Digital Engineering: MBSE Quick and Easy with AI

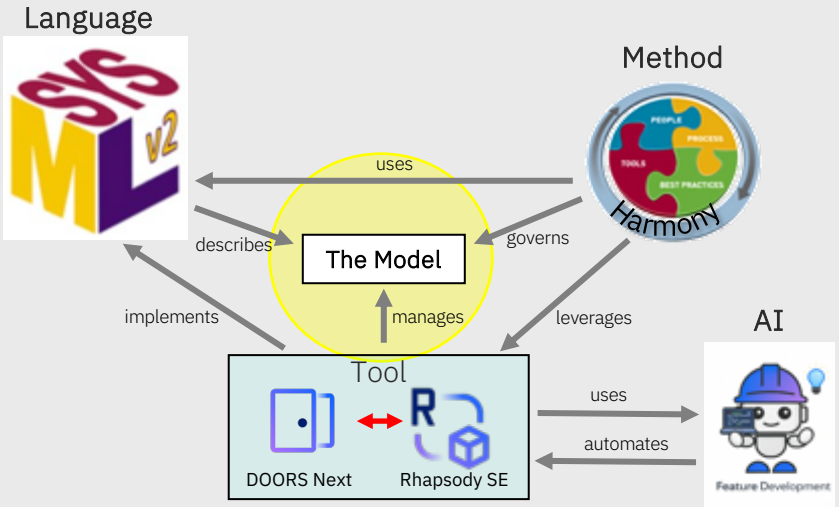
think

Peter Schedl
Program Manager
IBM Engineering Lifecycle Management
peter.schedl@de.ibm.com

Executive Summary -
Be Smart use AI wisely orchestrated in Engineering

Engineering requires models defined in Model-based Systems Engineering (MBSE).

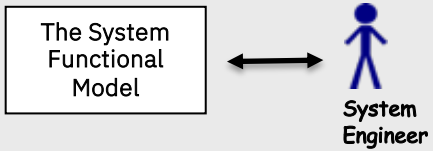
MBSE consists of Language + Method + Tools **enhanced by AI**



What is MBSE

Model-Based Systems Engineering (MBSE) is a methodology that uses **formal models** as the primary means of information exchange and system development, replacing traditional document-centric approaches

...Or...
A visual approach to understanding **requirements** and realizing them into a robust **system design**



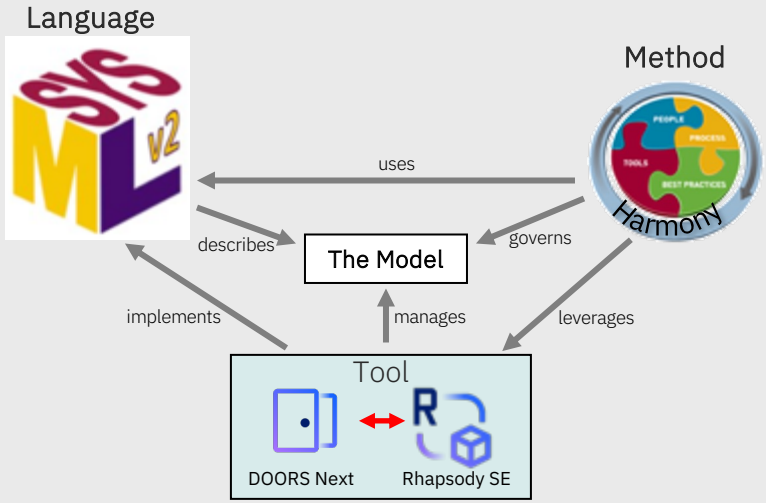
INCOSE Vision 2035: The Future of Systems Engineering is **Predominantly Model-Based**

INCOSE = International Council on Systems Engineering

MBSE Modeling Solution

MBSE is a Systems Engineering practice that incorporates:
- Modeling language
- Modeling method
- Modeling + Requirements Mgt tool

