ISTANBUL SOFTWARE TESTING CONFERENCE

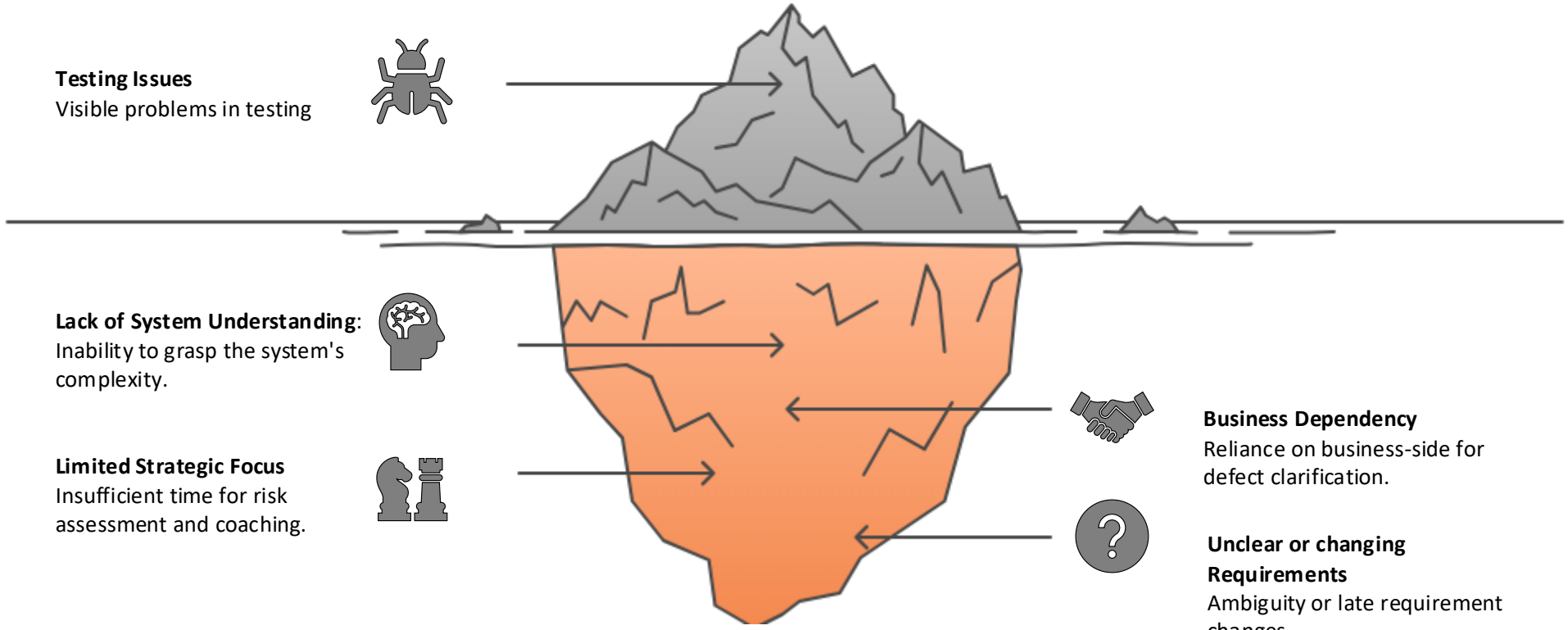
Fusion Forward: Let AI Be Your Partner, Not Your Master

Elanur Coskun and Anastasia Simou
22. May 2026

#ISTC2026

Challenges in Testing Environment

Testing Issues
Visible problems in testing



Lack of System Understanding:
Inability to grasp the system's complexity.

Limited Strategic Focus
Insufficient time for risk assessment and coaching.

Business Dependency
Reliance on business-side for defect clarification.

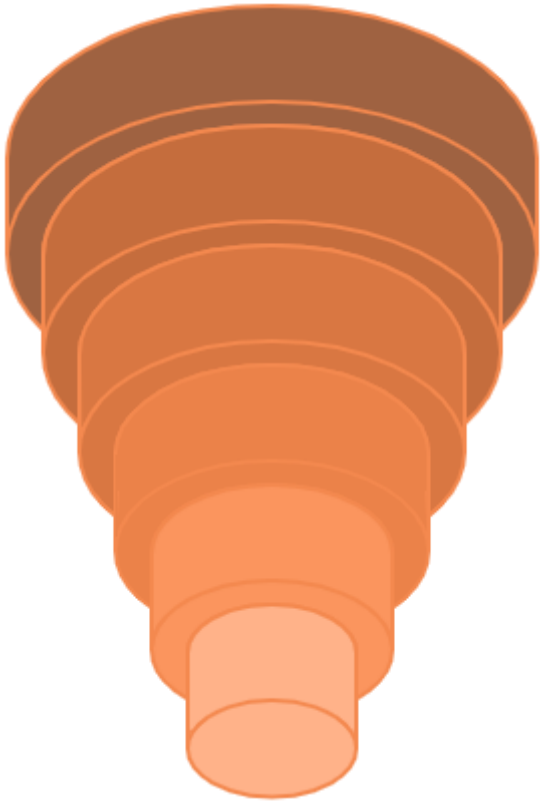
Unclear or changing Requirements
Ambiguity or late requirement changes

