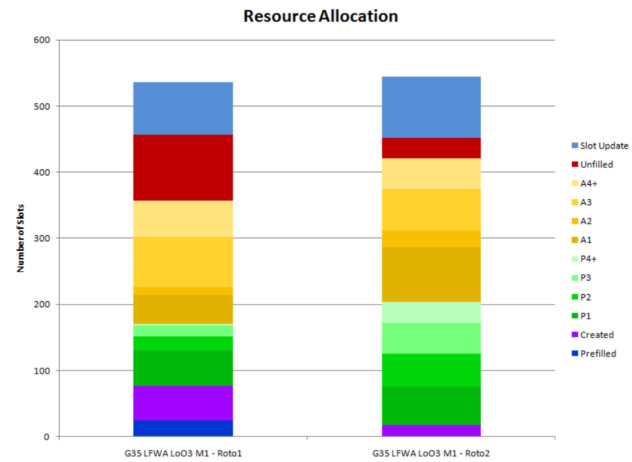
File: OUTPUT_OrgTasks



Title: ResGrp Resource Sourcing

Description: For selected ResGrps, displays the number of slots filled from various sources (prefilled, created, primary + level and augmentee + level). Unfilled slots and slots selected for slot attribute updating are also displayed. A simplified version of the chart is also included that hides the distinction between levels in the primary and augmentee sources.

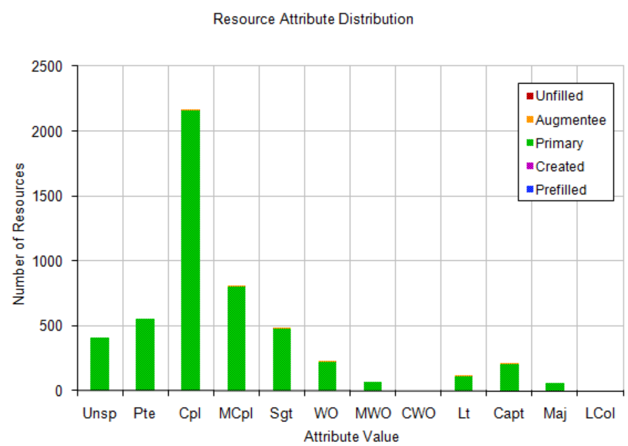
File: OUTPUT_ResAllocLevels



Title: ResGrp Resource Attribute Distribution

Description: For selected ResGrps, displays the number of selected resources categorized by source (prefilled, created, primary and augmentee) broken down by a chosen attribute. The attribute ID is specified in the RUN sheet.

File: OUTPUT_ResAttrDist



Title: ResGrp Resources

Description: For selected ResGrps, displays the number of assigned Resources broken down by occupation and rank.

File: OUTPUT_ResGrpResources

Count of ResID	Rank								
MOSID	PteR	PteB	PteT	Cpl	MCpl	Sgt	WO	MWO	CWO
ARMD (178)									
ARTYMN FD [8]	1	1			1	1			1
CBT ENGR [339]							1		
COOK [164]			1						
CRMN [5]									
INF (180)									
INFMN [10]									
INT [213]								1	
INT OP [99]									
LCIS TECH [110]									
LOG [328]				1					
SIG OP [329]									
Total	1	1	1	1	1	1	1	1	1

Title: ResGrp Source Organizations

Description: For each ResGrp shows which Organizations supplied resources and the number of resources supplied.

File: OUTPUT_ResGrpSrcOrgs

Time	Name	ResSourceName	SeqLevel	OrgName	Total
222	zxcv	primary	1	LFWA	3
			2	PPCLI	2
		augmentee	1	orgA	3
				orgB	1
555	qwer	primary	1	orgC	3
				orgD	2
			2	orgE	3
				orgF	6
		augmentee	1	orgG	5
				orgH	6
			2	orgJ	7

Title: Simulation Summary

Description: For each Task run in a scenario, displays the number of activities completed out of the total number of planned activities.

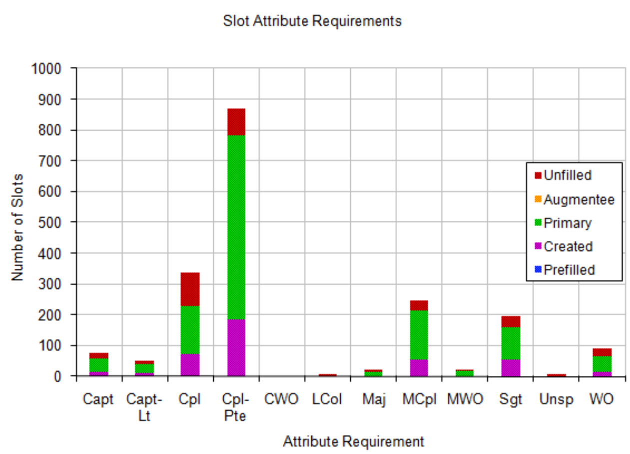
File: OUTPUT_SimSummary

TGID	TaskName	RotoSeq	Activities Completed
21	TF 3-09	1	3/6
22	TF 1-10	1	5/6
23	TF 2-10	1	6/8

Title: ResGrp Slot Requirements Distribution

Description: For selected ResGrps, shows the number of slots filled from various sources (prefilled, created, primary, augmentee and unfilled) broken down by a chosen ResGrp slot requirement attribute.

