



Exploring Large Language Models for Collaborative Scenario Development in Simulation-Based Military Training

Joost van Oijen^(✉), Thomas Bellucci, and Chihab Amghane

Royal Netherlands Aerospace Centre, Amsterdam, The Netherlands
{Joost.van.Oijen, Thomas.Bellucci, Chihab.Amghane}@nlr.nl

Abstract. This study explores the role of Large Language Models (LLMs) in the context of human-AI collaborative scenario development for military simulation-based training. We propose a conceptual framework that organizes scenario development into three key phases, supported by a sequence of LLM modules: from defining a training design and creating a conceptual scenario, to generating a scenario specification for execution in simulation. The approach supports the integration of external domain-specific knowledge sources, such as trainee qualification profiles, to ensure scenarios adhere to organization-specific training principles. Using a case study of helicopter pilot training, we evaluate how LLMs can enhance the efficiency and effectiveness of scenario development compared to traditional methods. Our findings highlight the promise of LLMs in streamlining scenario development workflows while maintaining instructional integrity.

Keywords: Training · Simulation · Scenario Development · Large Language Models

1 Introduction

In the military domain, warfighters require continuous training to maintain mission readiness and operational competence. Simulation-based training in synthetic environments provides a cost-effective means for personnel to develop and stay current on relevant competencies, tactical skills, or mission-specific expertise through customized scenarios. To meet the demands of modern military training, training sessions should support personalization to address individual skill gaps, allow team training, and be adaptable to current operational needs regarding specific missions, environmental characteristics, or emerging threats and conflicts.

The effectiveness of simulation-based training thus heavily relies on the quality of training scenarios to meet these demands and the speed at which they can be developed. Training scenarios should be grounded in instructional design to ensure the necessary learning experiences for trainees. They should be conceptually coherent with realistic environments, actors, and interactions to enable effective skill transfer through targeted fidelity in simulation. Finally, scenarios should be adaptable and efficiently modifiable

to address evolving training needs, whether tailored to specific skill-, task-, or mission contexts. These requirements call for an effective scenario development workflow.

In current practice, scenario development is a process that commonly involves multiple personnel with distinct expertise. On one side, instructional designers or instructors translate training objectives into scenario needs. Conversely, scenario developers or programmers translate those needs into executable scenarios within a target simulation system. Earlier studies have shown that inefficiencies in the workflow occur due to ineffective communication between both sides on scenario needs and details, leading to additional interactions and iterations in scenario design, and prolonging development time [1]. Furthermore, the authoring of executable scenarios was found a highly time-consuming and labor-intensive task in many current editing tools.

This paper explores the potential of generative AI for computer-assisted scenario development to enhance efficiency and effectiveness within the workflow. Recent advancements in generative AI, particularly Large Language Models (LLMs), have demonstrated significant benefits across various domains for optimizing human workflows [2]. For example, in the context of adaptive instructional systems (AIS), LLMs can alleviate the authoring bottleneck in AIS development, facilitating the authoring of instructional content and the development of domain, student, or pedagogical models [3]. Specific to scenario creation, LLMs exhibit relevant abilities, such as information retrieval [4], extensive world knowledge [5], storytelling [6], and coding [7]. This study examines these qualities in an approach to human-AI collaborative scenario development. Based on a generic model of the scenario development process developed in collaboration with end users, we examine how LLMs can support this process. Using a proof-of-concept system for LLM-enabled scenario creation, we investigate its application to simulation-based training for helicopter pilots.

The outline of this paper is as follows. First, we describe related work regarding military simulation scenarios, automated scenario generation, and LLMs in AIS (Sect. 2). Following, we present a method for LLM-based collaborative scenario development (Sect. 3), after which we describe a proof-of-concept implementation (Sect. 4). Based on a use case example (Sect. 5), we discuss opportunities and challenges of the proposed approach (Sect. 6) and summarize our findings (Sect. 7).

2 Background

This study on human-AI collaborative scenario development aims to combine three areas of work, namely (1) the scenario development process from a human engineering perspective, (2) approaches that support training needs-driven scenario generation, and (3) the role of LLMs in AIS. Below we touch upon related work in these areas.

2.1 Scenario Development for Military Simulation

Scenario development is an important part of military simulation across different application domains such as training, analysis, and decision support. To promote a common view on scenario development, the Simulation Interoperability Standards Organization (SISO) has developed a guideline on scenario development for (distributed) simulation

environments [8]. The aim is to provide a common view of the development process while describing the relationship with the overarching Distributed Simulation Engineering and Execution Process (DSEEP) [9]. The guideline emphasizes the need for thoughtful scenario development and well-specified scenarios to ensure proper representation in a simulation environment to reflect a designer's original purpose. In this respect, scenarios should be complete (contain sufficient information), consistent (operationally correct), and understandable (easily accessible by future users). Three types of scenarios are distinguished, corresponding to three development steps. Operational scenarios are authoritative descriptions by subject matter experts (SMEs) that specify what has to be represented in a simulation environment to fulfill user's needs and objectives. Conceptual scenarios provide more fine-grained descriptions of the part of the world to be simulated, providing sufficient information for the scenario to be implemented. Finally, executable scenarios are scenario specifications at the level of detail necessary for initialization and execution in a simulation environment. A widely known standard for specifying executable scenarios is the Military Scenario Definition Language (MSDL), allowing the description of the initial state of a scenario [10].

