DeepQuali: Initial results of a study on the use of large language models for assessing the quality of user stories.

Dr. Adam Trendowicz
Dep. Data Science (DS)
Fraunhofer IESE

Workshop on Applications of Artificial Intelligence for Software Engineering (AI4SE)
49th German Conference on Artificial Intelligence, **KI2026**
Bremen, 12. August 2026

Focus of DeepQuali

DeepQuali focuses on the validation of software requirements (ISO/IEC 15288:2023)

Elicit

- Characterize context and identify stakeholders, in particular intended users of the component
- Understand stakeholders and discover their needs
- Define high-level requirements: functional and non-functional (quality)

Analyze

- Derive more detailed requirements
- Create multiple views of the requirements, e.g., prototypes, graphical analysis modes, and tests
- Prioritize defined and identify missing requirements
- Evaluate technical feasibility, risk, and failure modes
- Provide feedback and refine requirements

Specify

- Document requirements in a form that facilitates communication among project stakeholders

Validate

- Ensure that requirements are correct and consistent (verification: *Are the requirements defined right?*)
- Ensure that requirements reflect actual customer needs of relevant stakeholders (validation: *Are the right requirements defined?*)

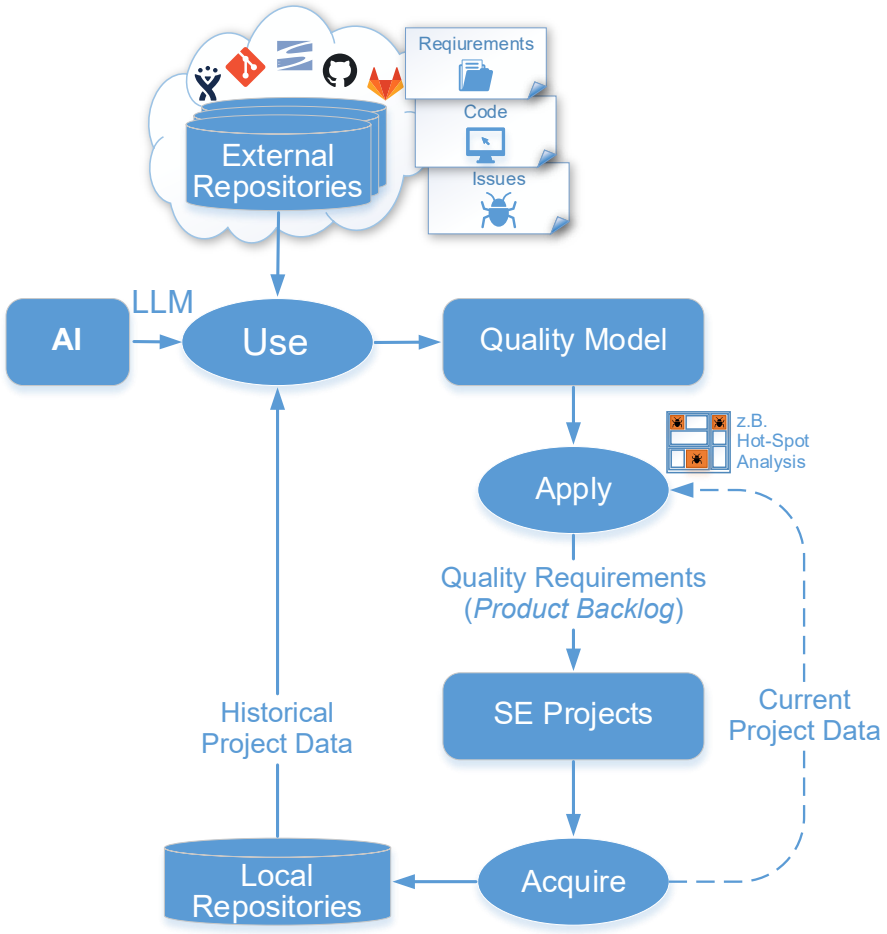
Requirements Validation Tasks Supported by AI

AI can be used for several specific evaluation tasks, which can be organized into an evaluation pipeline

Purpose/Task	AI Usage	Input	Output
Assess quality of requirements	Rate the quality of requirements	• Quality model, e.g., criteria, rules • Context, e.g., domain, infrastructure	Quality assessments, e.g., scores
Detect requirements quality deficits	Describe quality deficits of requirements	• Quality model, e.g., criteria, rules • Context, e.g., domain, infrastructure	Specific quality deficits (problems, issues)
Classify quality deficits, e.g., criticality	Group requirements deficits	• Quality deficits • Classification criteria	Classification of quality deficits
Derive problem resolutions	Generate potential resolutions of given quality deficits	• Quality assessments / deficits • Classification of quality deficits • Solution criteria	• Resolution recommendations: How requirements can be improved • Improved requirements: Examples & alternatives
Explain (reason) the results	Explain & justify specific outcomes of the quality evaluation, e.g., ratings, deficits, classification, or resolutions	Outputs of the above tasks: • Quality assessments • Quality deficits • Classification of quality deficits • Problem resolutions	Reasoning underlying a specific output

DeepQuali Projekt

AI-based assessment and improvement of software quality



Context

- Joint research & innovation project with two SMEs, funded by German Ministry of Research, Technology and Space.