File: OUTPUT_SlotAttrDist



Title: ResGrp Slots

Description: For selected ResGrps, displays the number of slots broken down by required occupation and required rank.

File: OUTPUT_TaskSlots

Count of SlotID	Rank							
MOSID	Capt	Capt-Lt	Cpl	Cpl-PteT	Lt	MCpl	Sgt	WO
ARMD [178]	10	4			13			
ARTY [179]	12	16						
ARTYMAN FD [8]			44	132		42	33	20
CRMN [5]			12	250		28	26	17
FCS TECH [327]			2	6		5		
GEO TECH [238]							1	
INT OP [99]							1	

Title: ResGrp Slots Filled

Description: For selected ResGrps, displays the number of slots that were filled and not filled broken down by required occupation and required rank. Numbers of unfilled slots are shown in red.

File: OUTPUT_TaskSlotsSummary

Count of SlotID		Rank						
MOSID	Filled	Capt	Capt-Lt	Cpl	Cpl-PteT	WO	WO-Sgt	Grand Total
ARMD [178]	0							3
	1	10	4					33
ARMD [178] Total		10	4					36
ARTY [179]	0		10					10
	1	12	16					30
ARTY [179] Total		12	26					40
ARTYMAN FD [8]	0				130	8		183
	1			44	132	20		273
ARTYMAN FD [8] Total				44	262	28		456
CBT A [90027]	0							
	1	20						24
CBT A [90027] Total		20						24
CBT ENGR [339]	0							
	1			98	218	12		442
CBT ENGR [339] Total				98	218	12		442

7 Conclusion

This paper described the new graphical output system implemented in MARS V2. The primary objective of the new system was to allow the analyst to create customized outputs using standard analytical tools and to incorporate the output into the MARS V2 application. This is an improvement over the previous MARS output system which was implemented in Visual Basic as part of the GUI and was, therefore, not directly accessible to the analyst for customization or development of new outputs.

The requirement for an analyst-driven system for developing new MARS outputs derived from the dynamic needs of military clients. New ways of looking at simulation data to generate useful insights routinely emerge as no two problems are alike. The new output system allows the analyst to design custom outputs using MS Access and Excel and then to automate the process of generating the output using a special set of instructions entered in Excel sheets. The automated output can then be easily incorporated into the MARS application where it can be called up whenever it is need.

The operation of the various components of this new output system was described in detail. The steps include narrowing the scope of the simulation data and preparing and extracting the relevant information using Access queries. Then Excel is used to further process the data and display the resulting graphical output in the form of a chart or table. The capabilities of the interpreter that executes the automation instructions were also fully documented. Finally, an example was provided that described the complete process of creating a new output using this system.

The set of 18 MARS outputs existing as of the time of publishing this paper was described, but because of the changing needs of military clients and the capability to easily modify and create new outputs for MARS, it is expected that this set of outputs will expand over time. It is this ease of creating customized output that provides value to clients as it enables analysts to respond more rapidly than before to emerging client requirements.

References

[1] Ormrod, M., Young, C., and Pall R. (2007). Modelling Force Generation with the Managed Readiness Simulator (MARS): Modelling Concept and Requirements for MARS v1.0. DRDC CORA Technical Memorandum TM 2007-65.

[2] Kelton, D.W., Sadowski, R. P., Sturrock, D. R. (2004) Simulation with Arena. 3rd ed. Boston: McGraw-Hill Higher Education, 2004.

[3] Pall, R., Young, C., and Ormrod, M. (2007). Modelling Force Generation with the Managed Readiness Simulator (MARS): Implementation of MARS v1.0 in a Discrete Event Simulation Environment. DRDC CORA Technical Memorandum TM 2007-52.

[4] Young, C., Pall, R., and Ormrod, M. (2007). A Framework & Prototype for Modelling Army Force Generation. DRDC CORA Technical Memorandum TM 2007-54.

[5] Ormrod, M., Young, C. (2007). Preliminary Analysis of Task Force Afghanistan Sustainability Using MARS. DRDC CORA Technical Memorandum TM 2007-40.

[6] Okazawa, S., Ormrod, M., Young, C. (2009). Managed Readiness Simulator (MARS) V2: Assessment of a Simulation Runtime Database Approach. DRDC CORA Technical Memorandum TM 2009-043.

