An Executable Architecture Tool for the Modeling and Simulation of Operational Process Models

Natalie M. Nakhla
Member, IEEE
Canadian Forces Warfare Center
Center for Operational Research and Analysis
Defence Research and Development Canada
Department of National Defence
E-mail: Natalie.Nakhla@forces.gc.ca

Kendall Wheaton
Canadian Forces Warfare Center
Center for Operational Research and Analysis
Defence Research and Development Canada
Department of National Defence
E-mail:Kendall.Wheaton@forces.gc.ca

Abstract—This paper presents an executable architecture tool for the modeling and simulation of operational process models using MATLAB. The new capability provides the means to validate and refine processes, perform dynamic behavioural analysis and to identify bottlenecks and gaps in the process. Using ubiquitous software eliminates the need for specialized training and enables multi-national coordination and iterative development. This paper includes a literature review, background information on process models and architecture products, development details of the new executable architecture capability and numerical examples which demonstrate the features of the proposed tool.

I. INTRODUCTION

Architecture products are significantly beneficial in exercises and experiments. They can be used in all phases of events: framing, planning, execution and analysis. Architecture products can be employed to help refine hypotheses, document baseline as-is structures and investigate to-be possibilities. The products can be used to identify people, processes and systems, and map experimentation objectives to an organizations resources and systems [1]. Specifically, operational architecture views (process models or activity diagrams) describe what activities and tasks need to be done in order to accomplish a mission, and what process needs to be followed. They are a means to articulate to the war-fighter what to do and to document standard operating procedures (SOPs).

The Canadian Forces Warfare Centre (CFWC) conducts large joint (multi-service), national and multinational, distributed experiments and exercises. These events are designed to explore concepts, test hypotheses and develop/assess tactics and procedures which in turn will benefit the Canadian Armed Forces (CAF) and the Department of National Defence (DND). There has been a recent push by the CFWC and coalition partners to utilize architecture products in all stages of their complex distributed events, and to assist in the development of SOPs and tactics, techniques and procedures (TTPs).

Although architecture products capture vast amounts of information, they are still static in nature. Static representations limit the ability to conduct dynamic behavioural analysis which is key in identifying causal effects and relationships. As a result, there have been several attempts in the literature to address this issue through the development of executable architectures [2]–[7]. An executable architecture consists of an enterprise architecture description, combined with the appropriate tools such as compilers and translators which allow the simulation and analysis of the architecture structure. Executable architectures can be used to validate and refine processes and provide the capability to perform what-if analysis and to identify gaps and bottlenecks in the process or system design.

Several commercial tools exist which allow executable architectures and process model simulation [8]–[10]. However, these tools are quite specialised and consequently, they are limited in terms of the scope of the application and/or require extensive training. In addition, the procurement process of these tools can be quite difficult due to time delays and high cost. This in turn inhibits the long term development of projects, introduces delays and hampers multi-national collaboration.

To address the above issues, this paper presents a new executable architecture tool for the simulation of operational architecture models. The requirement for this capability stemmed from the need to develop and share process models using tools which are common to all coordinating nations and have the potential for advanced analysis. Thus, the proposed tool uses ubiquitous software: MATLAB [11] and MS Visio, which do not require specialised training. A static process diagram is transformed to a dynamic simulation scenario in the MATLAB SimEvents environment [12], allowing for behavioural and what-if analysis, as well as statistical simulation using Monte Carlo techniques and sensitivity analysis [13]. This makes it possible to simulate as-is architectures combined with what-if scenarios, resulting in the development of to-be models. The use of MATLAB opens the door for sophisticated methods such as models based on neural networks [14]–[16], or genetic algorithms [17] to represent activities in the process flow. Furthermore, since the model is in the MATLAB environment, the results can be analysed using a myriad of techniques. Another feature of the tool is its bi-directional interface between the simulation model and MS Visio, which in turn facilitates coalition coordination and combined iterative development. The proposed capability can be used to test drive different combinations of processes and technologies. This work can be used to build better system of systems, provide better integration of systems and processes, help to improve the planning, execution and analysis phases of joint experimentation and support joint

systems engineering analysis.

The tool proposed in this paper was validated within the CAF/DND environment using a collection of distributed mission threads which were utilized in joint exercises and experiments. The results demonstrated the many advantages of the new tool and the usage of its capabilities is gaining increasing popularity.

The paper is organized as follows. Section II provides a literature review, and Section III discusses the Department of National Defence Architecture Framework (DNDAF) and process models. Section IV describes the development of the proposed capability, with details regarding the individual modules, such as the XML processor and the transformation engine module which converts the static elements to a dynamic simulation scenario. Section V provides examples which demonstrate the capabilities of the new tool, and Section VI discusses conclusions and future work.