In the context of training scenarios, in an earlier study, we identified a similar three-step development process based on findings from various military simulation training systems in the Netherlands [1]. First, the training needs of the trainee(s) are captured to establish high-level scenario needs in terms of e.g. what competencies, skills, tasks, or mission types are to be trained. Second, these scenario needs are translated to a conceptual scenario that expands on scenario specifics such as geographical mission areas, environmental conditions, players and their roles, and storylines. Finally, the conceptual scenario is built and tested within a target simulation system using the system's scenario editing tools. Depending on the specific organization, different personnel roles with distinct expertise were involved in the process, including instructional designers, instructors, and scenario developers. Based on stakeholder interviews, several inefficiencies were identified in the process. First, it was found difficult to clearly define training needs and translate these into concrete scenarios that can offer the learning experiences to fulfill those needs. Second, it was recognized that the personnel building the actual scenarios do not always have sufficient operational knowledge about the military operations being simulated and trained, leading to additional iterations and interactions with SMEs. Overall, the complete process was found to be time-consuming, taking days or sometimes weeks to develop new scenarios from scratch.

2.2 Training Needs-Driven Scenario Generation

(Semi-)automated scenario generation techniques can alleviate the authoring process as described earlier. Of particular interest are approaches that can relate explicit training or learning objectives with domain-specific scenario elements, enabling the creation of purposeful scenarios and training events. Such scenarios ensure that trainees can be formally assessed on the original objectives during or after training. Several researchers have proposed frameworks that integrate these techniques, demonstrating their potential to enhance scenario design.

In [11], the authors propose an automated scenario generation method applied to scenario-based disaster management training. The method is capable of selecting relevant injects into a scenario tree as training events (c.f. experiences) that maximize potential learning for current given training objectives. The range of possible scenario learning experiences is managed by a (human-authored) scenario catalog that can be used to train a neural network-based adaptation policy [12]. In [13], a competency-based scenario generation framework is proposed and applied to the domain of synthetic training of soldiers. Instructors can provide learning objectives in the form of competencies, skill levels, or complexities. These objectives are used during scenario generation to select relevant training events on which learners can be assessed. The framework can leverage different AI-based techniques for developing interactive narratives and storylines. For example, the authors have explored the use of reinforcement learning for scenario generation [14]; others have explored the use of genetic algorithms [9]. Another approach is described in [15], where the authors introduce the concept of trend-aware scenario authoring. It addresses an important need to reflect trends from real-life operations into training scenarios. In the explored domain of pilot training, identified trends include e.g. new climate/weather conditions, type of threats, specific emergencies, aircraft updates, etc. The system can recommend instructors to incorporate scenario elements in the scenario design as instructional events. The authoring approach is inspired by so-called Training Objective Packages (TOPs), which formally encode the conditions for trainees to meet the training objectives, accompanied by metrics to measure and assess trainee performance [16].

2.3 Large Language Models for AIS

Since their mainstream introduction, Large Language Models (LLM) have been explored in many domains to optimize human workflows. Their ability to comprehend context and perform language-related tasks at a human-like level makes them highly valuable in boosting productivity [17]. When considering the task of scenario development, relevant abilities include e.g. information retrieval [4], extensive world knowledge [5], storytelling [6], and coding [7]. Recent advancements have led to the development of multi-modal LLMs, which extend their capabilities beyond text to include other modalities such as images, audio, and video. This enables these models to achieve a richer and more comprehensive understanding of the world by integrating diverse types of information [18].

In the domain of AIS, LLMs provide ample opportunities to alleviate the authoring bottleneck in AIS development. They can be utilized across major components of AISs such as content creation, domain models, student models, and pedagogical models [3]. In [19], the authors highlight the opportunity for LLMs to empower teachers in learning analytics development, where LLMs can support teachers as co-designers, increase personalization in learner feedback, or generate learner material. LLMs can facilitate a human-centered design approach with inter-stakeholder involvement to reflect the needs of teachers, educational designers, and learners, and support the theoretical grounding of educational design [20]. Their capabilities in knowledge engineering can be leveraged to support knowledge acquisition, elicitation, and organization of knowledge in computational form [21]. Approaches for incorporating external, domain-specific knowledge

sources, such as retrieval augmented generation (RAG), have been shown to enhance the performance of LLMs in the education domain [22].

3 Method

In this study, we investigate the role of LLMs in human-AI collaborative scenario development for military simulation-based training. The proposed method follows the three steps for scenario development outlined in the SISO guideline described earlier – operational, conceptual, and executable scenarios – while framing them from a training application perspective. The method is illustrated in Fig. 1 and shows the process of transforming high-level training needs into executable training scenarios for military simulation. Throughout this process, LLMs take on various roles to collaborate with, and assist, the human user. The method is applied within the scope of a use case focused on helicopter aircrew training. In the remainder of this section, each development step is discussed in more detail, followed by a description of a proof-of-concept system that implements the method (Sect. 4). Afterwards, the system is evaluated through a case study (Sect. 5).

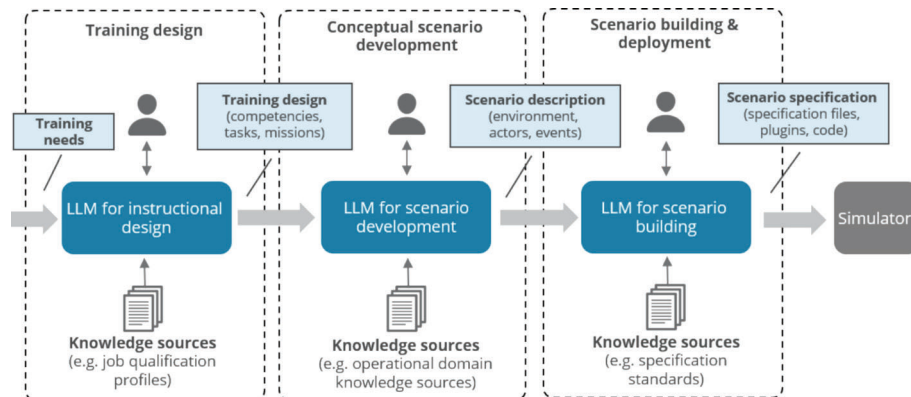


Fig. 1. Training needs-driven human-LLM collaborative scenario development where an LLM assistant supports the user throughout different steps of scenario development.