AI as an enabler of even better testing

Defect Management: AI creates, identifies duplicates, and categorizes defects

Automated Test Case Creation: AI converts manual test cases to automated

Code Exploration: AI checks the quality of automated scripts



Manual Test Case Creation: AI generates manual test cases

Test Case Evaluation: AI assesses the quality of test cases

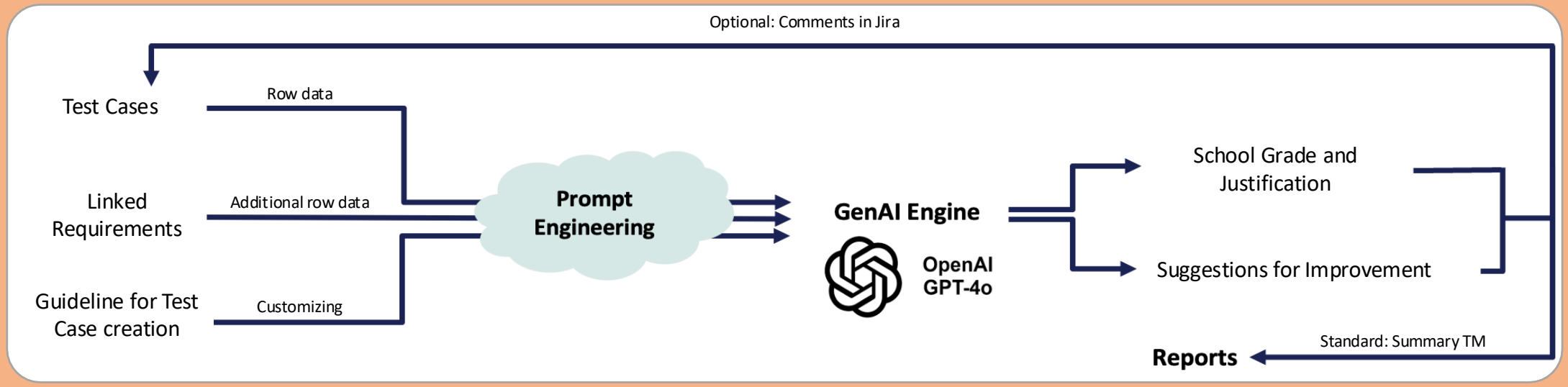
PR Analysis: AI analyzes pull requests before merging

AI Use Cases – Architecture 1

Enhanced Validation of Test Cases Using GenAI for a Banking Client with SAP S/4HANA Application. GenAI was employed to validate the content and formal aspects of test cases in JIRA/Xray, improving system testing quality. The process provided standardized evaluations, fast automated feedback, and continuous optimization.

Advantages of using GenAI:

- Uniform and standardized evaluation of test cases
- Fast and systematic feedback through automated processing
- Continuous quality improvement of test cases through regular review and optimization



AI Use Case 1 | AI enables Automated AI Review of Test Cases

1

Note (Quality Category: 4)

- **Granularity:** The test step contains multiple actions that should be split into separate steps to improve granularity.
- **Wording:** The wording is partly unclear and could be more precise.
- **Test Objective:** The expected result is not described in sufficient detail.
- **Completeness:** Preconditions and specific test data are missing.
- **Requirements Coverage:** The requirements from the linked user stories are only partially covered.

3

Summary

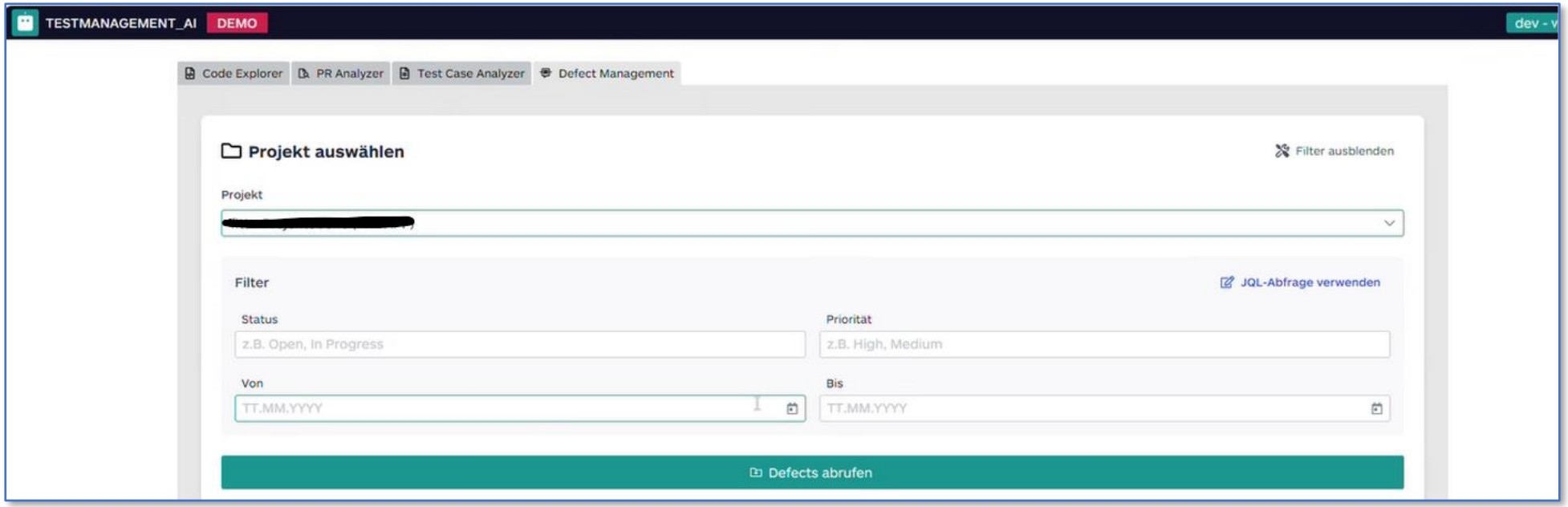
- **Preconditions:** Add specific preconditions, e.g., "The system must be initialized with the relevant test data."
- **Test Steps:** Split the test step into several steps to improve granularity and make the wording more precise.
- **Requirements Coverage:** Add specific test steps that fully cover the requirements from the user stories.

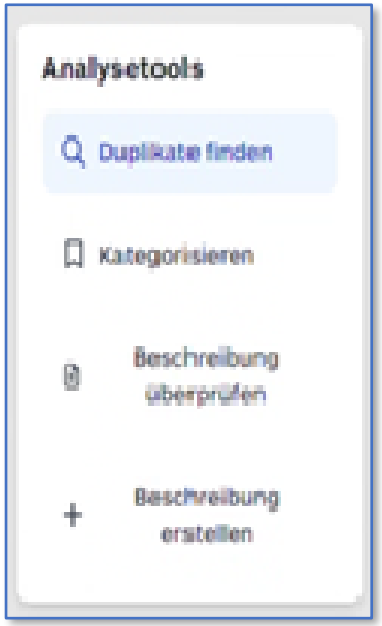
2

Details

Area	Feedback	Suggestions
Title	The title is informative and understandable.	No improvement suggestions necessary.
Description	The description is comprehensive and provides a good overview of the test case.	No improvement suggestions necessary.
Preconditions	Preconditions that are necessary for executing the test are missing.	Add specific preconditions, e.g., "The system must be initialized with the relevant test data."
Test Steps	The test step contains multiple actions and is not formulated clearly enough.	Split the test step into several steps, e.g.: 1. "Check the delivered fields and attributes in the results view." 2. "Ensure that the correct posting entries have been derived." 3. "Verify the posting direction and that the posting was made to the correct sub-ledger accounts." 4. "Ensure that the transaction amount has been converted correctly."
Requirements Coverage	The requirements from the linked user stories are partially covered, but specific details for full coverage are missing.	Add specific test steps that fully cover the requirements from the user stories.

