Towards more effective and efficient tactical scenario generation

Arno Gerretsen, Joost van Oijen
Netherlands Aerospace Centre (NLR)
arno.gerretsen@nlr.nl, joost.van.oijen@nlr.nl

Philip Kerbusch, Gwen Ferdinandus
TNO Defence, Security and Safety
philip.kerbusch@tno.nl, gwen.ferdinandus@tno.nl

Introduction

Nowadays the use of simulation for training and mission preparation of military personnel is increasing. To ensure the effectiveness of simulation systems the timely availability of sufficient tactical scenarios both in quality and number is essential. In training simulations the scenario has to be adapted to the specific needs of the trainee and when the trainee uses the simulation multiple times there has to be enough variation in the scenarios to remain effective. When simulation is used for mission preparation it is required that new scenarios can be built quickly and efficiently in preparation of future missions.

However, the development of scenarios is an often laborious process that can take a lot of time and effort. This means it is difficult to ensure enough variation in scenarios and to respond quickly to requests for new scenarios. The problems with the scenario development process have several underlying reasons. Most of the tools involved in scenario development are more engineering than operational oriented, requiring a technician with limited operational experience to translate the operational concepts and learning goals into an executable scenario. This often results in the technician having to consult with the scenario requester several times to get all the required information, which in turn delays the development process. Secondly, in most cases scenarios have to build from scratch every time. Reuse of complete scenarios within the same simulation system is difficult as the requirements differ depending on the user. Reuse of scenarios between different simulation systems is often not possible because each system uses its own proprietary tools and formats. This means that similar scenarios have to be developed multiple times for different systems.

To make the development of tactical scenarios more efficient and effective the Netherlands Aerospace Centre (NLR) and the Netherlands Organisation for applied scientific research (TNO) have defined a harmonized approach between process, technologies and standards for the Dutch Defence. The research focussed on three key themes:

1. How to ensure that the person requesting a new scenario and the builder of the scenario have a common understanding of the scenario requirements, so that the right scenario is developed. Which technologies can support the definition of the scenario concept?

2. How to build a new scenario efficiently and effectively? Can reusable and configurable scenario components and smart algorithms ease the generation of scenarios and ensure that operational personnel are able to quickly turn their ideas into a scenario?
3. How to allow the reuse of existing scenarios across different simulation systems? Which standards are needed to share a scenario and which supporting technology is needed to convert the scenarios from the standardized scenario specification into the native representation of the simulation system?

Within the research project prototypes of new technologies and standards have been developed to address these issues. The prototypes have been evaluated by scenario builders of the Dutch Armed Forces to ensure their applicability and added value in an operational setting. This paper presents the results of the project and identifies the forward to improve scenario generation for tactical simulation systems.

Scenario development process

A key concept in the project was the optimal integration of the process, tools and standards used in scenario development. Therefore the first step in the project was to study the current scenario development process that is used in the Dutch Armed Forces and identify options for improvement. The high level scenario development process consists of the five steps as shown in Figure 1:

1. Definition of the requirements for the new scenario. This is typically done by operational personnel that are responsible for the training or mission preparation.
2. Definition of the conceptual scenario, this is a first design of the scenario where all key entities, events and interactions are identified. This step is typically performed jointly by the operational personnel and the simulator instructor or simulator technician.
3. Build the scenario into the simulation system. The idea of the conceptual scenario is implemented in the scenario tool of the simulator and all details of the entities, behaviours and events are entered into the tool. This is typically done by the simulator instructor or the simulator technician.
4. Test the scenario in the simulator to ensure that all events and behaviours execute as intended and that the intended usage can be satisfied.
5. Execute the scenario and use it to perform training or mission preparation.

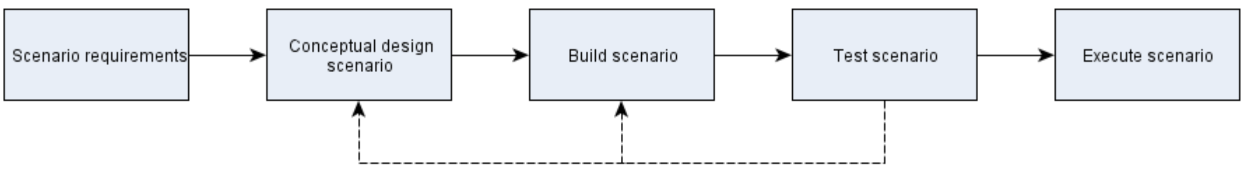

Figure 1: Scenario development process

In the next sections a number of challenges within the scenario development process are discussed, along with the approaches that have been investigated in the project to solve these challenges.