3.1 Training Design

In the first step, the goal is to translate high-level training needs into concrete training designs, describing who has to be trained on what kind of capabilities (e.g. competencies, tasks, missions). In this step, the role of the LLM is to support a human instructor in this translation. Hereby it acts as an ‘SME assistant’ equipped with relevant knowledge about the training domain within the organization.

In this study, we employed existing knowledge sources in the form of job qualification profiles of helicopter pilots, which have been established based on training

needs analysis. These qualification profiles are documents that contain detailed function descriptions for specific military roles, with required capabilities for achieving combat readiness. Relevant concepts include:¹

- **Missions:** Type of missions that are relevant for the role. For example, relevant missions for an attack helicopter pilot include *Close Air Support* and *Reconnaissance*; or for a transport helicopter pilot *Aeromedical Evacuation* and *Trooping* (transport of combat troops).
- **Tactics, Techniques & Procedures (TTP):** These are individual tasks that should be mastered for a role. Different mission types can be related to different sets of TTPs. TTPs can be trained individually through so-called part-task training. Examples include *Close Combat Attack*, *Reduced Visibility Landing*, *Formation Flying*, or *Sensor and Weapon use*.
- **Complexity factors:** These are operational conditions that make a mission or task more or less complex to perform successfully. Complexity factors are organized in categories and include for instance: *Meteorological* (day/night, precipitation, wind), *Geographical* (terrain, ground surface, vegetation, infrastructures), *Material* (fuel, equipment), *Threats* (enemies, weapons), *Contingencies* (emergencies, failures), or *Cooperation* (teamwork, communication).

The above concepts from the qualification profiles are used to define a formal specification for a training design and can support different types of training: part-task training for training specific TTPs; or whole-task or mission training for training specific mission types. Complexity factors can be specified to offer more tailored training environments and adjustable skill levels during training, enabling more personalized training.

During this step, the human instructor can converse with the assistant and express the intent to train one or more trainees on specific missions or TTPs in specific conditions. The assistant can then propose a suitable training design, or set of designs with varying complexities. Additionally, the instructor may pose open-ended questions to the ‘SME assistant’ concerning the proposed training design or retrieve general information regarding TTPs, mission types and complexity factors from the available knowledge sources, serving as a general question-answering system.

3.2 Conceptual Scenario Development

In the second step, the goal is to translate training designs into conceptual scenarios, which provide descriptions of the training environment in which specified training tasks or missions can be trained. Similarly, to the previous step, the role of an LLM is to support a human instructor in this translation, where it acts as a ‘creative designer’ who can propose or adapt scenarios while maintaining consistency with the instructional objectives outlined in the previously-developed training design.

Conceptual scenarios should include sufficient details to enable the generation of executable scenarios in the subsequent step. Relevant aspects of military mission environments encompass geographical features such as locations and terrain characteristics,

¹ For purposes of publication, the listed examples of missions and TTPs were modified from the original source profiles.

environmental conditions like weather and time of day, and the presence of actors, including locations, equipment, and behaviors of friendly forces, adversaries, or civilians. Additionally, conceptual scenarios may outline specific storylines, events, or interactions that guide scenario evolution to elicit targeted learning experiences associated with particular training tasks.

In this step, the generative capabilities and extensive world knowledge of the LLM are utilized to create a coherent and operationally realistic scenario that meets the constraints of the training design. The instructor can converse with the assistant to refine scenario elements, introduce modifications, or specify additional constraints. The assistant can fulfill the important requirements for conceptual scenarios as defined in [8]: *completeness*, by enforcing the generation of all necessary scenario elements; *consistency*, by ensuring alignment with the training design; and *understandability*, by producing scenario descriptions in various output formats. To enhance the LLM's ability to generate accurate, domain-specific content, external knowledge sources can be leveraged, such as doctrine documents, military mission or Intel reports.

3.3 Scenario Building and Deployment

In the final step, conceptual scenarios are translated into executable scenarios suitable for deployment on a target simulation platform. Executable scenarios are defined as machine-readable specifications. The role of the LLM is to alleviate the manual authoring burden on scenario builders by leveraging its code generation capabilities, hereby acting as a 'developer' or 'programmer' assistant. For example, an LLM can assist in specifying scenarios using MSDDL, an international standard for military scenario definition [10], or alternative custom formats supported by a specific simulation platform.

The translation process requires careful orchestration, as it involves precise and customized transformations of individual scenario elements to comply with the syntactic and semantic standards of the target specification. This process frequently entails converting semantic and fuzzy concepts into specific value types defined by ontology schemas. For instance, relative location descriptions (e.g., '5 km east of <specific village>'), may be translated into geocoordinates and descriptions of military units or equipment may be mapped to standardized types, such as those defined by SISO [23] (e.g., a numerical identifier associated with an Apache AH-64 helicopter). Additionally, descriptions of events or actor behaviors may need to be transformed into compatible scenario scripts or behavior models supported by the target simulation platform.

Due to the diversity of translations required in this step, a detailed analysis lies beyond the scope of this study. However, the success of an LLM in this context largely depends on its ability to meet the required syntactic and semantic standards. Syntactic compliance refers to the generation of valid specifications in the target programming language. For example, LLMs generally perform well with formats such as JSON or Python due to their relative simplicity and abundance within LLM training datasets. Semantic compliance, on the other hand, involves generating specifications that adhere to the ontological standards of the target system. To support semantic compliance, external knowledge sources, such as documented standards or ontologies, can be integrated into the process.

