Systems Engineering Language Modeling Assistant (SELMA)

Jyotirmay Gadewadikar, Tomi Esho, Jeremy Marshall

MITRE | SOLVING PROBLEMS FOR A SAFER WORLD®

Motivation for Generative Systems Engineering

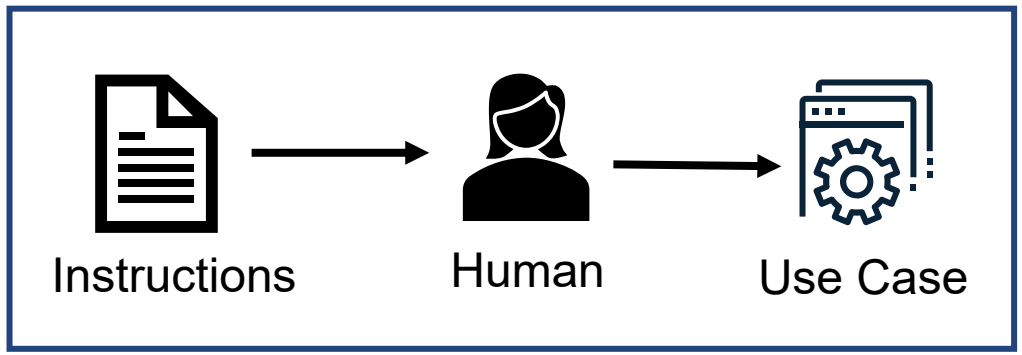
- Generative Systems Engineering uses natural language processing to support Systems Engineering and generate / refine requirements, architecture, code, testing approaches etc..

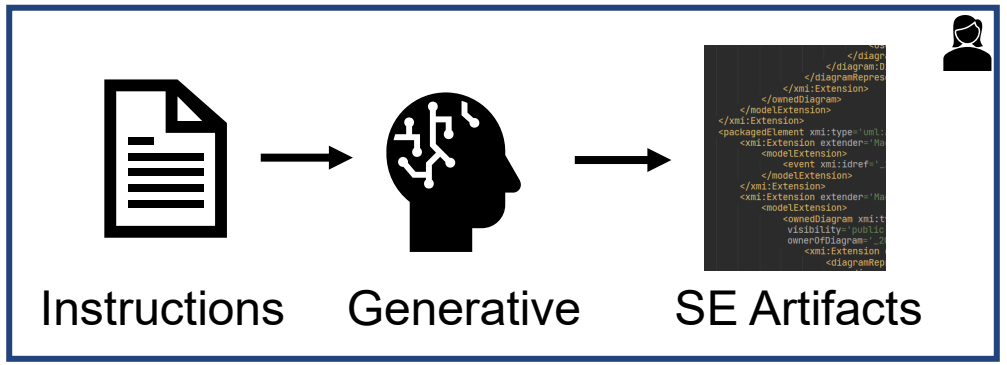
As-Is: Humans are creating artifacts which are used in the SE process, testing/using those, manual process of code/artifact generation → putting that into a tool, testing it, supporting the underlying use case.

- Slow
- Manual
- Error prone
- Focused on coding

To-Be: Humans can use NLP to create artifacts and focusing on how to use SE to support the underlying use case.