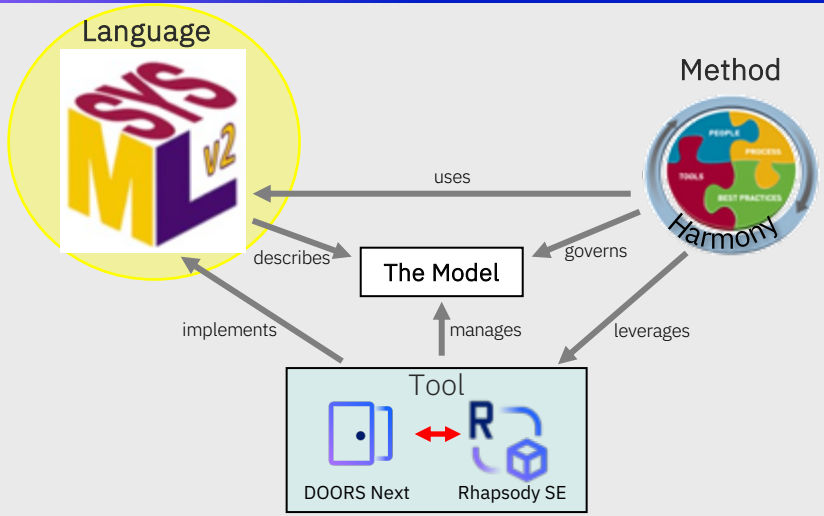
MBSE transforms system development by
- Providing a **single source of truth** for all stakeholders
- Enabling early **validation** and error detection via standard, **formal notation** (SysML)
- Improving **communication** through visual models
- **Automating** repetitive tasks and documentation
- Ensuring **traceability** throughout the lifecycle
- **Managing complexity** in large-scale systems



MBSE is not just a tool –
It's a fundamental shift in how we think about and develop complex systems

There has been an industry-wide call for a more comprehensive MBSE standard

Requires NEW:
- Modeling language – SysML v2
- Modeling method
- Modeling tool



SysML v2 Key Enhancements
- For System Engineers (not based on UML/SysML v1)
- Well defined Graphical & Textual **language semantics**
- **Integration API**
- **Extensibility** to Domain Specific Languages
- Greater coverage of systems engineering activities (e.g. geometry)

AI in Product Engineering?

Engineering teams face increasing complexity

- More software-defined products with cross-domain dependencies (mechanic, electric, ...)
- Faster release cycles with stricter compliance
- Large volumes of requirements, models, interfaces, variants and test artifacts

AI advantages

- Faster analysis of large engineering datasets
- Better traceability across lifecycle artifacts
- Higher automation in repetitive knowledge work
- Improved consistency in models, code and documentation
- Decision support for architects and systems engineers

AI for Engineering Challenge

Engineering as high trust domain **vs** GenAI as probabilistic technology

Approach for GenAI supported teams

– Agents augment tasks

– Orchestration and stepwise automation across workflows

– Human-centered decisions

The Goal: From Requirements to Models to Code (and back)

– Requirements analyzed by AI

– Candidate SysML v2 MBSE models generated stepwise

– Models reviewed and approved by engineers stepwise

– Code and interfaces generated from models

– Tests and documentation generated from the same source

– Feedback from implementation refines model and requirements

AI enables continuity across engineering artifacts

SysML v2 Textual vs API: Two complementary ways to work

Textual

- Human-readable and version-control friendly
- Easy to diff and merge
- Good for AI-assisted generation

– Textual is good for AI authoring

API

- Programmatic access to model elements and relationships
- Enables integration with engineering toolchains
- Supports automation, synchronization, validation and analytics
- Better for scalable system-to-system workflows

– API is ideal for automation and enterprise integration

ChatBot, MCP and Agents

ChatBot
- Conversational interface
- Good for Q&A, summarization and simple assistance
- Usually reactive and **user-prompt driven**

MCP
- Model Context Protocol for **standardized tool access**
- Connects models to tools, files, APIs and external systems
- Gives LLM-based solutions reliable context and action capabilities

Agents
- Goal-oriented software entities
- Can plan, reason, call tools, coordinate tasks and **act autonomously**
- Useful for multi-step engineering workflows

Examples
- “Explain this requirement”
- “Retrieve related model”
- “Analyze requirement, update model and generate code stub” stepwise orchestrated

Target Operating Model for AI in Engineering – Human-in-the-loop

- **AI proposes** requirements interpretations, incremental models and code
- **Engineers approve** safety-critical and architecture-relevant decisions
- Toolchain **integration via MCP** and APIs
- **MBSE backbone** ensures consistency and structure

Operating principles

- Separate proposal from doing / approval
- Log all AI-generated changes
- Validate against engineering rules and standards