AI Use Case 2 | AI accelerates Defect Management





DETECT THE DUPLICATE DEFECTS IN THE TOOL LIBRARY

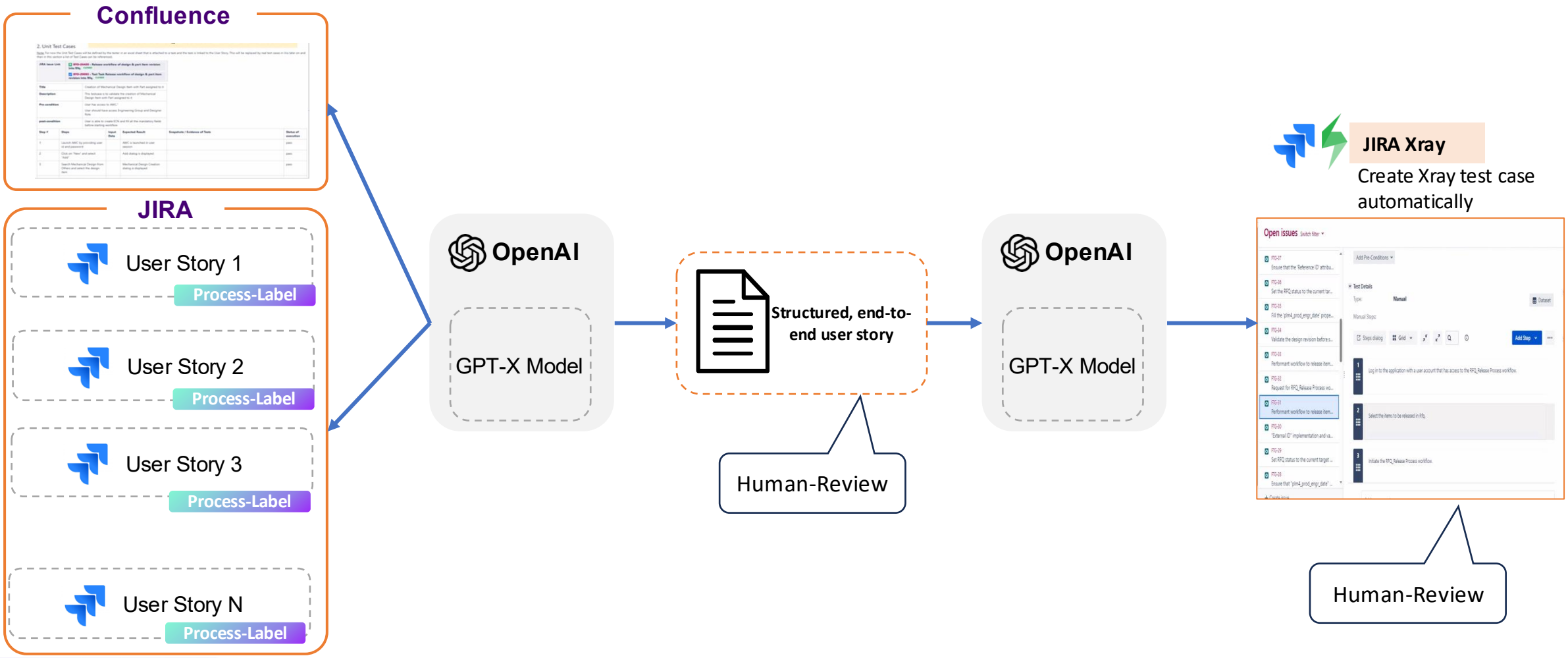
CATEGORIZATION OF DEFECTS (IS THIS A PERFORMANCE DEFECT, USABILITY DEFECT? ETC.)

DEFECT QUALITY CHECK: DESCRIPTION CORRECT?, ACTUAL TO EXPECTED BEHAVIOR INHERITED?, FUNCTIONAL/TECHNICAL IMPACT INCLUDED ETC.?

DEFECT DESCRIPTION CAN BE CREATED FOR A NEW DEFECT BASED ON TEST CASE AND EXECUTION ID

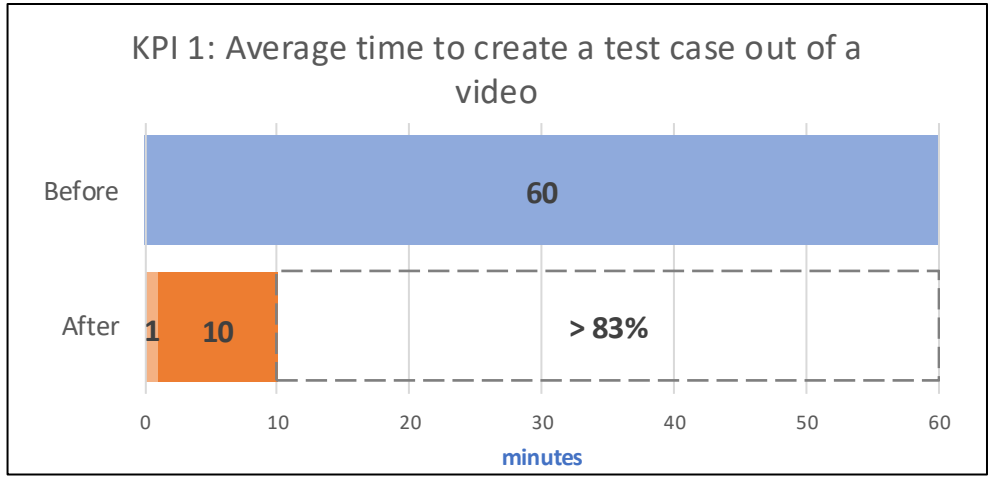
AI Use Case 3 – Architecture 2 | AI accelerates Test Case Creation