Conceptual scenario development

In the second step of the scenario development process information has to be exchanged between operational personnel and technical simulator personnel. Currently this information exchange is typically done in an ad-hoc manner. Optimizing this information exchange has two benefits:

1. By making sure the scenario intents are defined clearly at the start, the right scenario can be developed more quickly and less iterations will be needed to reach the intended end result.
2. If the intents of the scenario are defined in structured way this information can be stored together with the scenario. Future modifications or reuse of the scenario will become easier when the original intents of the scenario are available with the scenario.

Within the project it has been investigated if a structured scenario template can be used to assist in the first phase of the scenario development. Using a structured template enforces the operational personnel to define their requirements in detail, addressing all aspects that need to be considered during scenario development. Furthermore, this information can be stored and reused for other scenarios.

Evaluations showed that in general this concept works well, but the required information about a scenario varies a lot over the different military domains and systems. Therefore the template would have to be flexible enough to accomodate these variations.

In the prototype the template was integrated into a scenario development tool. To the scenario builder this offers the benefit that he does not have to re-enter all the information manually, saving time and reducing the chance of mistakes. The scenario development tool can also analyse the entered information and speed up the scenario development, for example by suggesting inserting certain entities into the scenario based on the information in the requirements.

For the operational personnel it is not practical however to have to enter the information directly into the scenario development tool, because they often do not have access to these tools. Therefore the best implementation would be to have the templates available on the normal operational systems, e.g. the military intranet, and save the information in a structured format that can easily be read and interpreted by scenario development tools.

Efficient scenario building

The second topic that has been investigated in the project is efficient scenario construction. Building a scenario from scratch requires a lot of work, since all entities and their behaviours have to be added manually. Moreover the builder needs extensive technical know-how of the simulator system to implement the required behaviours. Finally, to ensure the entities behave as expected the builder will have to test the scenario every time he makes a change. Since the test environment is not always the same as the development environment, and might not always be available, the development is often delayed.

We investigated if scenario development can be made more efficient by using scenario components. These components are designed to encapsulate technical details and provide easy-to-tune parameters at the operational level. They are reusable over different scenarios and the task of the builder is mainly to select, tune and connect the required components.

To see how scenario components can ease the development process, consider how a convoy of vehicles that has to follow a route is typically implemented:

The scenario builder has to insert all the vehicles of the convoy one by one into the scenario. He has to make sure that they are positioned correctly on the road and with the correct spacing between each other. Next, for each vehicle, the builder has to define behaviour to follow a specified route at a given speed. This route has to be defined by specifying all waypoints of the intended route. The builder has to make sure that this route follows the roads correctly.

Defining a convoy requires a lot of actions to be performed by the builder. By using scenario components this task can be reduced to:

The scenario builder selects the convoy component in the scenario tool. In the properties of this component he can specify how many vehicles the convoy should have. He can also specify the starting and end point of the convoy route. With that information the scenario component can be initialized and the vehicles are automatically placed in the right formation and, based on analysis of the road network, the optimal route to the end point is assigned to them.

This approach is called declarative modelling where the builder specifies *what* he wants to achieve and the system determines *how* to achieve this. With this approach the scenario builder has to enter less information as many of the technical details are automatically derived. To achieve this, the scenario tool needs to have information about the elements from which the component is build up and how these interact and can be integrated into the scenario. These semantics are stored in an ontology. Combined with procedural generation techniques, it allows scenario components to be instantiated into a scenario.

In the example of the convoy, the procedural generation techniques determine the best location and route for the entities. To achieve this, the algorithms require information about the terrain and the roads available. The same techniques can also be used to derive other properties of the entities within the component, for example to assign the right tasks to all subordinate entities based on the task of the commanding unit.

Within the project we demonstrated using a prototype that the concept of semantic scenario components and procedural generation can be used for the development of scenarios. It can be a useful technique to reduce the workload of the scenario builder and to increase the ability to reuse parts of scenarios.

At the evaluation of the concept at the Dutch Armed Forces it was recognized that such an approach could reduce the workload a lot, but concern was also voiced that algorithms can never replicate all aspects of military operations accurately. However, there will already be an efficiency increase if the techniques reduce the amount of manual and repetitive input that is required. The builder can spend his attention on checking and refining the scenario layout as suggested by the algorithms, instead of entering low-level details. This will reduce the amount of time needed to develop a scenario.