AI for Engineering Examples

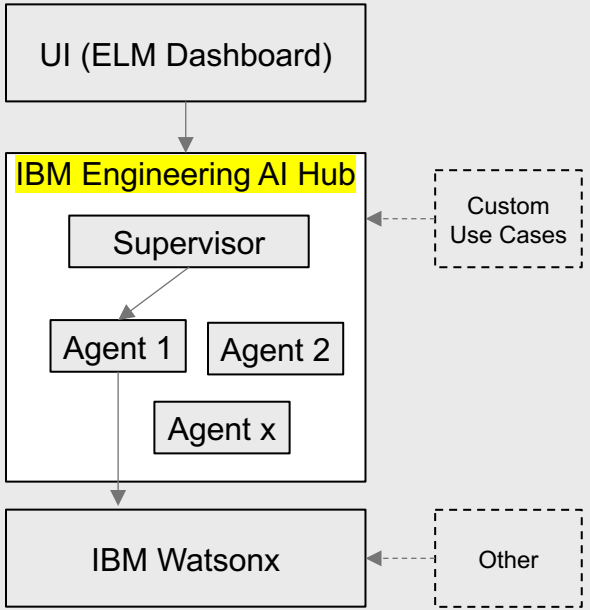
AI for Engineering Example – IBM Engineering AI Hub utilizing Agents

AI agents to improve clarity and speed in complex, mission-critical systems and software

- Automate repetitive, error-prone tasks to focus on high-value work
- Outputs grounded in your engineering data
- AI agents propose; Engineers review and confirm
- Designed to grow with your needs

Actual Agents / Use Cases

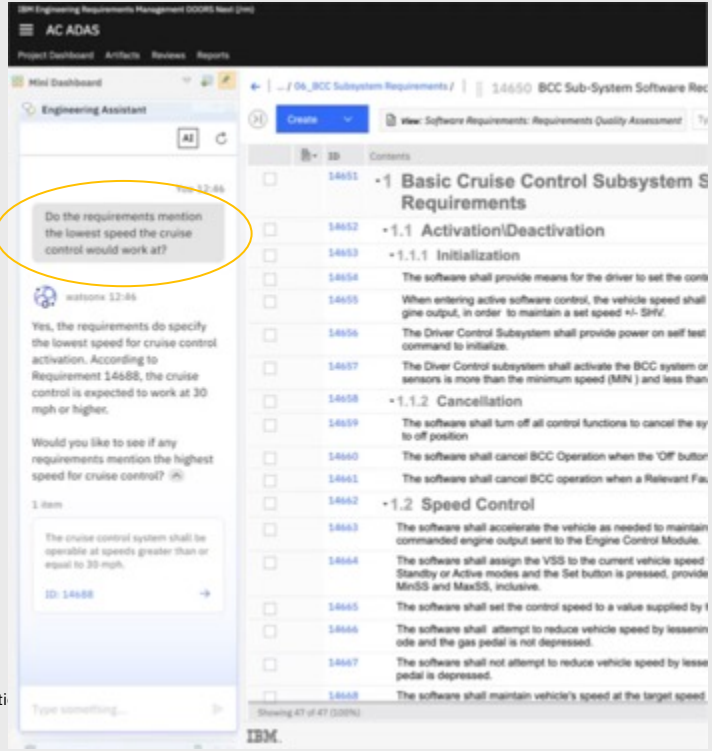
- Engineering Assistant
- Requirements Quality Analysis & Recommendation
- Work Item Synopsis & Work Item Compose
- **Actors and Use Case Discovery**