II. Literature Review

This work was inspired by earlier projects that used architecture products to assist in the development of simulations of military and other government processes. A decade ago, researchers at the Naval Postgraduate School [6], [7] simulated a new concept from the United States Joint Forces Command (USJFCOM) for a deployed headquarters. They used architecture products based upon the Department of Defense Architecture Framework (DoDAF) and simulated the headquarters using ExtendSim (known as Extend at that time). Security studies for the 2010 Vancouver Olympics [5] used DoDAF and Gensym's G2/ReThink simulation system [8] to model information management processes. Researchers from Defence Research and Development Canada built process models in both COREsim (from Vitech Corporation) [10] and G2/ReThink, based upon the same DoDAF architecture products, in order to examine the advantages and disadvantages of working with these particular simulation systems. Finally a comprehensive study [18]–[21] of a military planning process employed DoDAF and COREsim to complete the research.

These papers are examples of studies that prepared executable architectures in order to study various processes. It is not the purpose of this paper to examine the pros and cons of the particular tools used in the past. This is to show some practical examples of the use of executable architectures for process modeling and establish that this is done with diverse and sometimes quite specialized tools. This paper describes a new approach for preparing executable architectures that is based upon the examples cited above and upon discussions with staff at USJFCOM in 2003 [22] that had devised a method to link DoDAF architecture products directly to G2/ReThink. So far as the authors know, this important work is undocumented. It was, however, an important contribution to concept of a tool that automates the movement of information from an architectural view to process model simulation and back.

III. Architecture Products and Process Models

This section provides a brief overview of DND architecture products, and presents several examples of process models developed in MS Visio which can be used with the proposed tool.

A. DNDAF Products

The Department of National Defense Architecture Framework (DNDAF) prescribes a guided framework on how to develop a complete standardized set of views describing an architecture [1]. The DNDAF is tool-independent in that it does not advocate one notation or methodology over another, it simply defines what information such as data elements and relationships should be present in each view. There are numerous categories of views defined in the DNDAF, some of which are: common views, capability views, operational views, systems views and technical views. Of all the views defined in the DNDAF, operational and system views are particularly relevant to experiments and exercises. The operational views provide a logical description of the activities and information exchanged between components in order to accomplish a set of goals (e.g. Complete a mission). The system views describe the systems and their connections which support the operational concepts and functions (defined in the operational views).

For this work, operational activity models (OV-5b diagrams) and more detailed event trace (OV-6c) diagrams are of significant interest. As mentioned previously, although DNDAF products capture vast amounts of information regarding all aspects of the architecture, they are still static in nature. The work proposed in this paper addresses this issue.

B. Process Models in MS Visio

Figure 1 shows a sample process model developed in MS Visio (OV-5b diagram). All of the models developed for this work contain all of the information prescribed by the DNDAF. *Note that generic examples are presented in this paper due to the sensitivity of information typically used at the DND.*

The relevant objects in the process model are as follows:

- Action states: These are the activities/tasks which represent the operational activities performed in the process

- Swimlanes: These rectangular boxes represent the role/responsibility performing that task. The swimlanes are denoted 'Person A', 'Person B' etc. in the example diagram. Note that the swimlanes do not have to be individuals, they can also be organizations/departments, etc.

- Control Flow Objects: These are the connections between activities/tasks.

- Decision Blocks: The decision blocks are represented by diamond shaped objects and have more than one output path.

- Simulation Parameters: The simulation parameters define the wait and execution times, the branching probabilities, etc., and are embedded in the diagram as shape data. For example, the green bubble callouts under the decision object in Figure 1 represent the branching probabilities for the decision block.