- Fast
- Semi-automatic
- Reduced errors
- Focused on core mission





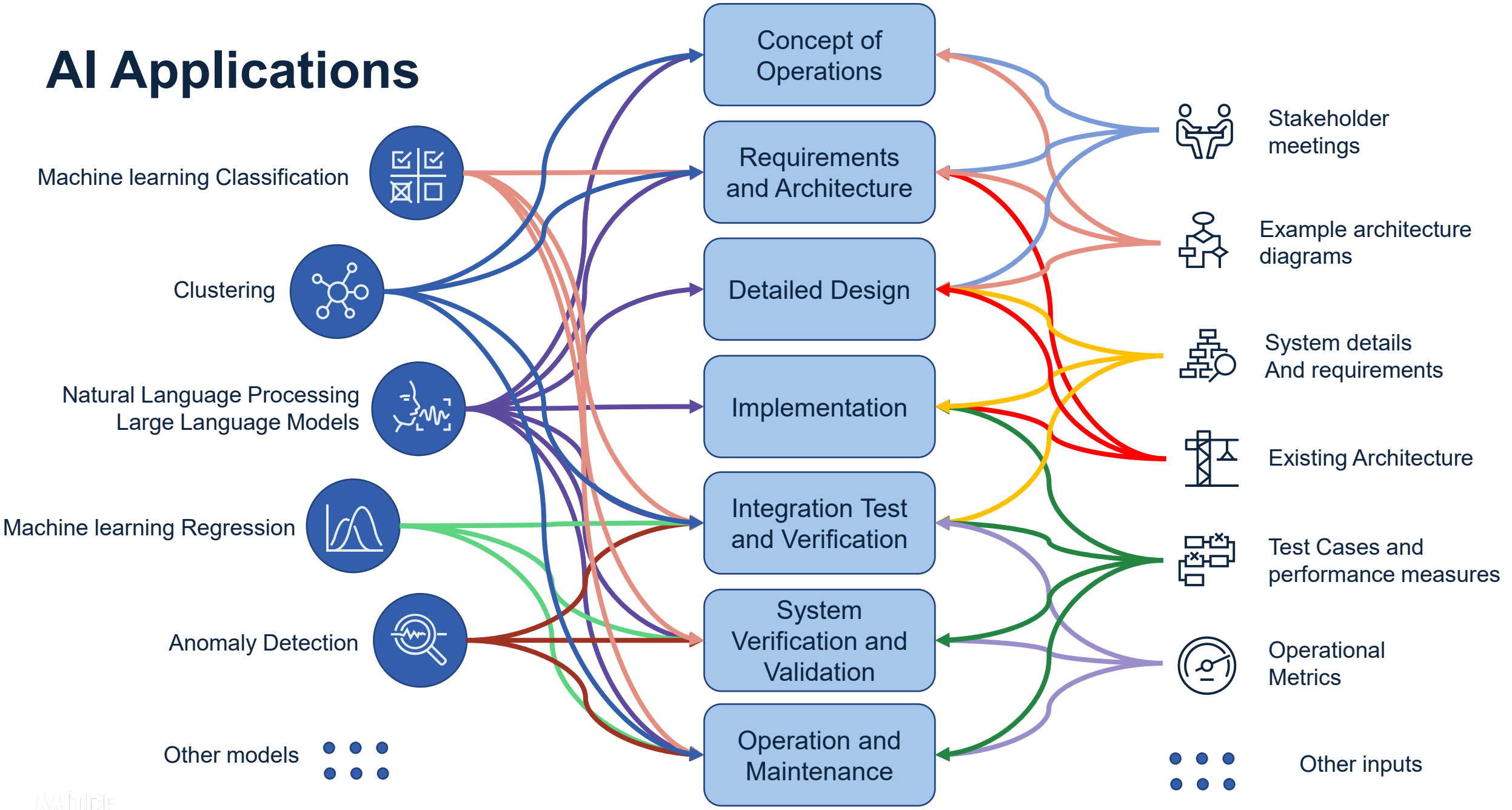
MITRE

AI Opportunities in SE lifecycle

Lifecycle	Activities	Pain Points	Where AI can help
Concept of Operations	- Define system objectives and scope - Identify stakeholders and their needs - Develop operational scenarios - Draft Concept of Operations document	- Difficulty in gathering comprehensive stakeholder requirements - Ambiguity and inconsistency in defining operational scenario - Time-consuming document drafting and review process	**- Initial Data Analysis:** AI can analyze stakeholder data and operational scenarios to identify and validate system objectives. **- Concept of Operations Generation:** AI-powered tools can assist in drafting, reviewing, and ensuring consistency in the Concept of Operations document or identify and categorize initiatives and themes
Requirements and Architecture	- Elicit and document requirements - Develop system architecture - Perform trade-off analysis - Validate requirements and architecture	- Incomplete or conflicting requirements - Difficulty in prioritizing requirements - Time-consuming trade-off analysis	**- Requirements Elicitation and Generation:** AI can generate and analyze stakeholder inputs and historical data to gather and prioritize requirements. **- Automated Consistency Checking:** AI can identify inconsistencies, redundancies, and conflicts in requirements. **- Architecture Creation:** AI algorithms can aid in generation and evaluation of multiple design alternatives and find the best system architecture.
Detailed Design	- Develop detailed design specifications - Create design models and diagrams - Review and validate designs - Selection of tools and products	- Complexity in translating high-level requirements into detailed designs - Time-consuming design validation process - Risk of design errors	**- Design Automation:** AI-driven tools can automate the creation of schematics or code from high-level specifications. **- Design Validation:** AI can validate designs against requirements and constraints, identifying potential issues early. **- Tool Selection:** AI can assist in identifying a list of tools/products and vendors suited for the task
Implementation	- Develop and integrate system components - Write and review code - Perform unit testing	- Manual coding errors - Time-consuming code reviews - Incomplete unit testing	**- Code Generation:** AI can assist in code generation from models or specifications, reducing manual coding effort and errors. **- Unit Testing Support:** AI-driven static and dynamic analysis tools can help identify bugs, security vulnerabilities, and performance bottlenecks.