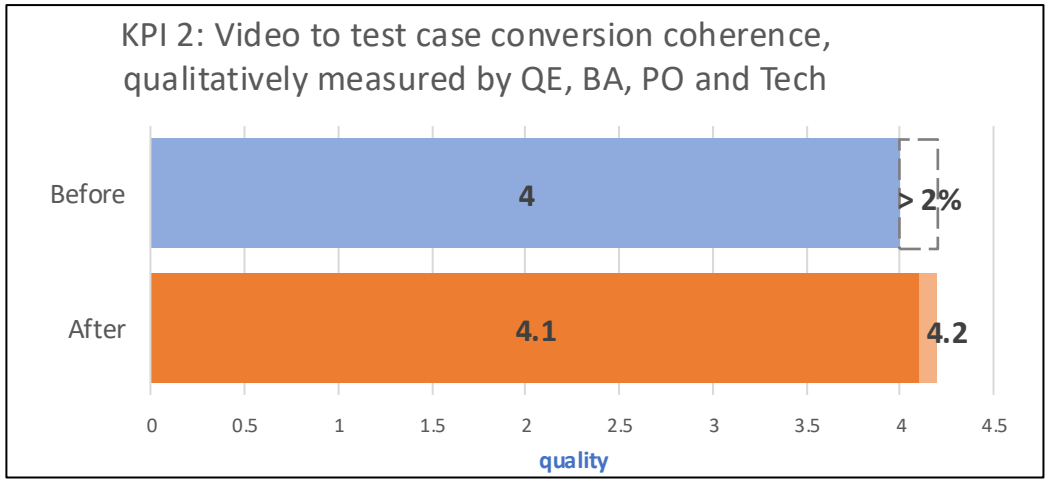
Automated generation of manual end-to-end (E2E) test cases using GenAI for a banking client's SAP S/4HANA application, leveraging inputs from Confluence and JIRA.



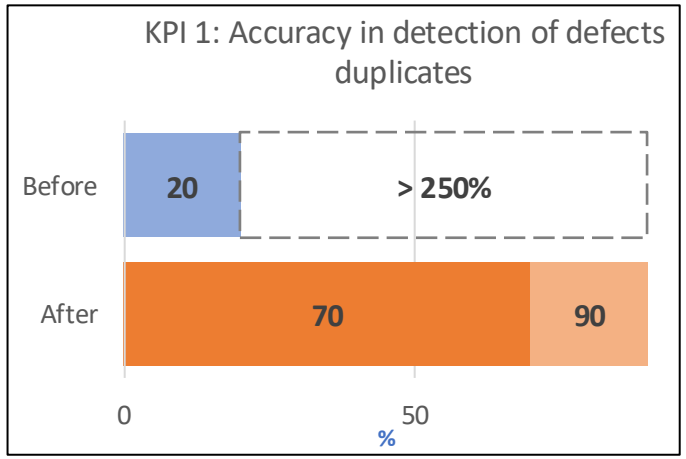
AI Use Cases | Real Project Measurements