The process starts with an initial state, and flows through the activities until a final state is reached.
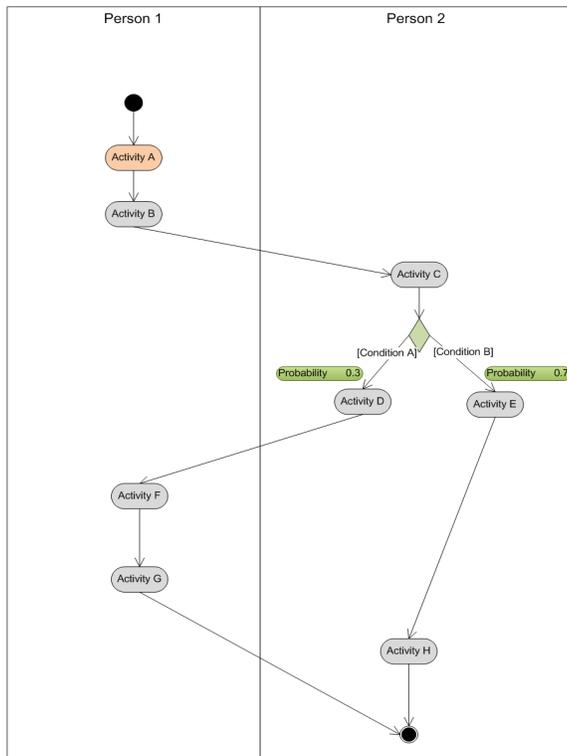
Fig. 1.   Example Process Model in MS Visio

IV.   DEVELOPMENT OF THE PROPOSED CAPABILITY

This section describes the development details of the proposed executable architecture tool. An overview of the tool is presented, along with a description of each of the modules developed.

A. Overview

The proposed tool provides an executable architecture capability using MS Visio and the Matlab discrete event simulation environment SimEvents [12]. Starting with the activity diagram in MS Visio, the algorithm goes through a series of modules which process the MS Visio file and transform the static model to a dynamic simulation scenario. Within SimEvents, the user can modify the model, simulate the process, and analyze the results as often as desired. Subsequently, using the tools bi-directional capability, a new MS Visio model can be automatically generated for sharing with colleagues and coalition partners. The computational flow and modules are presented in Figure 2. Details of each module are provided in the subsequent sections.

B. XML Processor Module

The method starts with the static process model developed in MS Visio as depicted in Figure 1. In order to interface with Matlab, the information from MS Visio must be exported in some readable manner. The proposed tool uses MS Visio's XML export capability. The MS Visio XML file contains all of the shapes in the activity model (even ones that cannot be seen). However, all of this information is in Microsoft proprietary format. Using the Microsoft Visio Object Model

([23]), the developed XML processor module parses the MS Visio XML file and extracts all of the relevant information for setting up the simulation scenario. Several of the key features of this module are provided below.

1) Consistency Check: Once the relevant information is extracted in Matlab, the tool performs a consistency check on the data to ensure there are no logical mistakes in the diagram. This involves checking for missing connections/floating states and initial/final states. If any errors occur, the user is prompted to modify the model and try again. Otherwise, if the diagram check passes, the tool progresses to the next step.

2) Simplified XML File: Due to its proprietary format, exporting the raw output from MS Visio to other customized tools (such as in-house data collection and analysis tools in use at the CFWC), is not achievable without significant effort. To address this, this module also generates a simplified XML file which contains only the relevant information required to generate a simulation model. This file contains all the activities, connections, and any transitions (such as merges and forks). It also includes detailed parameters such as timing and measurement metrics. In addition, for a typical one-page process model, the raw MS Visio XML file is usually approximately 1MB in size, whereas the size of the corresponding simplified XML file is in the range of 5-10 KB.

3) Sub-Processes (Nested Processes): Very commonly, processes can be decomposed further into sub-processes, and sub-sub-processes, etc. In MS Visio, this can be represented by linking an activity in a process, to another page in the MS Visio file, where the diagram on that page represents the sub-process for that activity. This module looks for these types of links in the MS Visio output file and creates a variable to represent this information (the next module does the actual conversion to a MATLAB element). As a result, nested processes can be simulated in the dynamic scenario.

C. Transformation Engine Module

The transformation engine is one of the core modules of the tool. Its role is to map the MS Visio objects to SimEvents simulation objects. The major components of this step are described below:

1) Initial state: The initial state in the activity diagram represents the starting point of the process. This is realized using an entity generator in SimEvents. Within SimEvents context, entities represent discrete items of interest, such as packets in a communication system. For the purpose of this work, the entity can be viewed as 'walking' through the process and executing the activities. A large intergeneration time is chosen in order to ensure that one entity flows through the process. In the event of a fork (where multiple activities are performed in parallel), additional entities are created.