Objective

- Development-time assessment and improvement of software quality, in the context of agile software development, based on the direct analysis of unstructured software artifacts, using AI technologies.

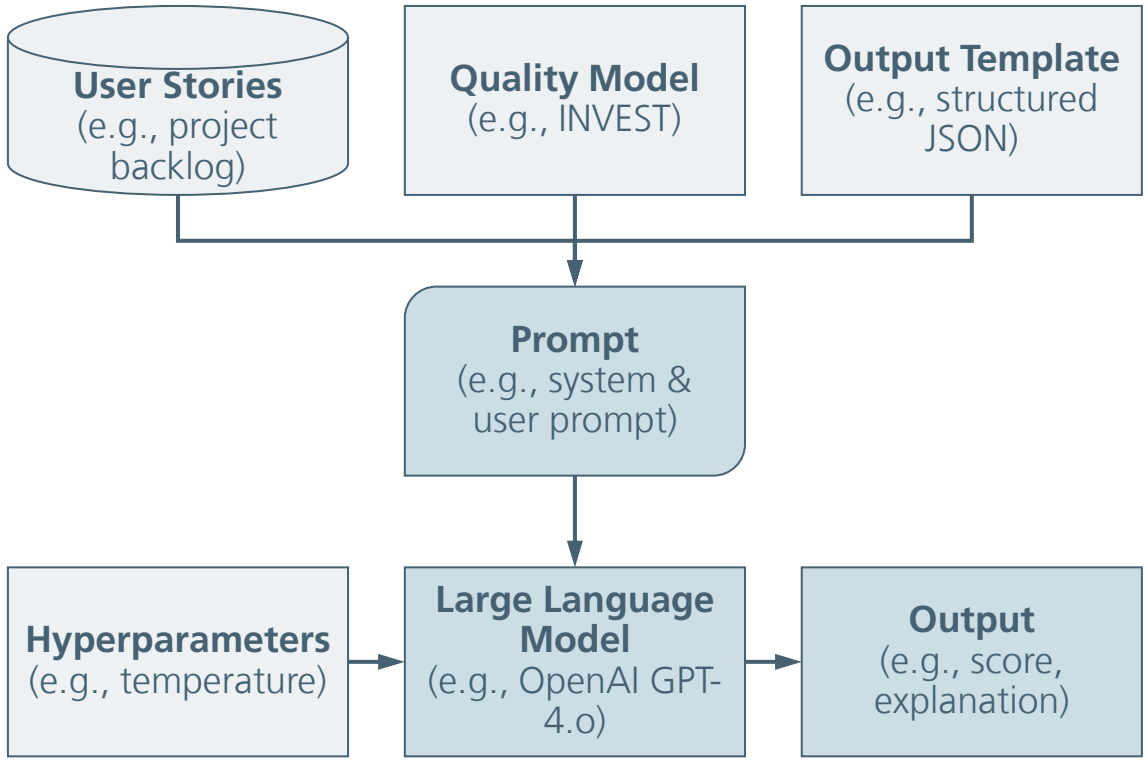
Components

- Requirements: Quality assessment of requirements specifications, specifically user stories.
- Source code: Quality assessment of software architecture and code, based on the analysis of source code.

Web: https://deepquali.de/

DeepQuali Approach

Using LLM upon custom-defined decision criteria to assess the quality of user stories