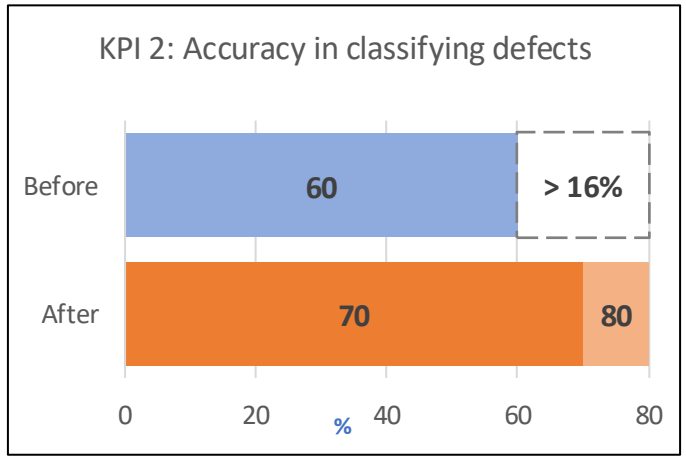
Video/Audio To Test Case Converter

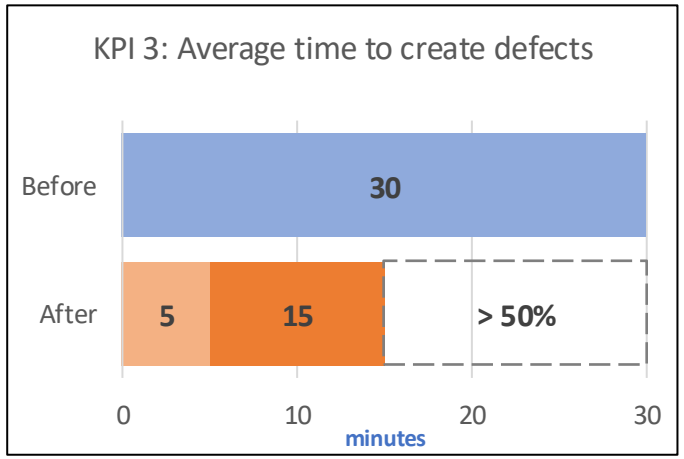




Defect Management Copilot





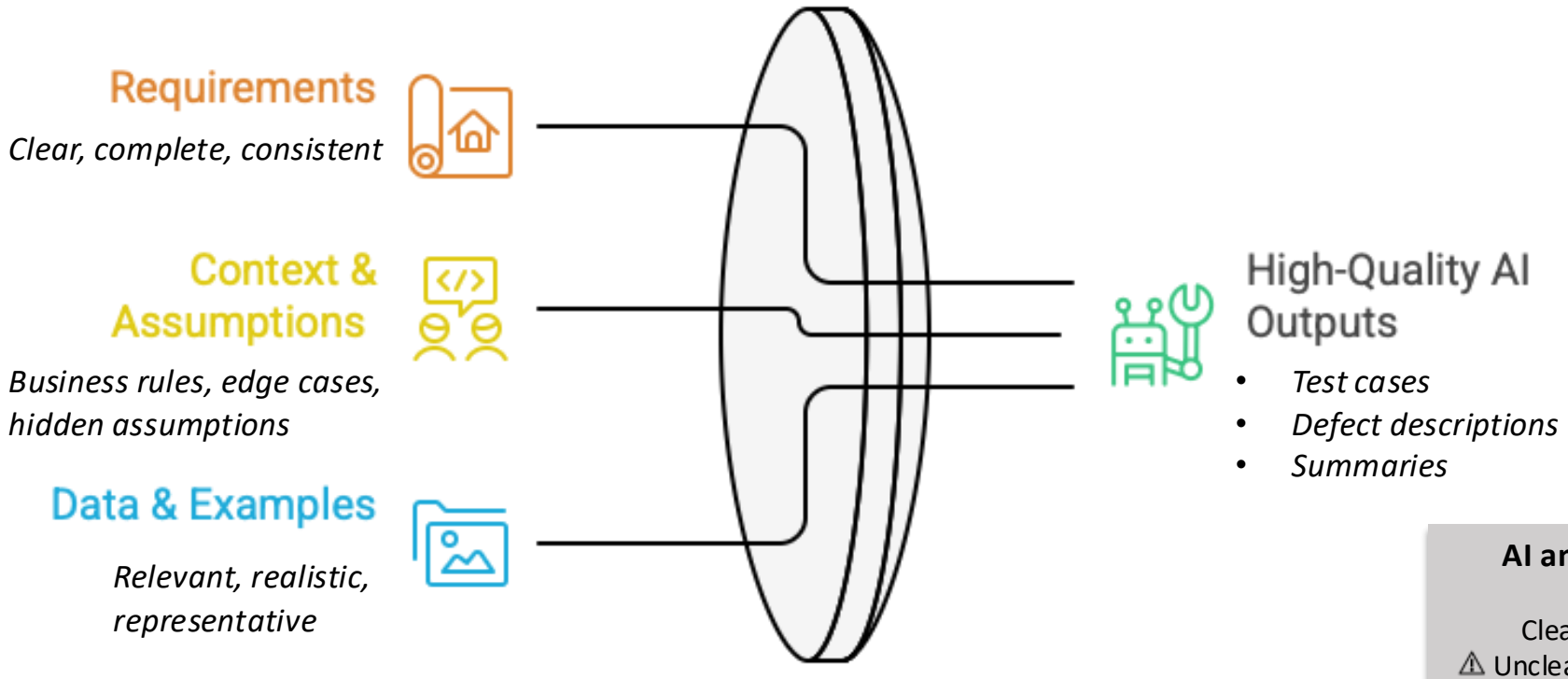


Improvement	Original	AI - Improvment	Potential

Where to stay cautious in AI usage | Meaning and Intent

AI doesn't understand intent, it assumes it!

What goes in shapes what comes out

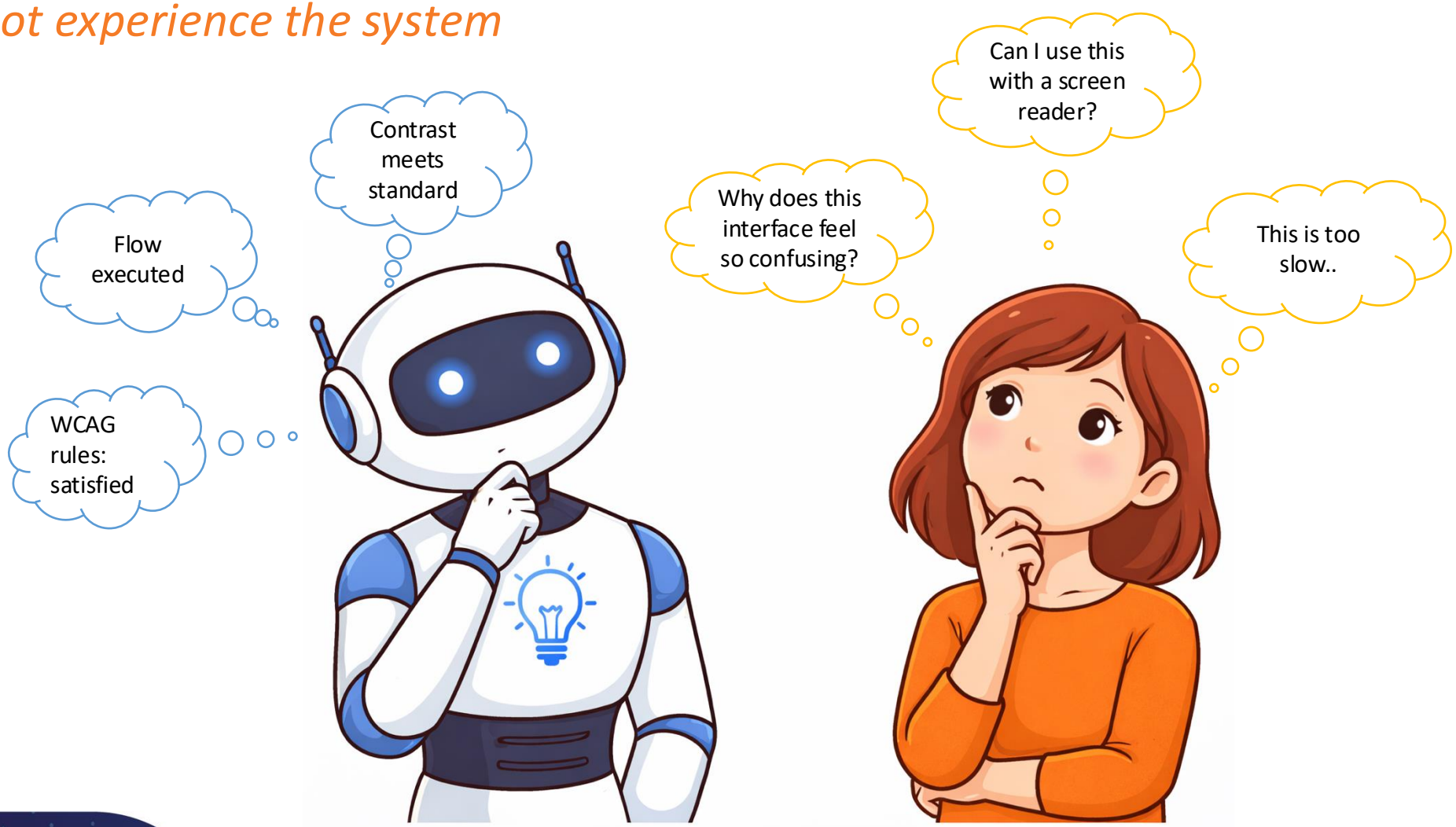


AI amplifies input quality!