2) Activities: The action state objects in the MS Visio model represent the operational activities performed in the process. Each activity is realized using a SimEvents 'subsystem' block. The activity subsystem consists of infinite and single servers which represent the action of waiting for the appropriate resources to become available and then subsequently executing the activity.
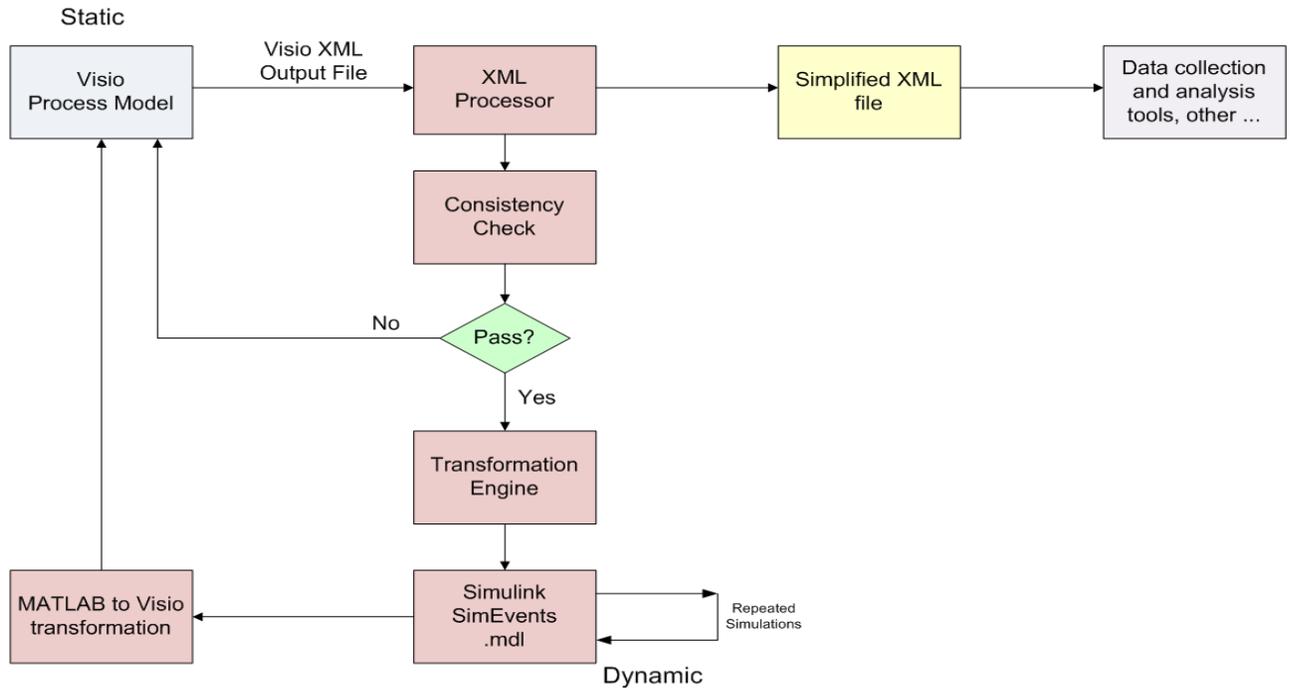
Fig. 2. Flow between Computational Modules of the Proposed Tool

The wait and execution times can be represented by any type of random distribution supported by SimEvents (e.g. Normal, Uniform, Exponential, Poisson, etc. [11]). The parameters can be exported from the original MS Visio file (via shape data) and extracted by the XML processor module, or they can be set within the algorithm in MATLAB. Currently, default values have been set for these times; however, future work involves coordination with Subject Matter Experts (SMEs), and extracting data from experiments to assign more appropriate values.

3) Sub-Processes: As mentioned above, the proposed tool can handle nested processes where an activity can be decomposed into another seperate sub-process. If nested processes exist in the original MS Visio diagram, this is translated in SimEvents using a subsystem block, where input and output ports represent the initial and final states of the sub-process.

4) Control Flow (Connectors): The control flow objects in MS Visio represent the connections between activities and the flow of the process. These objects are represented by connections between systems in SimEvents.

5) Decisions Blocks: The decision block objects in MS Visio represent a decision point in the process flow. In MS Visio, the deciding criteria are represented using guard conditions in the control flow object, and the branching probabilities are

denoted by the shape data. Once the transformation engine detects a decision block in the flow, it replaces it with a SimEvents subsystem consisting of a set attribute block and an output switch. The entity's attribute (which represents the output port) is determined by the branching probabilities defined in the MS Visio model. Whenever the process flow regroups into a single path, this is realized using a Path Combiner block.

6) Final State: The final state represents the end of the process. This is represented using an entity sink in SimEvents.

D. Simulation Capabilities

The SimEvents simulation model captures all of the action states (operational activities), swimlanes (operational nodes), and connections present in the MS Visio process model file. By default, timers are included at the beginning and the end of the process to compute the total execution time. In the case where intermediate timing labels are included in the activity diagram, intermediate start and read timers are also added to the SimEvents model.