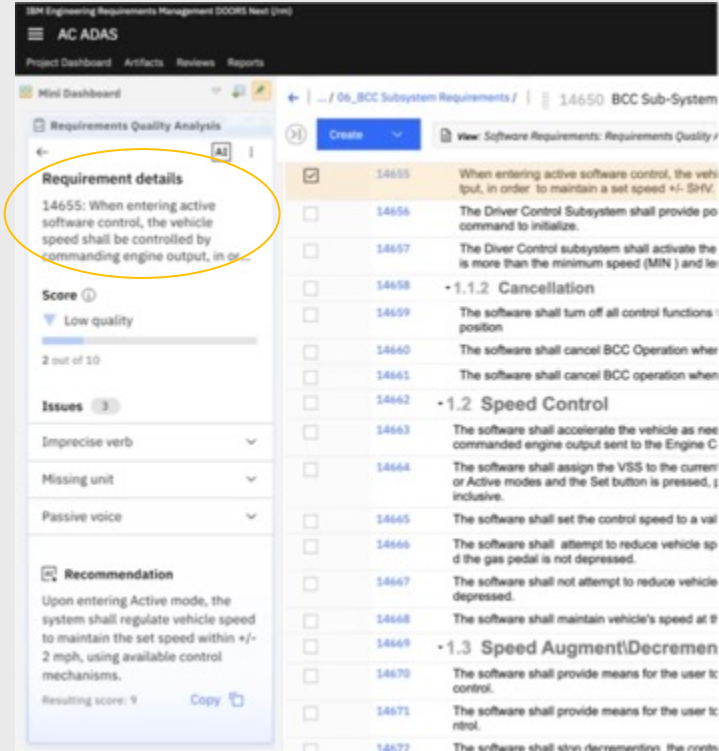
Engineering AI Hub Announcement

AI for Engineering Examples – Engineering AI Hub w DOORS Next

Engineering Assistent

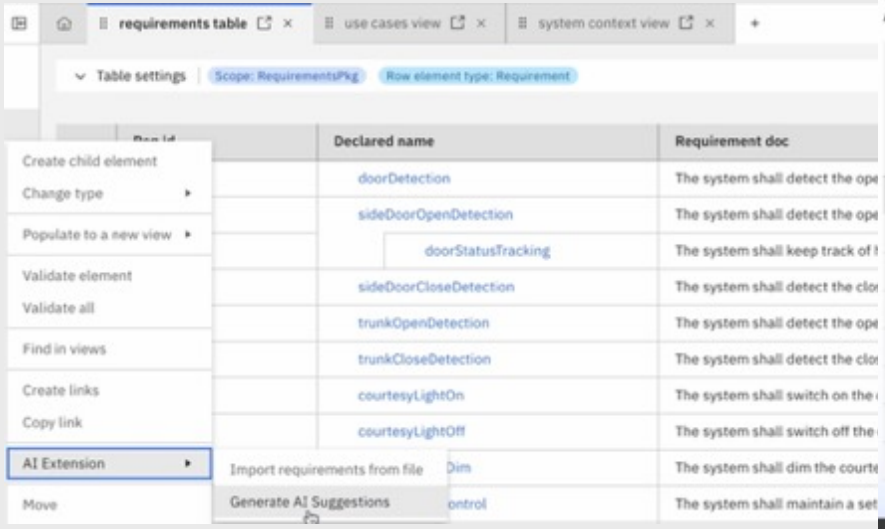


Requirements Analysis

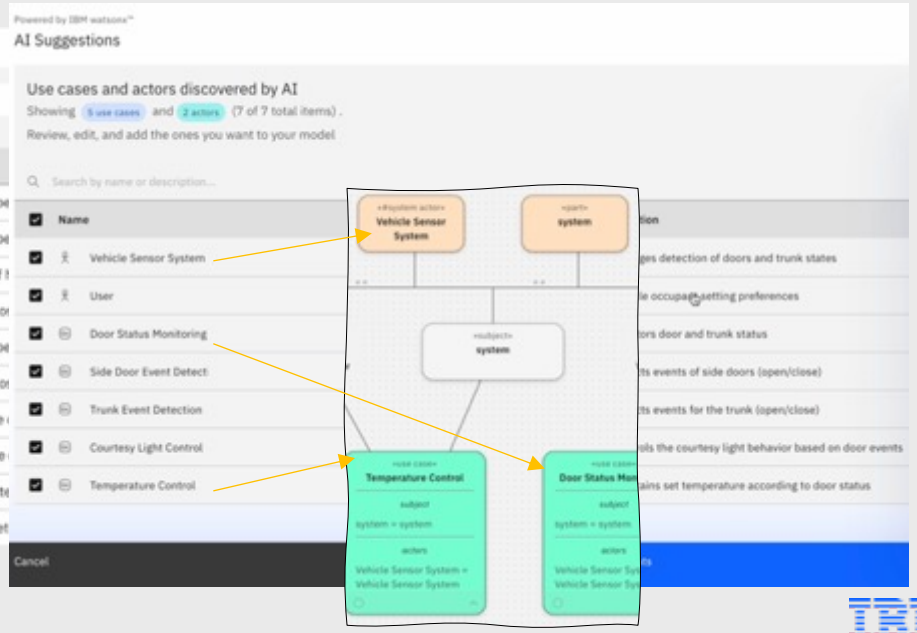


AI for Engineering Examples – “Identify use cases and their system actors” w Rhapsody SE