Clear input → useful output
⚠ Unclear input → misleading output

Where to stay cautious in AI usage | Discovery & Experience

AI cannot experience the system



All checks passed ✅

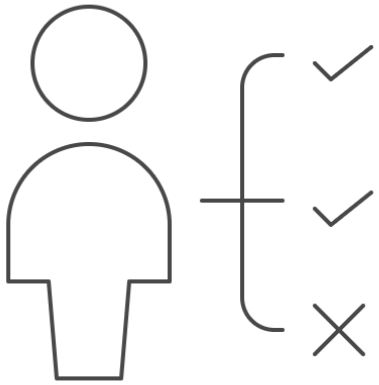
Still a poor experience 😟

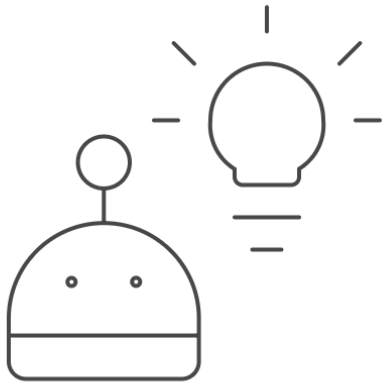
Where to stay cautious in AI usage | Risk Management & Decision Making

AI can process data, Humans are responsible for it!

Accountability cannot be automated

- Evaluate business impact
- Balance risk and value
- Decide Go / No-Go
- Take responsibility



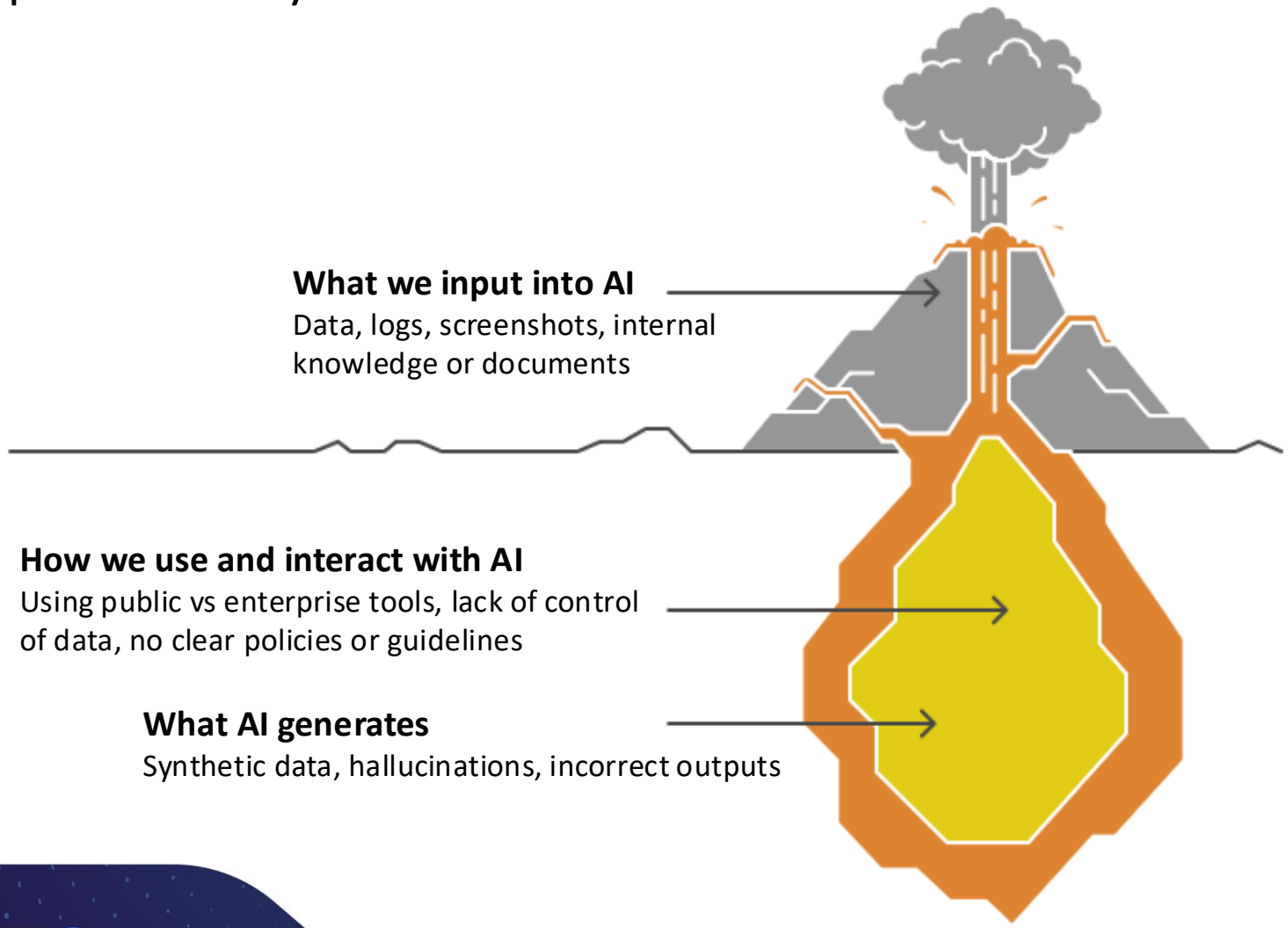


- Analyzes data & trends
- Highlights risks
- Suggests recommendations
- Provides insights

Humans 100% | 0% AI

Own decisions | **Support** decisions

Where to stay cautious in AI usage | Data Safety, Security & Responsibility



What we input into AI
Data, logs, screenshots, internal knowledge or documents

How we use and interact with AI
Using public vs enterprise tools, lack of control of data, no clear policies or guidelines

What AI generates
Synthetic data, hallucinations, incorrect outputs

⚠ Potential Risks

- Data leakage & exposure
- Compliance violations (e.g. GDPR)
- Loss of control over data
- Misleading or unsafe outputs -> False Confidence

Client Readiness in AI usage | How ready are the clients to use AI?