The number of simulation runs, the output format, and the random seeds can all be set by the user. Since multiple simulation runs can be executed, the tool is suitable for algorithms such as Monte Carlo and sensitivity analysis [13].

By default, all timing values are captured. If decision blocks are present in the model, the decision block output port number taken by the entity is recorded for each run. This is particularly important for identifying which route was taken in the process, which in turn will ultimately help identify bottlenecks in the process.

E. MATLAB to MS Visio Transformation Module

This module transforms the simulation scenario back to MS Visio format. This is a significant capability of the tool, since it allows the user, once finished optimizing the simulation model, to obtain an equivalent MS Visio diagram to be shared with colleagues and coalition partners, thus allowing for an iterative development process.

In order to map the simulation model back to MS Visio, all of the relevant information from the SimEvents scenario is captured. Next, the proposed tool dynamically creates an MS Visio Visual Basic file (.bas). The Microsoft Visio Object Model ([23]) is used to create the objects in MS Visio. Particularly, the timing information and conditional probabilities are encoded as MS Visio shape data. This way, the resulting MS Visio model reflects the parameters which were optimized in SimEvents. Once in MS Visio, a macro is used to automatically import the Visual Basic file as soon as the document is opened, and then the new process drawing is automatically generated.

F. Integration with IBM Rational System Architect

An additional feature of the tool is that it can also read in diagrams from IBM Rational System Architect enterprise architecture software (SA) [24]. The motivation behind this additional capability is due to the fact that the CFWC is in possession of several licenses of this IBM tool and hence, databases from several past events are in that format. As a result, it was seen as beneficial to integrate SA into the executable architecture suite of tools. Currently, the proposed tool can input SA diagrams and convert them to SimEvents simulation scenarios. Writing to SA from MATLAB is not useful for our purposes.

V. EXAMPLES

A. Example 1: Translation Capabilities

This example demonstrates the translation capabilities of the proposed tool. For this example, the process model depicted in Figure 1 is considered. This model has two swimlanes ('Person A' and 'Person B') in the main model. Activity A has a hyperlink (can be decomposed) to the sub-process model depicted in Figure 3. Also, there is a decision block after Activity C which branches out to Activities D and E. The branching probabilities are denoted as 0.3 for Condition A to be satisfied (go to Activity D) and 0.7 for Condition B to be satisfied (go to Activity E).

The MS Visio model was translated to a SimEvents simulation model using the proposed tool. Figures 4 and 5 depict the generated simulation models for the main and subprocesses. Note that the tool captured all of the information in the original MS Visio diagram and converted them to equivalent SimEvents objects. The start and read timers compute the total time to execute the entire process (intermediate timers could be

added to the model if desired). The decision block is realized using a subsystem which assigns an attribute (representing the branching probability) to the entity and then the output port for the entity is determined based on that attribute.
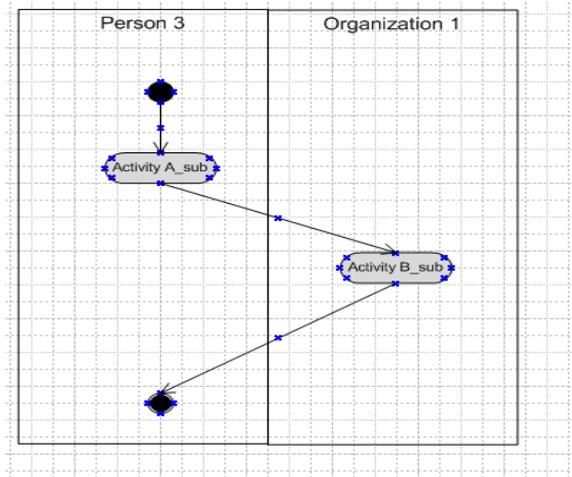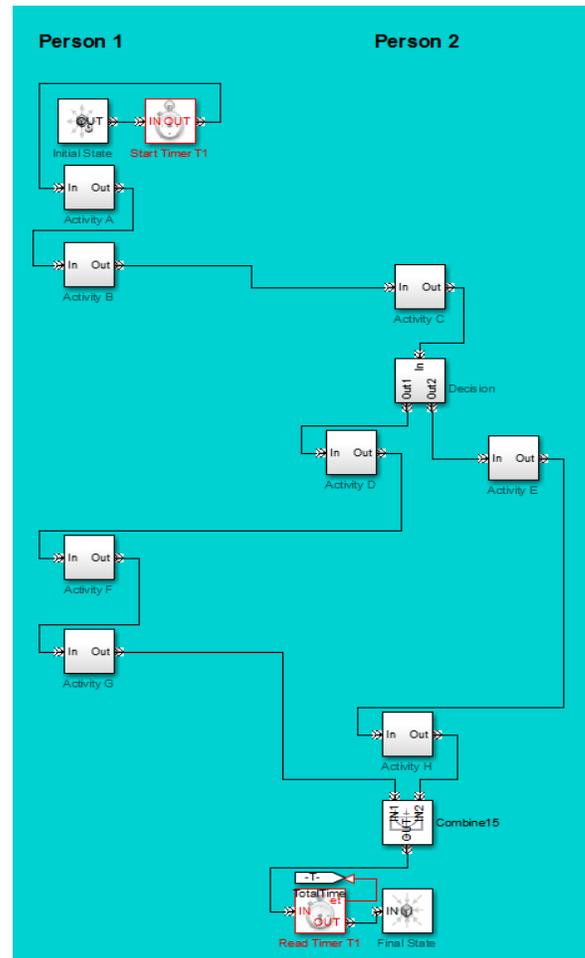