4 Proof-of-Concept

The human-AI collaborative scenario development method presented in Sect. 3 was implemented in a proof-of-concept system designed for explorative research. The system is illustrated in Fig. 2. And emphasizes the collaborative interaction design between the user and the LLM during scenario development. Leveraging their extensive conversational training, LLMs can facilitate open-ended, collaborative communication with the user, allowing the user to express intents and provide incremental feedback to the system in natural language.² This capability enables a ‘human-in-the-loop’ feedback mechanism, where users can refine suboptimal outputs, contribute additional details, and guide the scenario’s progression.

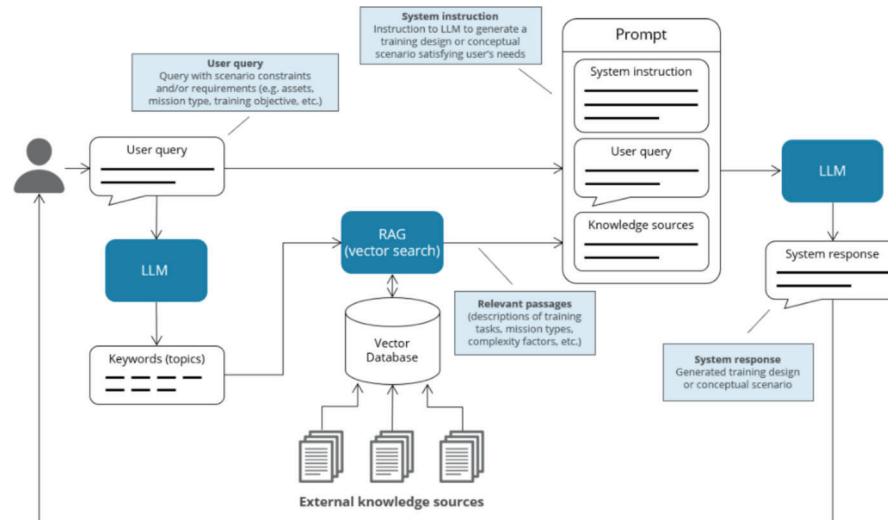


Fig. 2. Interaction design between the user and LLM³. The LLM’s response is generated based on prompting that combines user queries, system instructions, and external knowledge sources.

In the system, the LLM is aligned to the training domain by integrating external domain-specific knowledge sources, such as the pilot qualification profiles described in Sect. 3.1. To this end, Retrieval Augmented Generation (RAG) [24] is used to enable the system to dynamically incorporate relevant information from external sources in its responses as shown in Fig. 2. For each user instruction, the LLM first extracts a list of keywords or topics, which are used to retrieve pertinent passages from external sources stored in a vector database; the retrieved passages are then appended to the user’s instruction to generate a contextually informed response.

Additionally, the system employs distinct internal instructions to guide the LLM’s responses based on the user’s input at each stage, whether the goal is to generate a

² This holds for LLMs which have undergone a supervised conversational ‘chat’ finetuning.

³ The LLM used is based on an in-house facility that supports a variety of open-source LLMs.

training design, a conceptual scenario, or a scenario specification. This approach ensures that the generated output aligns closely with the specific requirements of each stage in the scenario development process.

During the interaction, the system is responsible for maintaining internal consistency of the scenario and adherence to the training design. Additionally, by virtue of its conversational nature, the system can provide explanations for its design choices when asked, offering a rationale for the scenario's structure and logic. This capability for self-reflection allows the system to answer user questions about the scenario, offering valuable insights into its development and evolution throughout the process. By maintaining transparency and consistency, these features enhance user trust and facilitate effective collaboration between the user and the system.

5 Case Study

To evaluate the feasibility of the proposed approach, the system was tested in the context of an exemplary use case for military helicopter aviator training. This use case focuses on scenarios for Multi-Ship-Multi-Type (MS/MT) training capabilities within the Virtual Battle Space (VBS) simulation platform. MS/MT training concerns training of multiple pilots of different helicopter types to conduct exercises within a shared scenario, and is considered a desired training capability in modern military training.

In this use case, the user collaborates with the system to develop tailored training sessions for an Apache attack helicopter pilot and a Chinook transport helicopter pilot, addressing their individual readiness gaps in preparation for an upcoming mission. Two example interactions with the system are described. The first demonstrates an individual part-task training for the Chinook pilot (Sect. 5.1), while the second expands on this by showcasing a combined whole-task (mission) training for both pilots (Sect. 5.2). Throughout these interactions, the system is able to reference pilots' qualification profiles (introduced in Sect. 3.1).

5.1 Part-Task Training (Single-Pilot)

In this example, the user's intent is to develop a series of scenarios for part-task training of the Chinook pilot on Landing Zone operations (LZ-ops).

Step 1: Establish Training Designs. The initial prompts issued by the user are shown below. In the first prompt the user provides the initial request; in the second prompt the user requests an adaptation based on mission-specific information.

Prompt: *“Propose two training designs of different complexities to train a Chinook pilot on a LZ operation.”*

Prompt: *“Adjust the designs for an expected mission which takes place at night time in a mountainous desert environment; the mission is time-critical because of possible threats in the area.”*

In response to the user's prompts, the system generates a structured output containing training designs with progressive complexity. Table 1 illustrates the system's output after the second prompt. The table's format has been adjusted for readability purposes.