Skepticism

Limited trust in AI

- No policy or governance
- No tool access
- Low team awareness

Assisted use

AI as a supporting tool

- Draft policy & governance
- Limited tool access
- Basic security review

Guided collaboration

Active human AI interaction

- Approved policy & oversight
- Broad tool access
- Compliance in progress

Confident use

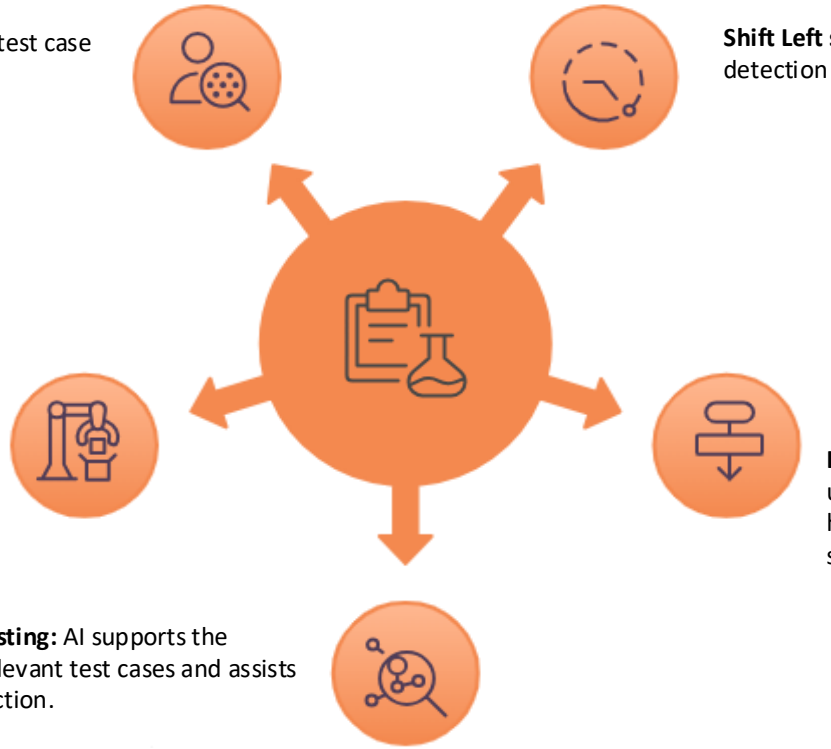
AI integrated into workflows

- Full governance & compliance
- Secure managed access
- Team AI-confident

Relevance of Traditional Testing Methods

Manual Testing: AI assists in test case creation and assessment.

Shift Left strategy: AI enables early defect detection and cost reduction..



Automated Testing: AI supports the rapid and high-quality creation of automated test cases.

Hybrid Agile Methodology: AI analyzes user stories and requirements to generate high-quality test cases and accelerate sprint preparation

Regression Testing: AI supports the selection of relevant test cases and assists in defect detection.

„In the future, incorporating AI techniques in different software testing activities will make it easier to perform testing activities.“ *

„AI extends and optimizes traditional automation but still relies on human supervision.“ *²

„AI's Impact on Software Development: AI has significantly influenced traditional software development practices by automating code generation, testing, debugging, and deployment processes.*³

#ISTC2026

* Source: Islam et al., AI in Software Testing (2023)

*² Source: Garousi et al., AI-powered software testing tools (2024)

*³ Source: Maduapati, AI's Impact on Software Development (2024)

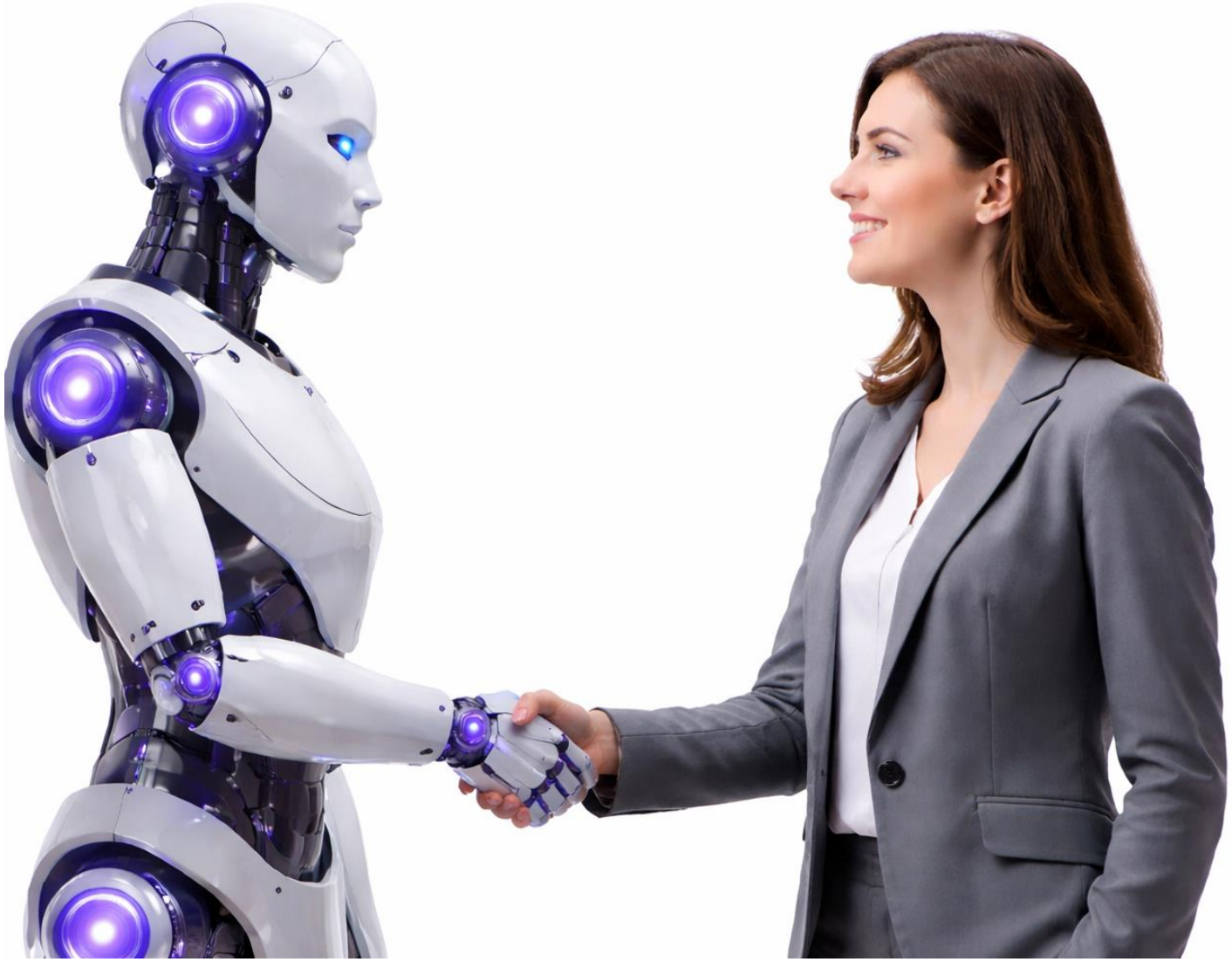
Conclusion | “Let AI Be Your Partner, Not Your Master”