[7] Okazawa, S., Ormrod, M., Young, C. (2009). Managed Readiness Simulator (MARS) V2: Design of the Managed Readiness Model. DRDC CORA Technical Memorandum TM 2009-057.

[8] Okazawa, S., Ormrod, M., Young,C. (2010). Managed Readiness Simulator (MARS) V2: Implementation of the Managed Readiness Model. DRDC CORA Technical Memorandum TM 2010-261.

List of symbols/abbreviations/acronyms/initialisms

CF	Canadian Forces
CORA	Centre for Operational Research and Analysis
DND	Department of National Defence
DRDC	Defence Research & Development Canada
GUI	Graphical User Interface
LFORT	Land Force Operational Research Team
MARS	Managed Readiness Simulator
MS	Microsoft
PC	Personal Computer
ResGrp	Resource group
SQL	Structured Query Language
SRDB	Simulation Runtime Database
V1	Version 1
V2	Version 2
VB	Visual Basic
VBA	Visual Basic for Applications

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.) Defence R&D Canada – CORA 101 Colonel By Drive Ottawa, Ontario K1A 0K2	2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.) (NON-CONTROLLED GOODS) UNCLASSIFIED DMC A REVIEW GCEC June 2010

3\. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

Managed Readiness Simulator (MARS) V2: The Graphical Simulation Output System

4\. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used)

Okazawa, S.; Ormrod, M.

5. DATE OF PUBLICATION (Month and year of publication of document.) February 2012	6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.) 48	6b. NO. OF REFS (Total cited in document.) 8

7\. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Memorandum

8\. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – CORA
101 Colonel By Drive
Ottawa, Ontario K1A 0K2

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) DRDC CORA TM 2012-038	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11\. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

Unlimited

12\. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

The Managed Readiness Simulator (MARS) is a versatile program that allows the user to quickly simulate a wide range of Canadian Forces readiness scenarios to determine if the Resources of an Establishment are able to satisfy the requirements of a set of operational Tasks. The first version of MARS (V1) demonstrated the managed readiness modelling concept and was successfully applied to a preliminary analysis of the Army's plans to generate the forces for Task Force Afghanistan. A Graphical User Interface (GUI), developed under software development contracts, integrated scenario setup, execution and output analysis tools into a single package written in Visual Basic. Following a recent major redesign, MARS V2 now incorporates an advanced software architecture that integrates database technology into simulation execution and a more flexible managed readiness model that includes support for Establishment dynamics. MARS V2 also implemented a new system for generating graphical simulation output. In practice, the need to customize existing outputs or create new outputs occurs frequently, and the expected turn-around time for these changes is short. The new MARS V2 output system allows the analyst to quickly mock up new simulation outputs using standard tools (MS Access and Excel) and then to automate the generation of those outputs so that they can be included in the standard set of outputs available in the MARS application. This paper documents the implementation of the new MARS V2 output system.

Le programme de simulation de gestion de la disponibilité opérationnelle (programme MARS) est un programme polyvalent qui permet à l'utilisateur de rapidement simuler une vaste gamme de scénarios de disponibilité opérationnelle des Forces canadiennes afin de déterminer si les ressources d'un établissement peuvent répondre aux besoins propres à un ensemble de tâches opérationnelles. La première version du programme MARS (V1) a démontré le concept du modèle de gestion de la disponibilité opérationnelle. Elle a été utilisée avec succès lors d'une analyse préliminaire des plans de l'Armée visant à mettre sur pied les forces pour constituer la Force opérationnelle Afghanistan. Une interface graphique (GUI), développée dans le cadre de contrats de développement de logiciel, a permis d'intégrer la mise en place et l'exécution d'un scénario ainsi que des outils d'analyse de données dans un seul ensemble rédigé en Visual Basic. Après une importante restructuration, le programme MARS V2 incorpore désormais une architecture logicielle plus sophistiquée qui intègre une technologie de traitement de bases de données dans l'exécution de la simulation, ainsi qu'un modèle de gestion de la disponibilité opérationnelle plus souple qui comprend du soutien au niveau de la dynamique de l'établissement. Un nouveau système a également été mis en place pour générer des résultats de simulation graphique. En pratique, il arrive fréquemment qu'il faille modifier des données existantes ou en créer d'autres. Les délais pour effectuer ces changements sont alors très courts. Le nouveau système de données du MARS V2 permet à l'analyste de faire un échantillonnage rapide des nouvelles données de simulation à l'aide des outils standard (MS Access et Excel) et d'automatiser la production de ces données afin qu'elles puissent être ajoutées à l'ensemble des données normalisé dans le programme MARS. Le présent document porte sur la mise en œuvre du nouveau système de données de MARS V2.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Managed Readiness Simulator; MARS; MARS V2; Simulation; Readiness; Arena; Access; Excel; Discrete Event Simulation; Analysis; Output