Context

- Small and medium enterprises
- Agile software development

Need

- Effective and efficient control of user stories' quality

Objectives

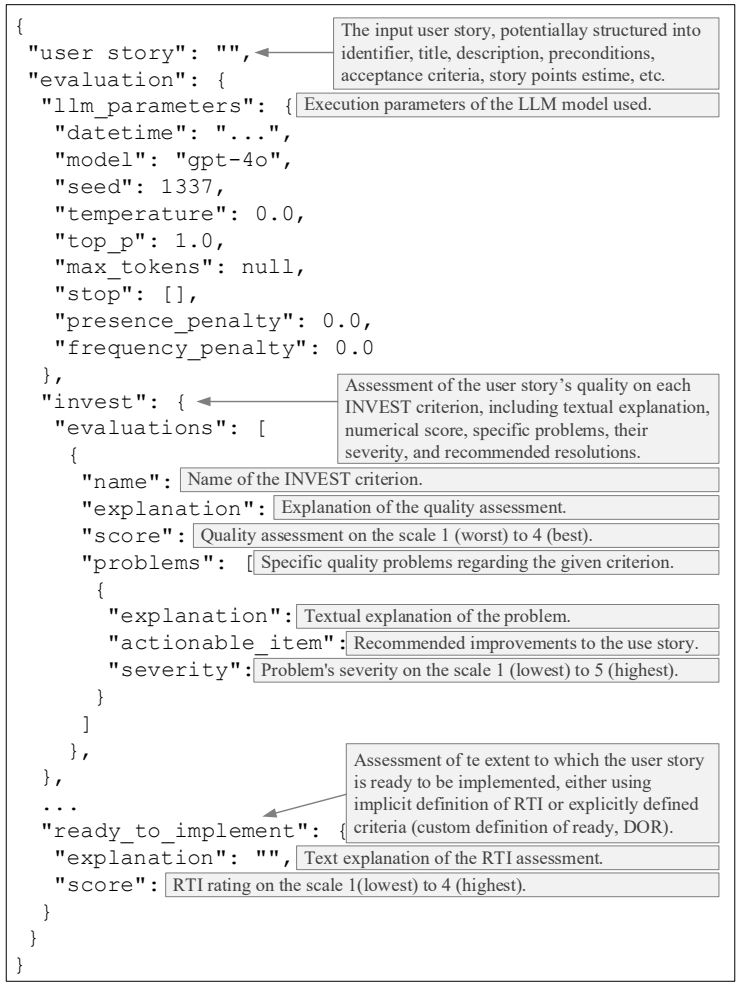
- Assess quality, identify deficits, propose improvements, explain reasoning
- Assure comprehensibility

Solution

- LLM: OpenAI GPT4.o
- Structured inputs & outputs (JSON)

DeepQuali Inputs and Outputs

Forcing structured form of inputs and output improves explainability of the approach



Inputs

- Parameters of the LLM model
- User story: anonymized, in original structure
- Assessment criteria: INVEST, DOR (Definition Of Ready)

Outputs

For each criterion:

- Score: Numeric quality rating of fulfilling given quality criterion
- Explanation: Explanation and justification of the numeric quality score
- Problems: description of specific quality issues regarding the criterion
- Severity: Numeric rating of problems' severity
- Actionable item: Concrete improvements recommended to solve problems

Based on all criteria:

- Ready to implement: readiness of a given user story to be released for implementation
  - Score: Numeric rating the readiness to be implemented
  - Explanation: Explanation and justification of the numeric quality score

Prompts Used in DeepQuali

Decomposing prompt into generic and specific part improves portability of the approach

System Prompt

System prompt represents the generic part of the prompt describing the context and the task

- **Role**: You are an experienced software engineer with deep knowledge in agile practices and user story evaluation.
- **Task**: You evaluate user stories on whether they are ready to be implemented.
- **Outcome**: You respond in a given JSON schema.