Integration Test and Verification	- Integrate system components - Develop and execute test cases - Analyze test results	- Integration issues due to component incompatibility - Time-consuming test case development - Difficulty in analyzing large volumes of test data	**- Automated Testing:** AI can assist in the creation and execution of test cases, analyzing results, and identifying areas needing further testing. **- Fault Detection:** Machine learning algorithms can detect anomalies and predict potential integration issues.
System Verification and Validation	- Verify system against requirements - Validate system performance in real-world scenarios - Document verification and validation results	- Ensuring comprehensive verification and validation - Difficulty in simulating real-world scenarios - Time-consuming documentation process	**- AI-driven Verification and Validation:** AI can assist in verifying and validating subsystems outputs by comparing it against requirements/architecture and design **- Simulation and Emulation:** AI can enhance simulation tools to test the system under various conditions and scenarios.
Operation and Maintenance	- Monitor system performance - Perform maintenance and updates - Provide user support	- Unplanned system downtime - Difficulty in identifying performance bottlenecks - Time-consuming user support	**- System Maintenance:** AI can predict component failures, allowing for proactive system maintenance and upgrades. **- Performance Monitoring:** AI can continuously monitor system performance, identifying inefficiencies and suggesting modifications. **- User Support:** AI-driven chatbots and virtual assistants can provide real-time support to users.

MITRE

AI Applications



SELMA – Overview

- SELMA leverages generative artificial intelligence to automate labor-intensive and time-consuming MBSE workflows
- This text-to-model prototype functions as a text interaction-based tool, allowing users to provide natural language instructions which are then converted to SE artifacts
- The conversation format allows users to make iterative changes to the model, enhancing flexibility and efficiency

Technology Stack

LLM Hosts

AI/ML @ MITRE

OpenAI

Sources models from

HUGGING FACE

Local LLM server

Flask

MITRE GitLab

- Repository
- Version Control

IDEs

PC

IJ

Programming Languages

Jython

Java

Model Based Systems Engineering Tools

CATIA | No Magic

Text-to-model process flow

Hello, how can I help you today?

I am looking to create a block definition diagram which displays the blocks of a Helicopter System …

MBSE Tool Graphic User Interface initiated

Text request through MBSE graphic user interface

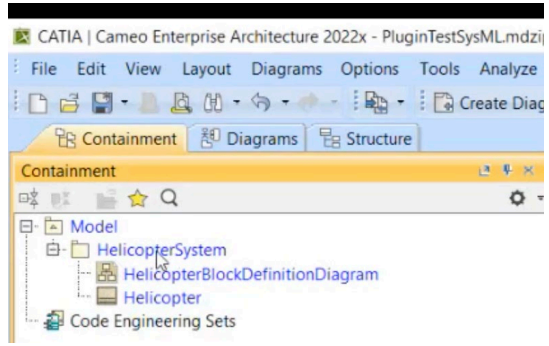
API call to LLM to process request

Functions such as create, modify, delete…

LLM identifies the need to call one or more functions

LLM returns function names and arguments

First response



Function names and arguments are used to create models programmatically through MBSE tool APIs

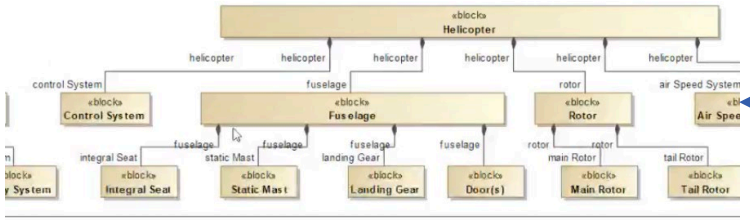
MBSE assistant response

I have successfully created….