Fig. 3. Sub-process for Model in Example 1

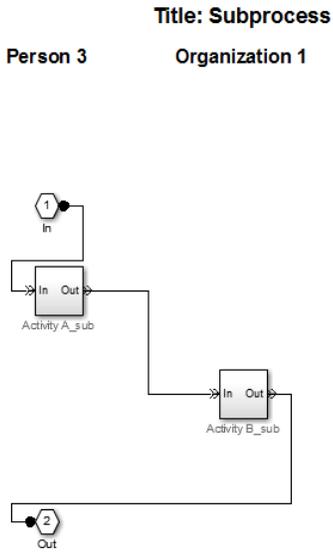Fig. 4. SimEvents Equivalent Model for Example 1

Title: Subprocess

Person 3   Organization 1

Fig. 5.   SimEvents Equivalent Model for the Sub-process in Figure 3

B. Example 2: Simulating Model with Decision Blocks

In this Example, the simulation capabilities of the proposed tool are demonstrated. Here, the SimEvents model in Figure 4 is considered. To demonstrate, 20 execution runs were simulated (the number of runs were set using the 'Simulation Parameters' window shown in Figure 6). The output results are provided in Figure 7 which shows a snapshot of the output Excel file. Note that the proposed tool can output the results to an Excel file, as well as to a MATLAB internal variable, allowing for analysis of the results.

Referring to Figure 7, it can be seen that the output port for the decision block is displayed for each run. Since there were two branches for the decision block in the model (with branching probabilities of 0.3 and 0.7 respectively), for each run, the entity traveled through one of the two blocks. As seen, when output port 1 was taken, the total process time was higher due the extra activity executed in that path. In addition, a description of the available paths in the model is provided at the bottom of the output file.
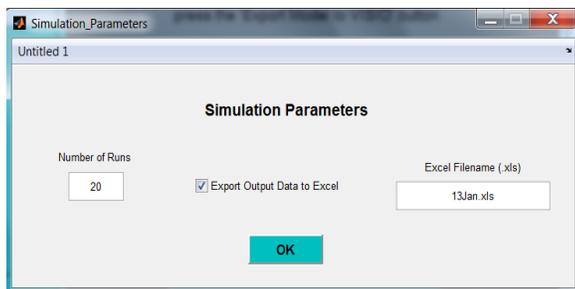
Fig. 6.   Simulation Parameters Window in MATLAB

	A	B	C	D	E
1	PortNum_Block5_Output	TotalTime_Output			
2	2	38.80061028			
3	1	45.53905565			
4	2	41.62262789			
5	2	39.07313626			
6	2	38.40059593			
7	2	39.54257872			
8	2	38.44576068			
9	2	39.67048125			
10	2	40.21101725			
11	1	47.07366108			
12	2	40.67480128			
13	1	44.98756518			
14	2	39.34995709			
15	2	39.46441981			
16	1	45.47193766			
17	2	39.81026673			
18	2	39.76603815			
19	1	47.08588697			
20	2	38.46459534			
21	2	39.81686107			
22					
23	Decision Output Blocks:				
24					
25	Block5 Output Ports	Block5 Destination Blocks	Block5 Destination Descriptions		
26	1	6	Activity D		
27	2	17	Activity E		
28					

Fig. 7.   Simulation Output Results for Example 2

C. Example 3: Process Diagram with Neural Network Models

In this example, the process model in Figure 1 is again considered. However, in this case, one of the activities (Activity B) is replaced with a Neural Network model [15], [16] to show the potential for advanced modeling using the proposed tool. For demonstration purposes, a fictional example is considered. Let's assume that Activity B can be decomposed into the sub-process model depicted in Figure 8. The activity 'Assign code' involves assigning a random three digit code to the execution run and then 'Classify into groups' involves the use of a neural network model which performs pattern recognition and classification in order to categorize the execution run into one of three groups.