Table 1. Generated part-task training designs

Part-task Training: Landing Zone Operations		
Trainee	CH-47 Chinook pilot	
TTP	Landing Zone (LZ) Operations	
Level 1: Easy		
Title	Basic Night LZ Operations in Desert Mountains	
Objective	<p>Focuses on foundational skills for nighttime LZ operations in a desert, mountainous environment under controlled and secure conditions</p> <ul style="list-style-type: none"> • Practice basic night-time descent using night vision goggles (NVGs) • Focus on precision in hover and touchdown • Build familiarity with LZ marking systems and coordination with ground personnel 	
Complexity factors	Meteorological	Night time, clear skies, visibility >2 km, no wind
	Geographical	Flat open desert terrain surrounded by low mountains. LZ is large, unobstructed, and clearly marked
	Threats	No active threats present; environment is secure
	Contingencies	All systems functional, incl. GPS, radar and radio
	Cooperation	Ground team is well-prepared and responsive, marking the LZ with infrared strobes visible through NVG
Level 2: Advanced		
Title	Time-Critical Night LZ Operations in Desert Mountains with Simulated Threats	
Objective	<p>Replicates the full mission environment, emphasizing operational stress, degraded systems, and active threats</p> <ul style="list-style-type: none"> • Execute a swift, precise descent under time pressure • Manage environmental and operational challenges, including degraded systems and limited visibility • Coordinate landing in a confined, poorly marked LZ while under threat • Demonstrate quick decision-making in high-stress, time-critical conditions 	
Complexity factors	Meteorological	Moderate dust reducing visibility to 500–1000 m; crosswinds of 10–15 knots
	Geographical	Confined LZ in a mountainous desert valley; obstacles such as large rocks and ridges within 50 m of LZ
	Threats	Simulated enemy fire near the approach path, requiring evasive maneuvers and rapid landing
	Contingencies	Simulated GPS failure requiring reliance on NVG, radar, and manual navigation
	Cooperation	Ground team under simulated stress, providing incomplete or delayed LZ marking (e.g., partial strobes)

Based on the initial prompt, the system determines the specific trainee role (Chinook pilot) and training task (LZ-ops) and uses this to determine appropriate complexity factors, as defined in the role's qualification profile. The generated output illustrates several capabilities. First, the system is able to come up with logical values for individual complexity factors that correspond to different levels of complexity. Second, the system is able to define representative titles for each design, together with a summary of training objectives. Third, the system is able to adapt designs based on provided information about the mission context. Finally, the system is able to generate the designs according to a desired format. The desired format has been provided as a system prompt hidden from the user.

Step 2: Towards a Conceptual Scenario. After the training designs have been generated, the user requests the system to propose a conceptual scenario:

Prompt: *“Propose a conceptual scenario for training design number 2”*

Prompt: *“Adjust the scenario to take place in a village near the city of Kandahar”*

In the first prompt, the user requests a training design to be expanded to a conceptual scenario description. In the follow-up prompt the user requests an adaptation of the generated scenario by providing additional constraints on the mission area. In response, the system proposes a modified conceptual scenario as shown in Table 2.

To generate the outcome, the system uses the selected training design from the previous step in order to come up with a consistent and complete scenario. The generated output illustrates several capabilities. First, the system is able to ensure completeness of the scenario by enforcing the generation of all required details required for the subsequent generation of an executable scenario in step 3. The required elements have been specified as part of a system prompt hidden from the user. Second, the system is able to translate the more abstract values from the complexity factors into concrete values regarding e.g. representative locations, time of day, conditions, friendly actors (personnel, platforms, equipment, tasks) and enemy actors (personnel, weapons, locations, tactics). Finally, the system is able to adapt the scenario description based on additional constraints provided by the user (similar to step 1).

One of the risks during the translation from a training design to a conceptual scenario is that inconsistencies can be introduced due to requested adaptations of the user. For instance, the user may request a location (e.g. in a Nordic country) which does not conform to the geographical specification of the training design (e.g. mountainous desert). To resolve such inconsistencies, the system is instructed to verify whether the desired adaptation is consistent with previous outputs and notifies the user when it considers the adaptation to be inconsistent.

Step 3: Generation of an Executable Scenario. Based on the agreed upon scenario, the user instructs the system to generate an executable scenario. In the current system, an executable scenario is represented by a JSON specification, as this was found to be a reliable description language for demonstration purposes. To run the scenario in the simulation, a converter was developed to configure a scenario in the VBS platform based on the generated specification (see Fig. 3).

Table 2. Generated conceptual scenario.

Part-task scenario		
Complex, Time-Critical Night LZ Operations in Desert Mountains with Simulated Threats		
Scenario environment		
Location	A small village approximately 20 km west of Kandahar city, surrounded by arid flatlands and low rocky hills. The LZ is an open dirt field on the outskirts of the village, bordered by mud-brick compounds and narrow roads	
Time	02:30 local time (night operations)	
Weather	Clear skies, moderate dust stirred by rotor downwash, causing brownout conditions during landing. Light haze reducing visibility to 800m. Winds at 10–15 knots	
Scenario actors		
Trainee	Entity	1x CH-47 Chinook with defensive countermeasures (flares, chaff)
	Location	Approaching the LZ on a west-to-east flight path over the village outskirts
	Role	Conduct rapid descent and landing while managing brownout conditions
Friendly ground team	Entity	4x friendly forces soldiers with M4 carbine rifles
	Location	Positioned near the LZ on the western edge of the village
	Role	Secure and mark the LZ while defending against enemy fire
Enemy small arms team	Entity	4x enemy soldiers with AK-47 assault rifles and fragmentation grenades
	Location	Scattered in alleyways behind low walls near LZ, approximately 500 m away
	Role	Attempt to engage the Chinook during its final approach

In this last step, the system makes final translations of scenario elements to values required by the specification. For instance, it is able to generate the specific geo-coordinates for location descriptions such as ‘20km west of Kandahar city’. Specific system prompts are employed to ensure complete and valid values for individual fields. For instance, ensuring unique entity names, correct unit types (e.g. meters for altitude), force values (red/blue), or entity types that have to match supported types by the simulator. Currently, a limited set of scenario elements were supported to demonstrate the concept in VBS. Similar to the previous step, there are risks involved where translations may fail. For instance, the conceptual scenario may specify military platforms or weapon types that are not supported by the target simulation. One approach to prevent



Fig. 3. Generated executable scenario with a specification snippet (left) and the result in the simulation platform of Virtual Battlespace (right)

such problems is to provide mechanisms that would prevent the generation of unsupported elements early on in the scenario development process, by leveraging knowledge about the capabilities of the simulator.