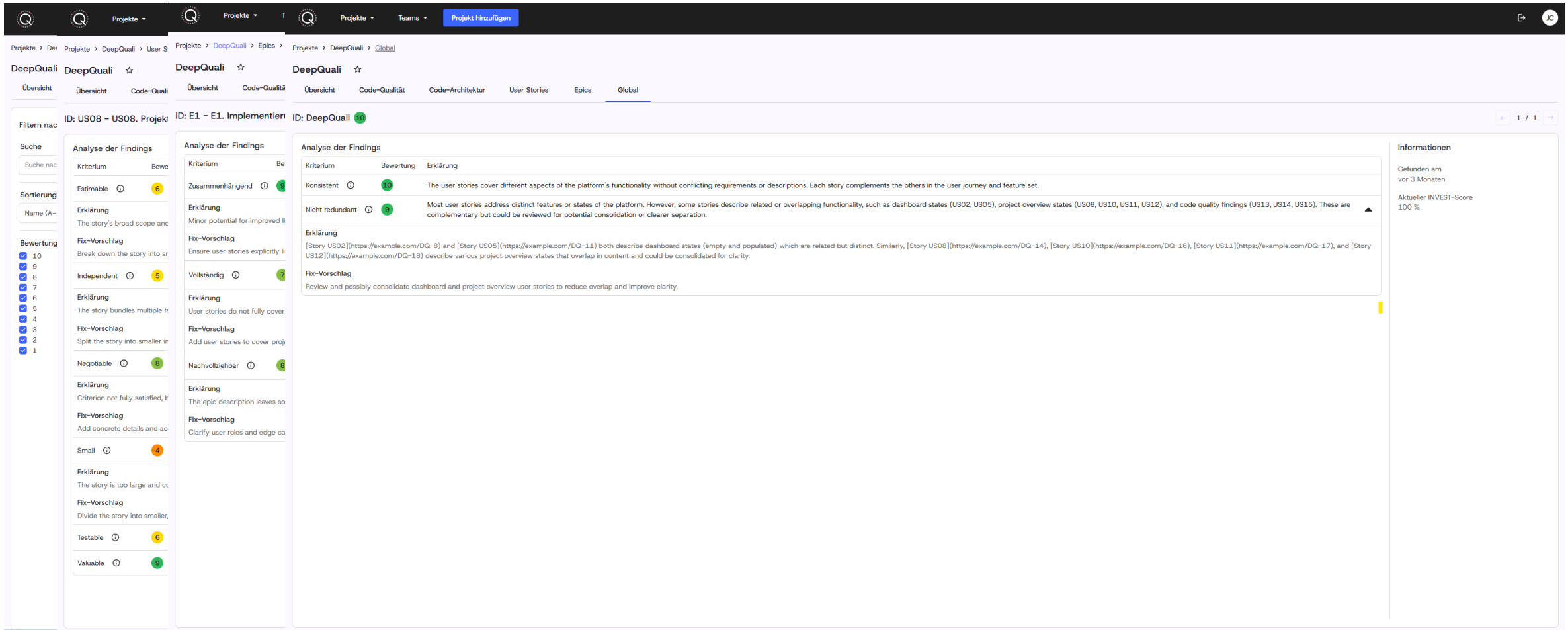
User Prompt

Specific part of the prompt explaining details of the task and its expected outcomes

- I will provide you with a user story below.
- Your task is to carefully read through the user story and evaluate it using the INVEST criteria.
- If an INVEST criterium is completely fulfilled and there is nothing to complain about, please don't hesitate to give the score 10.
- If an INVEST criterium is not completely fulfilled, please provide a brief explanation of what is missing and what can be done to improve it.
- Here is the user story in JSON: *{user_story_json}*

DeepQuali Implementation

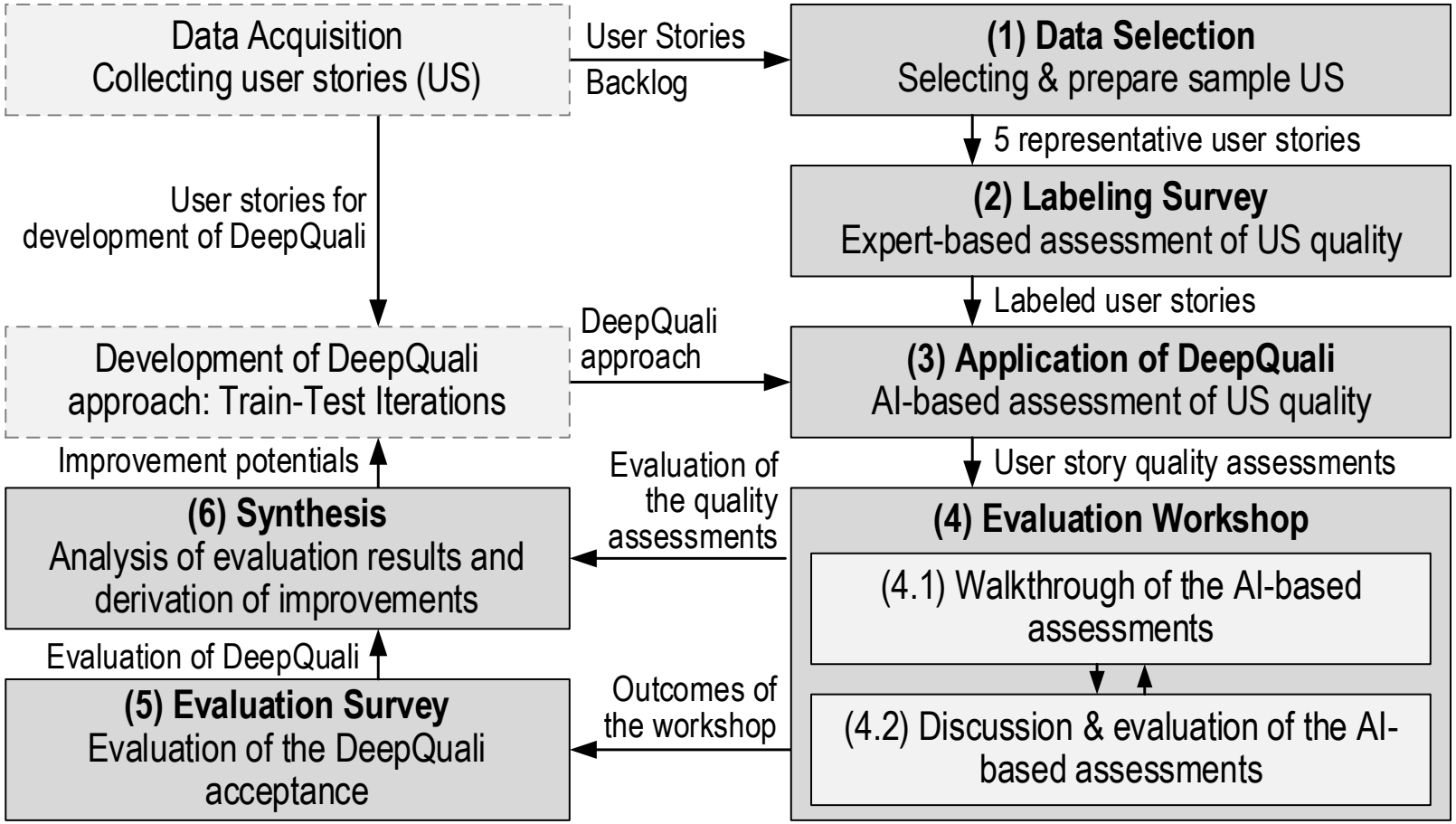
Used in the study command-line prototype was later implemented as GUI-Application