To achieve this, the neural network toolbox in MATLAB was used [16]. A data set with 1000 samples of uniformly distributed three-digit codes was generated (where each digit is between zero and nine), and depending on the sum of the three digits, the targets were either one, two or three, which represent the different group classifications. A two-layer feed forward network with sigmoid hidden and output neurons was created (with 10 hidden neurons and one output neuron) as shown in Figure 9. The model was trained using the scaled conjugate gradient backpropagation algorithm [14]. The creation of the data set and creating/training of the neural net was all performed offline, prior to the process model execution. Next, the model was executed with the inputs feeding into the already created and trained neural network. The results for 30 runs are shown in Figure 10. As seen, for each run, the execution was classified into one of three groups. Although this example is fictional and relatively simple, it demonstrates the capability of the proposed tool to use advanced modeling techniques for
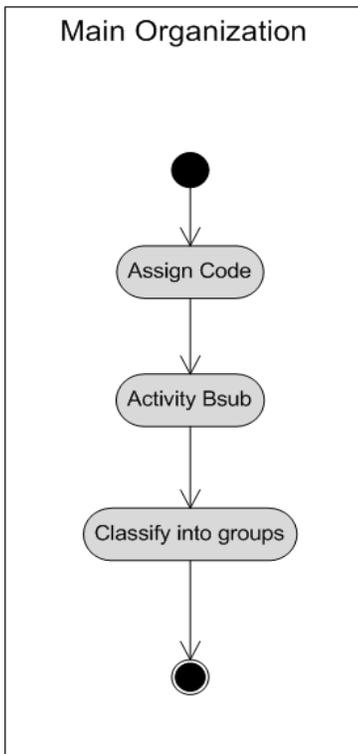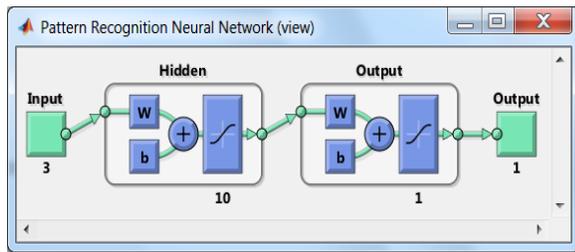
Fig. 8.   Sub-process for Activity B in Example 3

Fig. 10.   Snapshot of Simulation Output Results for Example 3

	A	B	C
1	Group	PortNum_Block5_Output	TotalTime_Output
2	3	1	51.59207997
3	2	1	50.91232242
4	1	2	44.25467338
5	2	2	43.71535048
6	1	2	44.34023301
7	1	2	43.87953133
8	3	1	51.03813628
9	2	2	44.99165788
10	1	2	44.67559434
11	2	1	50.68713662
12	2	2	45.31616608
13	3	2	44.27386336
14	2	1	50.90450779
15	1	2	45.37798224
16	1	2	44.32589224
17	2	2	44.18976507
18	1	2	43.47948562
19	3	2	44.6325098
20	2	2	44.08703151
21	2	2	43.86469372
22	3	1	51.68597136
23	2	1	50.74970826
24	1	2	44.28638659
25	2	2	43.94223201
26	2	2	46.40310447
27	2	2	43.90285242
28	1	1	51.49459215
29	1	2	44.00715246
30	1	1	51.1257278
31	2	2	43.81466683
32			

Fig. 9.   Neural Network Model for Group Classification used in Example 3

use in a wide variety of applications, with varying degrees of complexity. In addition, since the model resides in the MATLAB environment, any modeling and analysis algorithms which the user prefers can be employed.

VI.   CONCLUSIONS AND FUTURE WORK

A new executable architecture tool for the modeling and simulation of operational process models using MS Visio and MATLAB was presented in this paper. The new tool does not require any specialized software training (which is the case with some of the other available tools) and uses widely-employed software. This aspect of the approach allows for broad collaboration. The proposed capability converts a static architecture process diagram to a fully functioning dynamic simulation scenario in the MATLAB SimEvents environment. The bi-directional capability of the tool provides linkage back to MS Visio where a new static model can be automatically generated (based on the simulation scenario) which provides the means for sharing and iterative development. This paper provided a brief description of other available tools,

background information on architectures and process models, details of the development of the new approach and numerical examples which validate the capabilities and functionality of the proposed method.