Model is shown in the MBSE tool and inspected by the Systems Engineer, who can make additional requests or ask for modifications

Now we want to add the sub-systems to the model, these are sub-systems of the Helicopter element. These sub-systems are as follows: “Powertrain”…

Updated model



Core Technical Components

Input & Preparation

Function List Setup

Provide/Prepare LLM with list of available functions to call, along with their definitions

Prompt Engineering

Apply prompt engineering techniques such as few-shot prompting to ensure consistent and correct responses

Processing

API Interaction with LLM

API calls to LLMs (Llama3, Mixtral, GPT) with Python

Context maintenance

Each action is appended to the conversation history to provide context for subsequent actions

Execution

Communication

JSON-formatted LLM responses from Python script are sent to Java via HTTP requests with Flask

Function Execution

Functions are parsed and executed within Java MBSE API scripts

Optimization

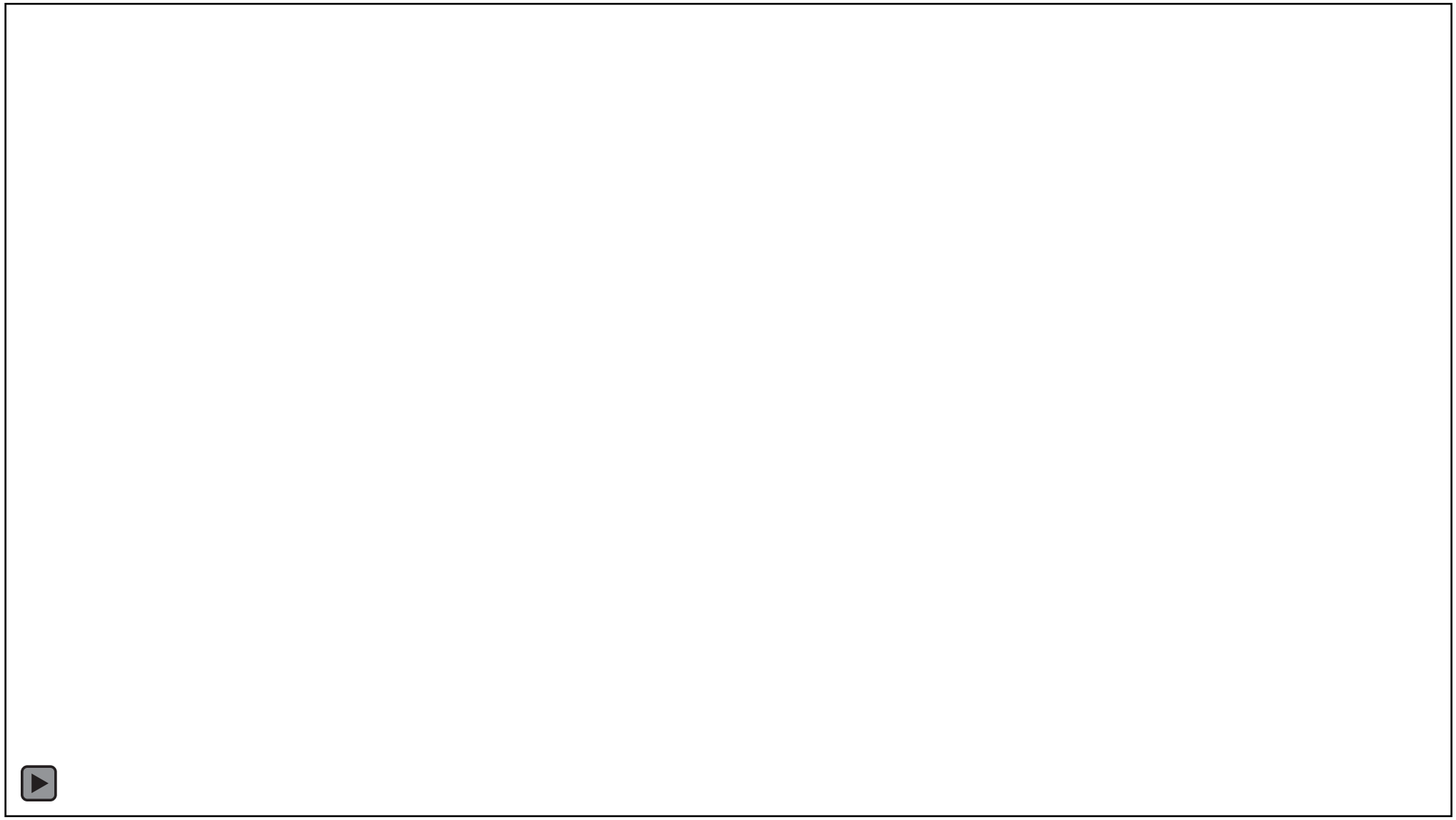
Token limitations

Sentence Transformers for semantic similarity ranking / information retrieval

Parallel function calling

Process multiple functions in response to single user input

Video demonstration: Helicopter System