5.2 Mission Training (Multi-pilot)

In the second example, the intent is to develop a mission scenario to train multiple pilots with personal training needs. Specifically, an aeromedical evacuation mission is considered where both a Chinook and Apache pilot are trained concurrently on individual part-tasks relevant for the mission. Based on the desired mission and part-tasks to be included in the mission, the user formulates the prompt:

Prompt: “Propose a training design to train a Chinook pilot and Apache pilot together in an Aeromedical Evacuation mission. The training should include a LZ operation task for the Chinook pilot, and a formation flying and close combat attack task for the Apache pilot.”

Similar to the previous example, the system generates a training design, though now including a mission type and two trainees with separate part-tasks (see Table 3). Furthermore, generated complexity factors (omitted in the table for brevity) now apply to the mission, which can be varied on user request to adjust mission complexity.

Following, upon user request, the system generates a conceptual scenario consistent with the training design. This process is similar to the single part-task scenario, but the inclusion of multiple part-tasks introduces additional scenario requirements, depending on the type of tasks. E.g. the formation flying task requires an additional (computer-controlled) Apache actor instantiated in the vicinity of the Apache trainee to fly in formation with; and the close combat attack task requires the presence of enemy ground forces engaging friendly ground forces to give reason for fire support during the mission. Finally, as the scenario concerns a mission training, mission-relevant fields are also generated, including a mission statement and mission phases, resulting in a complete training scenario (see Table 4).

Table 3. Generated mission training design.

Mission Training: Aeromedical Evacuation	
Trainees	CH-47 Chinook pilot AH-64D Apache pilot
Mission type	Aeromedical Evacuation
TTPs	CH-47 Chinook: <ul style="list-style-type: none"> • Landing Zone (LZ) Operations AH-64D Apache: <ul style="list-style-type: none"> • Formation Flying • Close Combat Attack
Objective	CH-47 Chinook: <ul style="list-style-type: none"> • Demonstrate the ability to safely extract personnel from a contested area AH-64D Apache: <ul style="list-style-type: none"> • Demonstrate the ability to provide effective escort for a MEDEVAC • Demonstrate the ability to perform a close combat attack • Demonstrate the ability to fly in close formation
...	...

Table 4. Generated conceptual scenario.

Conceptual scenario	
Aeromedical Evacuation with Enemy Opposition	
Scenario environment	
Location	A small village approximately 20 km west of Kandahar city, surrounded by arid flatlands and low rocky hills
Time	16:00 local time (day time operations)
Weather	Clear skies. Light haze reducing visibility to 800m. Winds at 10–15 knots
Mission statement	
A MEDEVAC mission to safely extract friendly ground forces from a contested Landing Zone (LZ). Two attack helicopters are utilized to provide escort for the transport helicopter and support ground troops	
Mission phases	
Ingress	The CH-47 Chinook and AH-64D Apache helicopters will approach the contested LZ, utilizing low level flying and tactical formations to avoid detection

(continued)

Table 4. (continued)

Conceptual scenario		
Aeromedical Evacuation with Enemy Opposition		
Evac	The CH-47 Chinook will land in the LZ and extract the friendly ground forces while the AH-64D Apache helicopters provide overwatch and close air support	
Egress	The CH-47 Chinook and AH-64D Apache helicopters will depart the mission area, utilizing mutual support and observation profiles to ensure safe exit from the contested area	
Scenario actors		
Chinook #1 (Trainee)	Entity	1x CH-47 Chinook with defensive countermeasures (flares, chaff)
	Location	Approaching the LZ on a west-to-east flight path over the village outskirts
	Role	Extract friendly ground forces from the determined LZ
Apache #1 (Trainee)	Entity	1x AH-64D helicopter equipped with 30mm M230 chain gun and 16 AGM-114 Hellfire missiles
	Location	Approaching the LZ with Chinook #1, flying in formation with Apache #2
	Role	Escort Chinook, secure LZ and provide fire support
Apache #2	Entity	1x AH-64D helicopter equipped with 30mm M230 chain gun and 16 AGM-114 Hellfire missiles
	Location	Approaching the LZ with Chinook #1, flying in formation with Apache #1
	Role	Escort Chinook, secure LZ and provide fire support
...

6 Discussion

In this study, a limited example was implemented to perform an initial qualitative analysis. Hereby we focused more on the scenario development process as a whole, rather than each phase of the process in-dept. Below we first discuss our experiences of employing LLMs to assist the scenario development process across different steps, as outlined in Sect. 3. Following, we reflect on the relation with AISs.

6.1 Lessons Learned

The scenario development process described in Sect. 3 was simulated as a single human-LLM interaction. The implemented use case shows that LLMs can provide valuable assistance throughout the process. However, the quality of the LLM assistance at each step depends on the reliability of the LLM model and its ability to successfully adapt to the domain via information retrieval.