from textual requirements



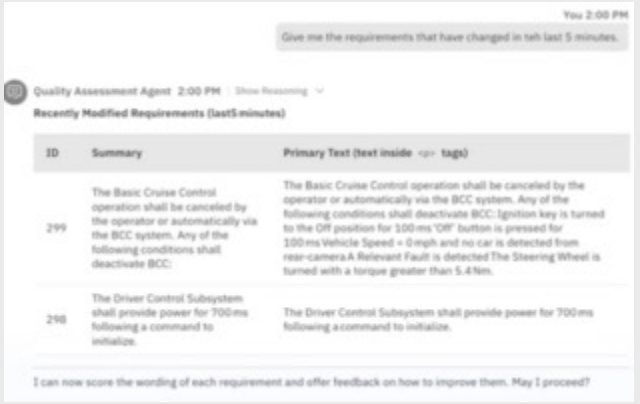
generate Use Cases and Actors



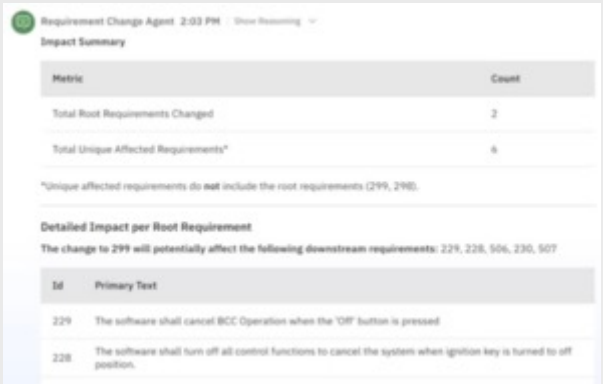
AI for Engineering Examples – Engineering AI Hub Extended (A2A) Examples

User creates extended uses cases combining existing agents and MCP functionalities

– Quality Analyze recently changed requirements (i.e. time or changeset)

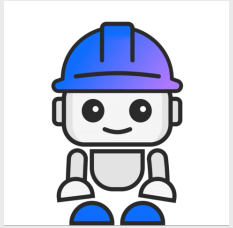


– Impact analysis on recently changed requirements



AI for Engineering - The complete Software Development Lifecycle in the Age of AI with IBM Bob

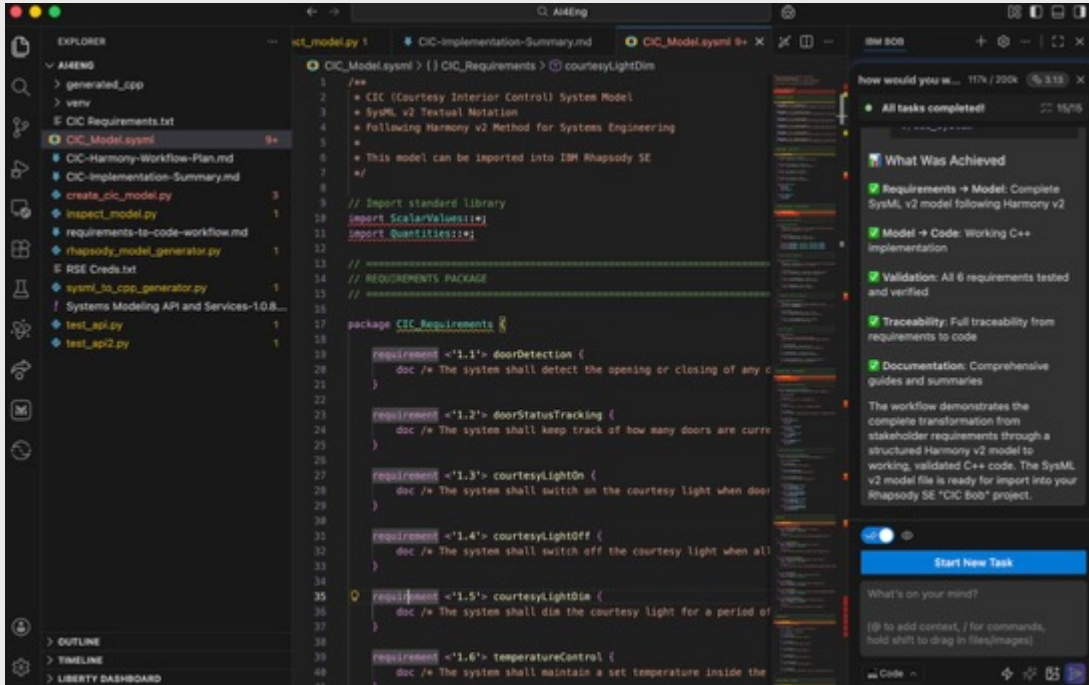
Hi, I'm Bob!