MITRE

Conclusion

Challenges and Opportunities

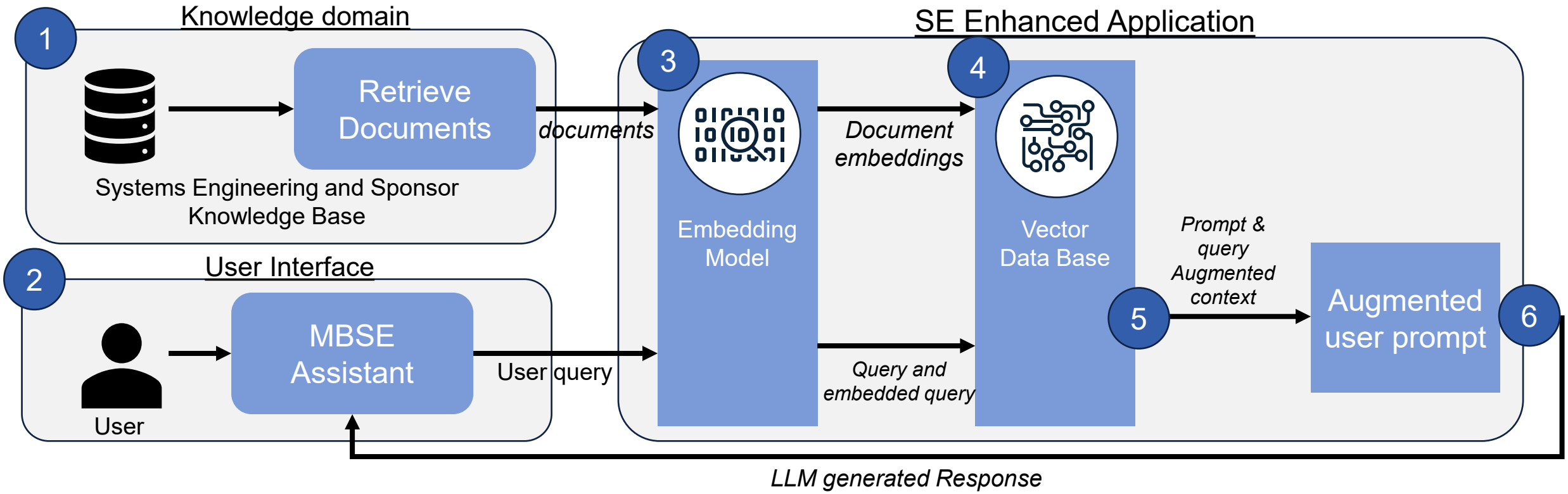
LLMs are not specialized in Systems Engineering, specifically Model Based Systems Engineering

Languages such as SysML alone are quite large and the model needs guide rails to ensure reliability

Developing a Generalized approach requires extensive planning and testing to identify the best trade off between the pros and cons of performance, token efficiency and scalability

Developing standards to evaluate LLMs in Systems Engineering Use Cases

Future State



1. System Engineering, Domain, and Approved Sponsor Information are stored to be used in vector database with dense embeddings.
2. User prompts MBSE Assistant with a request
3. Documents and user prompt embedded into vectors
4. Embedding of user prompt and compared against passage embeddings. Most similar passages are retrieved and augmented to the user prompt.
5. User prompt is augmented with contextual information and fed into LLM.
6. Response is parsed, and appropriate APIs are invoked to generate the artifact

Contact

Jyotirmay Gadewadikar, Chief Scientist – AI Integration and Systems Engineering

jgadewadikar@mitre.org

@JyoGadewadikar

https://www.linkedin.com/in/JyoMIT

MITRE | SOLVING PROBLEMS FOR A SAFER WORLD®