While RAG has shown to be effective in providing the LLM with domain knowledge without resorting to expensive re-training of the LLM itself, domain adaptation through retrieval has a number of limitations. First, knowledge resources need to be prepared and indexed a priori into short passages of retrievable text, ideally expressing a concept or single line of thought; however, as documents often contain cross-references to other parts of the document or external sources, this indexing is challenging to achieve in an automated manner. Moreover, documents are often available in a variety of source formats, ranging from text documents to PDFs with embedded tables and figures, making parsing these resources a non-trivial task. The limitations of RAG are further exacerbated by the inherent architectural constraints of LLMs, specifically, the LLM's potential limited context window in which all information must be provided. This necessitates concise formulation of background knowledge and input instructions, which can be a challenge when significant domain information must be embedded inside the prompt. Alleviating the above issues remains a topic of active research, to which several solutions have been proposed, including knowledge editing [25] and alternative approaches to RAG [26].

While measures have been put in place to minimize hallucination of previously non-existent entities (i.e. platforms, weapons, etc.) throughout later phases of the scenario development process, it has shown challenging to ensure that LLM outputs in subsequent phases are consistent with previous stages, especially when the user proposes an adaptation that is inconsistent with the previously generated training design or conceptual scenario. Through self-reflection mechanisms – i.e. invoking the LLM to critique its own outputs with respect to the training designs or conceptual scenario generated at the previous stage – inconsistencies caused by hallucinations can be identified and corrected, improving the consistency of the output, however prolonging development time.

Lastly, as LLMs are pretrained on public, web-scraped datasets with limited access to simulator-specific documentation, LLMs have acquired a limited ability to generate scenarios in formal syntactic and semantic specifications or standards such as MSDL; as a result, an intermediate description language such as JSON was required to be used as a meta specification which must then be converted to the desired schema via manually programmed scripts or simulator plugins. A typical approach to alleviate this dependency is to provide the LLM an example or description of the desired output schema and use context-free grammars to constrain the output of the LLM to this schema [27, 28]; however, this would require manual labor to convert the schema to a grammar specification.

6.2 Adaptive Instructional Systems

Currently, the scenario development process is positioned as an isolated process. However, as a means for generating training content, it can be viewed in a broader scope as part of an AIS or data-driven learning (eco-)system. In the current presented use case examples, we assumed the user (e.g. instructor) to know the individual training needs of trainees, based on specific TTPs to train. In a data-driven system, these needs may be deduced from information about learners' readiness managed by learner profiles,

such as competency ratings. When an LLM can access such learner profiles, personalized training needs may be automatically determined, and training designs may be proposed. In this sense, LLMs can contribute to the implementation and orchestration of common components of an (adaptive) instructional system, including the domain model (access to qualification profiles), learner model (access to learner ratings), instructional model (developing personalized training designs), and interface model (setting up training scenarios) [29]. In addition, we expect LLMs to contribute to methods for (semi-) automated performance assessment by associating observed behavior with known performance metrics, hereby further empowering automation and adaptivity in instructional systems. In future work, explore these additions in the context of increased integration of LLMs in data-driven learning systems, while seeking alignment with current data strategy initiatives such as ADL's Total Learning Architecture (TLA) [30].

7 Conclusion

In this paper, we explored the potential of LLMs in collaborative scenario development for simulation-based military training. Conforming to current standards in scenario development, we presented a collaborative development method to explore three roles for LLMs across three steps: (1) as an SME assistant to support effective training design based on high-level needs, (2) as a creative designer for conceptual scenarios that are sound, complete, and grounded in instructional design, and (3) as a programmer for generating executable scenarios in the computational format of a target simulation system. At each step, the human user can collaborate with an LLM to fine-tune or adapt proposals made by the system. By breaking down the scenario generation process into three distinct steps – training design, concept scenario development, and scenario building – the system provides a structured end-to-end approach to producing scenarios. The intermediate text outputs generated at each step offer a promising avenue for future expansion, as they can be additionally leveraged to create supplementary (multimodal) materials such as visual aids, instructor/trainee handouts, or presentation slides for scenario briefings.

The method was implemented and demonstrated in a use case for helicopter pilot training, using representative training material in the form of available pilot qualification profiles, and a representative training system for military simulation (VBS). Preliminary results indicate that LLMs contribute to a more efficient and effective process. On the front end of the development process, they decrease the reliance on SMEs where organizational knowledge on the training domain can be captured and made easily accessible to instructors. On the back end of the process, they relieve the manual authoring burden for scenario builders where large parts of a scenario can be generated in the format of the target simulation system. Finally, LLMs support maintaining the integrity and consistency of scenarios throughout the process, decreasing the risk of additional iterations due to miscommunication, and making sure the resulting scenarios comply with their intended instructional purpose.

In this initial study, we focused primarily on the end-to-end workflow of scenario development to create initial scenario specifications. To this end, we did not yet address dynamic aspects of a scenario such as specifying storylines, events, or interactions for triggering learning experiences. Although including such aspects seems promising for

inclusion in conceptual scenarios (considering the capabilities of LLMs for interactive storytelling), currently there is little standardization in this area for executable scenarios and the implementation is still highly platform-specific. In future work, the aim is to address these aspects, while also considering in-session instructional interventions through real-time scenario adaptations.