Key Takeaways

1. AI accelerates testing activities and complements existing testing methodologies, enabling quality engineers to work more efficiently

2. Responsibility remains with humans: AI requires human supervision, as outputs may contain inaccuracies or failures

3. Final validation and decision-making remain human-led



ISTANBUL
SOFTWARE
TESTING
CONFERENCE

Thank you!

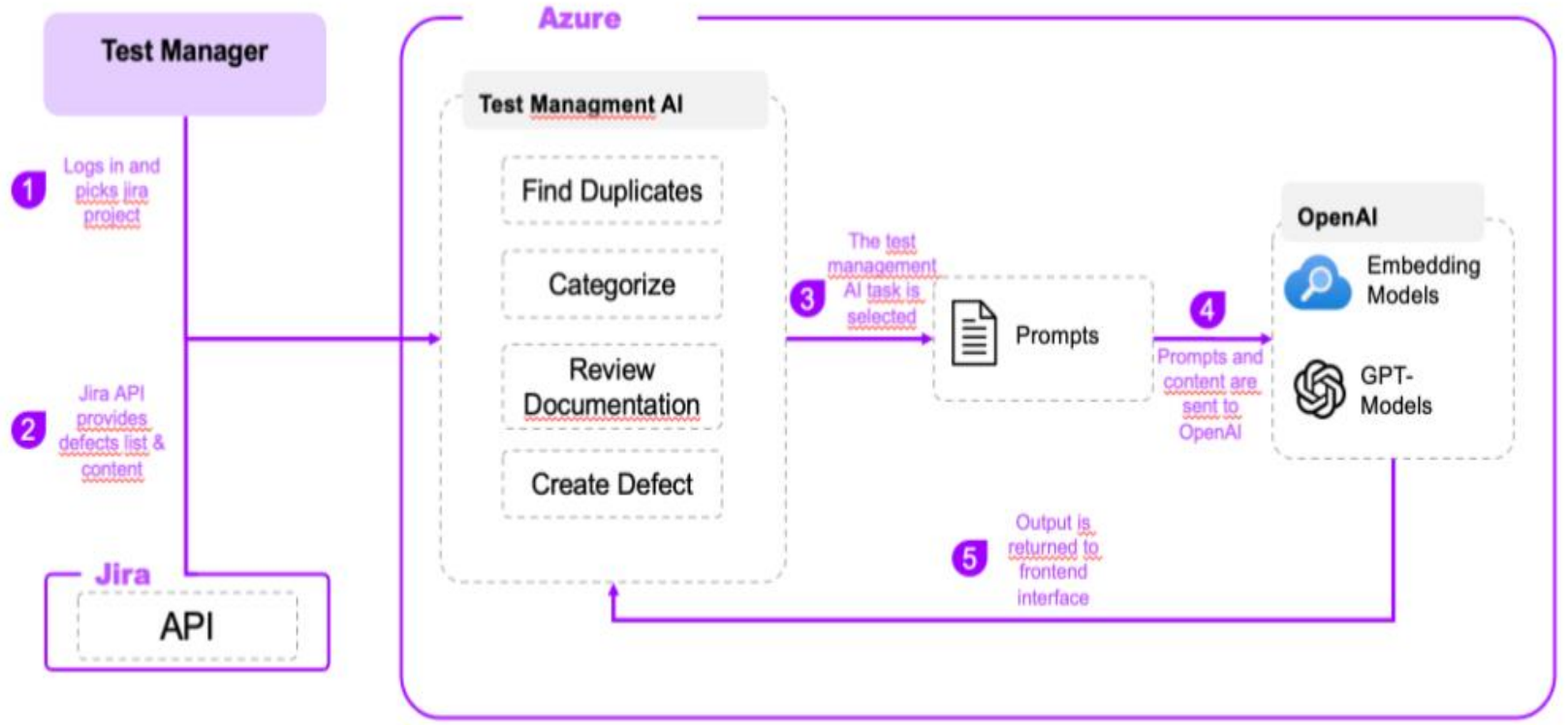
Elanur Coskun

Anastasia Simou

#ISTC2026

AI Use Case 2 | AI accelerates Defect Management

How does defect management copilot work?



1. The test manager logs in into the application and selects the Jira project to handle.
2. The Application retrieves the list of defects from Jira, together with further information and shows it in the UI.
3. The test manager selects the list of defects to include and the task to accomplish.
4. The defects together with fine tuned prompts are sent to the LLM.
5. The LLM provide the response, and the output is parsed and shown in the UI.