I'm more than a code assistant

- Modes & Skills
- MCP Library
- Diagrams
- Security Findings & Resolution
- Documentation
- DevOps & Automation

I can help you to



https://bob.ibm.com

AI for Engineering Examples –
IBM Bob used to build SysML v2 Model from Requirements

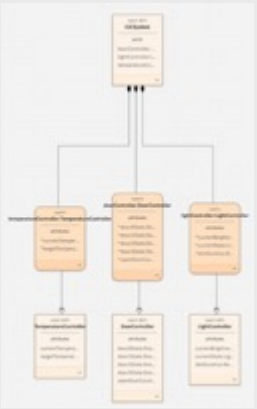
Input: 6 CIC Requirements

Generates SysML v2 textual notation

Imported in Rhapsody SE

	Declared name	Requirement doc
1	courtesyLightDim	The system shall dim the courtesy light for a period of 30 seconds before...
2	courtesyLightOff	The system shall switch off the courtesy light when all doors are closed.
3	courtesyLightOn	The system shall switch on the courtesy light when door is opened.
4	doorDetection	The system shall detect the opening or closing of any door on the vehicle.
5	doorStatusTracking	The system shall keep track of how many doors are currently open.
6	temperatureControl	The system shall maintain a set temperature inside the vehicle.

Views Populated



AI for Engineering Examples – IBM Bob to build Rhapsody SE Extensions

Task

Gwnerate an Rhapsody SE extension which reads the type Use Case lines including the used columns from the file UCD.xlsx. For example the Primary Text shall be the name. Generates these as use cases in Rhapsody SE with the approriate attributes.

Todo List 11/12

- Analyze UCD.xlsx structure and identif...
- Set up new extension project structure ...
- Install required dependencies (xlsx libr...
- Create Excel file reader utility to parse ...
- Implement data extraction logic for Use...
- Create mapping between Excel column...
- Implement Use Case creation logic usin...
- Create user action to trigger import fro...
- Add error handling and validation
- Add user notifications for import progr...
- Test the extension with sample data
- Document the extension usage and dep...

UCD Excel Importer Extension - Implementation Complete ✅

I have successfully created a complete Rhapsody SE extension that reads Use Case data from UCD.xlsx files and generates Use Cases in Rhapsody SE with appropriate attributes.

📦 What Was Created

Core Implementation Files

1. **src/types.ts** (82 lines) - TypeScript interfaces for Use Case data, Excel rows, and results
2. **src/excel-reader.ts** (186 lines) - Excel file parsing, validation, and data extraction
3. **src/use-case-creator.ts** (189 lines) - Use Case creation in Rhapsody SE model with RSE API
4. **src/extension-actions.ts** (128 lines) - User action registration and import orchestration

✨ Key Features

Excel File Processing

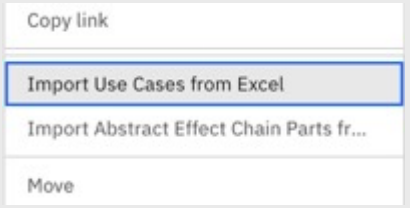
- Parses .xlsx files using the xlsx library
- Extracts rows where "Artifact Type" = "Use Case"
- Maps 10+ Excel columns to Use Case attributes
- Validates file structure and data integrity

Use Case Creation

- Creates UseCase elements in Rhapsody SE model

Status: ✅ Implementation Complete
Build: ✅ Successful (321 KB bundle)
Documentation: ✅ Comprehensive (6 guides, 2,562 lines)
Code: ✅ Production-ready (618 lines TypeScript)
Ready for: Deployment and Testing

Generated Use Case Importer:



Rhapsody SE UI

Key Takeaways

AI is a practical accelerator for engineering products

- AI reduces effort in requirements analysis, modeling and coding
- **AI + MBSE** creates a strong foundation for structured automation
- **SysML v2** textual and API approaches are complementary
- MCP enables reliable access to tools and engineering context
- Agents extend beyond chat into **step-by-step engineering workflows**
- Highest value comes from human-governed, traceable, end-to-end integration

The future is not just AI-assisted chat -
It is AI-enabled engineering workflows across requirements, models, code and validation

Thank You

Further information:

IBM Engineering Lifecycle Management Automotive Compliance
IBM Engineering Lifecycle Management Overview
IBM Rhapsody Systems Engineering
IBM DevOps Loop