Reuse of scenarios

The third problem that has been investigated in this project focuses on the reuse of existing scenarios in different simulation systems. Current simulation systems typically use their own proprietary tools and formats to develop and store scenarios. If scenarios can be reused in different systems, it would no longer be needed to develop similar scenarios multiple times for different simulation systems. Two main concepts have been investigated in the project that will allow reuse of scenarios:

1. A standardized scenario specification.
2. The use of converters to translate scenarios from standardized specifications into the specific formats required by a simulation system.

Within the military domain two standards already exist for scenarios and these have been investigated as starting point for a standardized scenario specific for tactical simulations. These standards are the Military Scenario Definition Language (MSDL) and the Coalition Battle Management Language (C-BML), both developed by the Simulation Interoperability Standards Organization (SISO).

MSDL is considered a good basis for a standardized scenario specification. However the MSDL standard has been developed for C2 systems in the land domain, so some extensions are needed to make it suitable for tactical simulation systems as well. The main areas requiring extension are:

1. Improved support for other military domains than land. For example for the naval or air domain different formations should be supported.
2. Providing more detail of the entities in the scenario. For typical C2 applications these entities need to be shown on a map, but for tactical simulations they are represented in a 3D world. For example more detail is needed to specify the exact type and subtype of an entity, but also aspects like the visual appearance.
3. MSDL is intended to store the initial situation of a scenario, but within a scenario it should also be defined how what developments take place during a running scenario, e.g. when events will occur and which behaviours the entities will show.

For point 1 and 2 we have defined extensions to the MSDL standard, such that this information can be stored as well. Wherever possible, existing enumerations from the SISO Enumerations Working Group (EWG) have been used, as these are common in many simulation applications.

For the third point C-BML has been evaluated to check whether this standard, which is intended for orders and reports between C2 systems and entities during simulation runtime, can also be used to express the desired flow of the scenario beforehand. The current C-BML standard was however not easy to use, so we defined our own data model to store information on the tasks and events in the scenario. The structure of this data model aligns with the concepts of C-BML and hopefully the new C2SIM PDG that is working on combining MSDL and C-BML will include a way to define the flow of the scenario in their data model.

Once the scenario is stored in a standardized format, the next step to achieve reuse is to use this information in different simulation systems. Because most systems do not support MSDL natively, we have investigated the use of converters to translate the information into the specific format of the simulation system. Within the project a number of prototype converters have been build that allow simulators and scenarios to be initialized from the standardized scenario specification. For systems that have a well-documented API or a well-documented scenario format, it is possible to create such a converter. In other cases the manufacturer of the simulation system would have to be involved in this development.

In the project it was demonstrated that two scenario tools, one used in a flight simulator and the other used in a land-based simulator at the command level, could be initialized from the same scenario file. And when the scenario was ran in both tools they showed the same flow of events and behaviours. This illustrates that reuse can be achieved with this approach.

The main problem encountered was to specify the intended behaviours of entities at the right level of detail. If they are specified too high level, different simulation systems have too much freedom in how the behaviour is implemented. But if the specification is too detailed it might not align with the different techniques that simulation systems use to represent the behaviour. Within the project we used relative high level tasks, in line with the JC3IEDM data model. In that case you have to rely on different simulation systems implementing the detailed behaviours in similar ways, e.g. a task for an aircraft to perform an air-to-air attack should be implemented similarly in different systems to get the same behaviour. Therefore it would be recommended to define a lower level specification of the intended behaviour that is still independent of a specific simulator implementation.

Conclusions and way ahead

Within this research project a number of concepts and technologies have been investigated and demonstrated that will allow the development of tactical scenarios to be performed more efficient and effective in the future. Thereby making it possible to meet the increasing demand for the development of new scenarios, resulting from the increasing usage of simulation systems for training and mission preparation.

The concept of declarative modelling and semantic scenario components has a lot of potential to reduce the workload while building scenarios. It also allows information to be entered at the operational level, instead of the technical level. To further operationalize this concept the right algorithms need to be defined that can perform procedural generation for the relevant military aspects.

By extending existing standards, tactical scenarios can be stored in a standardized way, which makes reuse in different simulation systems possible. Each system will need to have a converter that can translate to the internal scenario format. To make reuse easier the intended behaviour should be specified more accurately. There is no good standard for this yet. The specific extensions that were made for tactical simulators would ideally be integrated in the C2SIM standard, which is combining MSDL and C-BML.

Finally the information exchange between operational personnel requesting scenarios and the technical personnel who are building the scenarios should be optimized. This can be facilitated by making this an explicit step in the development process. A structured template offers a way to store the intent of the scenario. This also ensures that the information is available when the scenario needs to be modified or reused later on.