References

1. Gerretsen, A., van Oijen, J.: Towards more effective and efficient tactical scenario generation. R. Neth. Aeros. Centre, TP-2017–175 (2017)
2. Hadi, M.U., et al.: Large language models: a comprehensive survey of its applications, challenges, limitations, and future prospects. *Authorea Preprints* (2024)
3. Rus, V.: Large language models and their implications for conversational tutors and GIFT. In: *Proceedings of the Twelfth Annual GIFT User Symposium*, pp. 77–86 (2024)
4. Zhu, Y., et al.: Large language models for information retrieval: a survey (2024). <https://arxiv.org/abs/2308.07107>
5. Manvi, R., Khanna, S., Mai, G., Burke, M., Lobell, D., Ermon, S.: GeoLLM: extracting geospatial knowledge from large language models (2024). <https://arxiv.org/abs/2310.06213>
6. Xie, Z., Cohn, T., Lau, J.H.: The next chapter: a study of large language models in storytelling (2023). <https://arxiv.org/abs/2301.09790>
7. Jiang, J., Wang, F., Shen, J., Kim, S., Kim, S.: A survey on large language models for code generation (2024). <https://arxiv.org/abs/2406.00515>
8. SISO, Guideline on Scenario Development for Simulation Environments, Simulation Interoperability Standards Organization, Orlando (2018)
9. IEEE, IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP, IEEE Std 1730–2022 (Revision of IEEE Std 1730–2010), pp. 1–74 (2022). <https://doi.org/10.1109/IEEESTD.2022.9919118>
10. SISO, SISO-STD-007–2008 Military Scenario Definition Language (2015)
11. Noori, N.S., Wang, Y., Comes, T., Schwarz, P., Lukosch, H.: Behind the scenes of scenario-based training: understanding scenario design and requirements in high-risk and uncertain environments. In: *International Conference on Information Systems for Crisis Response and Management* (2017). <https://api.semanticscholar.org/CorpusID:3688819>
12. Pedersen, C., Togelius, J., Yannakakis, G.N.: Modeling player experience for content creation. *IEEE Trans. Comput. Intell. AI in Games* **2**(1), 54–67 (2010). <https://doi.org/10.1109/TCIAIG.2010.2043950>
13. Smith, A., et al.: Automated scenario generation to support competency-based experiential learning in GIFT. In: *Proceedings of the Twelfth Annual GIFT User Symposium*, pp. 127–136 (2024)
14. Rowe, J., Smith, A., Pokorny, B., Mott, B., Lester, J.: Toward automated scenario generation with deep reinforcement learning in GIFT. In: *Proceedings of the Sixth Annual GIFT User Symposium*, pp. 65–74 (2018)
15. Core, M.G., Nye, B.D., Fegley, B.D.: Trend-aware scenario authoring: adapting training toward patterns from real operations. In: Sottolare, R.A., Schwarz, J., (eds.) *Adaptive Instructional Systems*, pp. 15–24. Springer Nature Switzerland, Cham (2024)
16. Stacy, W., Freeman, J.: Training objective packages: enhancing the effectiveness of experiential training. *Theor. Issues Ergon. Sci.* **17**(2), 149–168 (2016)
17. Zhao, W.X., et al.: A survey of large language models (2024). <https://arxiv.org/abs/2303.18223>

18. Wu, J., Gan, W., Chen, Z., Wan, S., Yu, P.S.: Multimodal large language models: a survey. In: 2023 IEEE International Conference on Big Data (BigData), pp. 2247–2256 (2023). <https://doi.org/10.1109/BigData59044.2023.10386743>
19. Mazzullo, E., Bulut, O., Wongvorachan, T., Tan, B.: Learning analytics in the era of large language models. *Analytics* 2(4), 877–898 (2023)
20. Dimitriadis, Y., Martínez-Maldonado, R., Wiley, K.: Human-centered design principles for actionable learning analytics. In: Tsiatsos, T., Demetriadis, S., Mikropoulos, A., Dagdilelis, V., (eds.) *Research on E-Learning and ICT in Education: Technological, Pedagogical and Instructional Perspectives*, pp. 277–296. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-64363-8_15
21. Allen, B.P., Stork, L., Groth, P.: Knowledge engineering using large language models (2023)
22. Lu, R.-S., Lin, C.-C., Tsao, H.-Y.: Empowering large language models to leverage domain-specific knowledge in e-learning. *Appl. Sci.* 14(12) (2024). <https://doi.org/10.3390/app14125264>
23. SISO, Enumerations for simulation interoperability - SISO-REF-010 v33 (2024)
24. Gao, Y., et al.: Retrieval-augmented generation for large language models: a survey (2024). <https://arxiv.org/abs/2312.10997>
25. Wang, S., Zhu, Y., Liu, H., Zheng, Z., Chen, C., Li, J.: Knowledge editing for large language models: a survey. 57(3) (2024). <https://doi.org/10.1145/3698590>
26. Khan, A.A., Hasan, M.T., Kemell, K.K., Rasku, J., Abrahamsson, P.: Developing retrieval augmented generation (RAG) based LLM systems from PDFs: an experience report (2024). <https://arxiv.org/abs/2410.15944>
27. Netz, L., Reimer, J., Rumpe, B.: Using grammar masking to ensure syntactic validity in LLM-based modeling tasks. In: *MODELS Companion 2024*, pp. 115–122. Association for Computing Machinery, New York (2024). <https://doi.org/10.1145/3652620.3687805>
28. Dong, Y., et al.: XGrammar: flexible and efficient structured generation engine for large language models (2024). <https://arxiv.org/abs/2411.15100>
29. Sottolare, R.: Understanding the AIS problem space. In: *Proceedings of the 2nd Adaptive Instructional Systems (AIS) Standards Workshop* (2019)
30. Smith, B., Milham, L.: Total learning architecture (TLA) data pillars and their applicability to adaptive instructional systems. In: *HCI International 2021-Late Breaking Papers: Cognition, Inclusion, Learning, and Culture: 23rd HCI International Conference, HCII 2021, Virtual Event, 24–29 July 2021, Proceedings 23*, pp. 90–106. Springer (2021)