Future work for this capability includes simultaneous simulation of multiple process models. This is so that a theater operation center (TOC) can be represented as accurately as possible, where multiple missions and processes are executed simultaneously. This capability will give significant insight into resource requirements and bottlenecks. In addition, in order to improve the fidelity of the models (particularly for timing information) it is intended to use data from future exercises and experiments to refine timing parameters in existing and future models. Furthermore, the Qualiware Lifecycle Manager Tool has been recently adopted as the DND enterprise architecture standard and is now available to anyone working in DND. As a result, it is of interest to investigate the possibility of integrating Qualiware into the proposed executable suite of tools.

REFERENCES

[1]   *Department of National Defence and the Canadian Forces Architecture Framework (DNDAF) Volume 3: DND/CF Architecture Data Model (DADM) version 1.8.1.*   Ottawa, ON: Department of National Defence, 2013.

[2]   J. Garcia, "Executable architecture analysis modeling," in *Proc. 2007 Spring Simulation Multiconference*, March 2007, pp. 177–184.

[3]   W. Robbins, "Achieving DODAF-driven simulations through executable architectures," in *Proc. 2009 Spring Simulation Multiconference*, March 2009, pp. 231–238.

[4] R. Youssef *et al.*, "Toward an integrated executable architecture and M&S based analysis for counter terrorism and homeland security," *Defence Research and Development Canada Internal Report*, 2006.

[5] D. Allen, R. Chow, K. Trinh, and P. Farrell, "Information management processes in support of major event security," in *Proc. 13th International Command and Control Research & Technology Symposium(ICCRTS): C2 for Complex Endeavors*, Bellevue, Washington, June 2008.

[6] S. Hutchins *et al.*, "Modeling and simulation support for the standing joint force headquarters concept," in *Proc. 10th International Command and Control Research & Technology Symposium(ICCRTS): The Future of C2*, McLean, VA, June 2005.

[7] G. Schacher *et al.*, "SJFHQ Simulation," *Naval Postgraduate School Technical Report*, 2004.

[8] *G2 Rethink*. Burlington, Massachusetts: Gensym Corporation, 2007.

[9] *Simul8 Simulation Software*. Boston, Massachusetts: Simul8 Corporation, 2013.

[10] *COREsim*. Blacksburg, Virginia: Vitech Corporation, 2009.

[11] *MATLAB version 8.1.0.604 (R2013a)*. Natick, Massachusetts: The Mathworks Inc., 2013.

[12] *SimEvents Reference Guide (R2013b)*. Natick, Massachusetts: The Mathworks Inc., 2013.

[13] R. Rubinstein, *Monte Carlo Optimization, Simulation and Sensitivity of Queuing Networks*. New York: John Wiley, 1986.

[14] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New Jersey: Prentice Hall, 1994.

[15] C. Lin, "Neural network-based fuzzy logic control and decision system," *IEEE Trans. Comput.*, vol. 40, no. 12, pp. 1320–1336, 1991.

[16] M. Beale, M. Hagan, and H. Demuth, *Neural Network Toolbox 6 User Guide*. Natick, Massachusetts: The Mathworks Inc., 1992–2013.

[17] M. Gen and R. Cheng, *Genetic Algorithms and Engineering Optimization*. New York: John Wiley and Sons, 2000.

[18] M. Ball, R. Funk, and K. Wheaton, "Combined forces air component command: Technical note 1- Data collection," *Defence Research and Development Canada Contract Report (CR-2012-281)*, 2012.

[19] ——, "Combined forces air component command: Technical note 2- Department of defence architecture framework (DoDAF) analysis," *Defence Research and Development Canada Contract Report (CR-2012-282)*, 2012.

[20] K. Wheaton, "Combined forces air component command: Technical note 3- Hierarchical goal analysis and performance prediction," *Defence Research and Development Canada Contract Report (CR-2012-283)*, 2012.

[21] M. Ball, R. Funk, and K. Wheaton, "Combined forces air component command: Technical note 4- Systems analysis summary report," *Defence Research and Development Canada Contract Report (CR-2012-284)*, 2012.

[22] F. Brundt, *Private Communication*, USJFCOM Experimentation Directorate (J9), Suffolk, Virginia, 2003.

[23] "Microsoft Visio Object Model," http://msdn.microsoft.com/en-us/library/cc160740.aspx, [Online Reference].

[24] *IBM Rational System Architect version 11.4*. Armonk, New York: IBM Corporation, 2013.