Fraunhofer IESE

DeepQuali Evaluation Procedure

Evaluation is based on the ground truth provided by subject matter experts



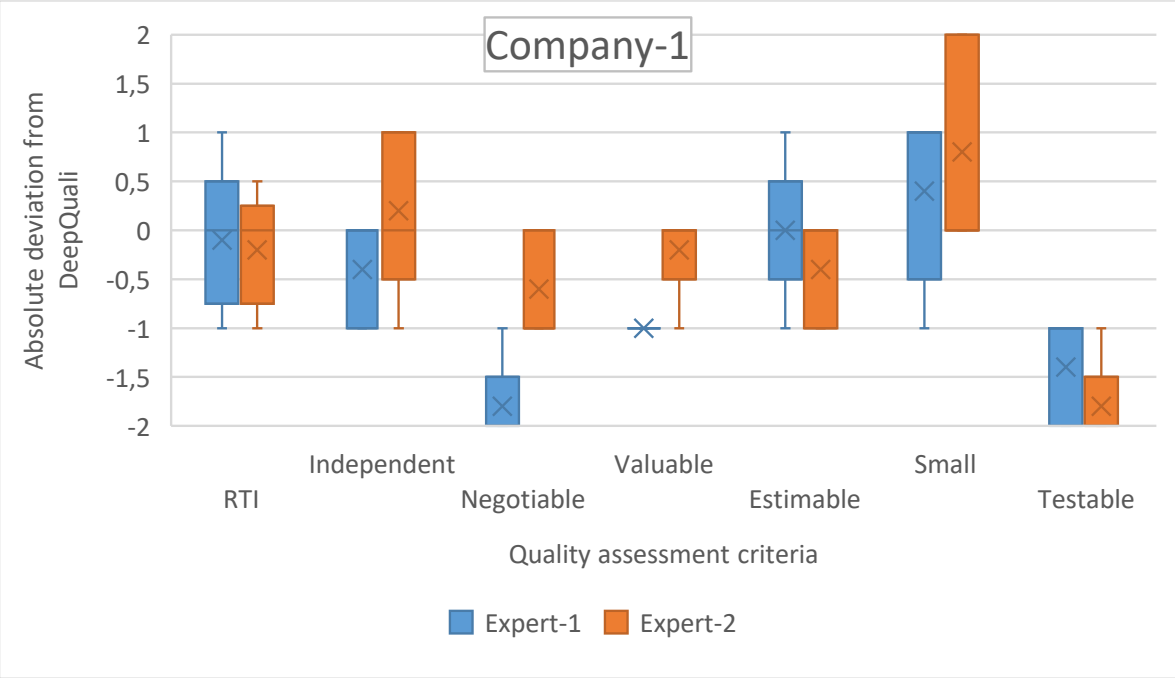
Evaluation context

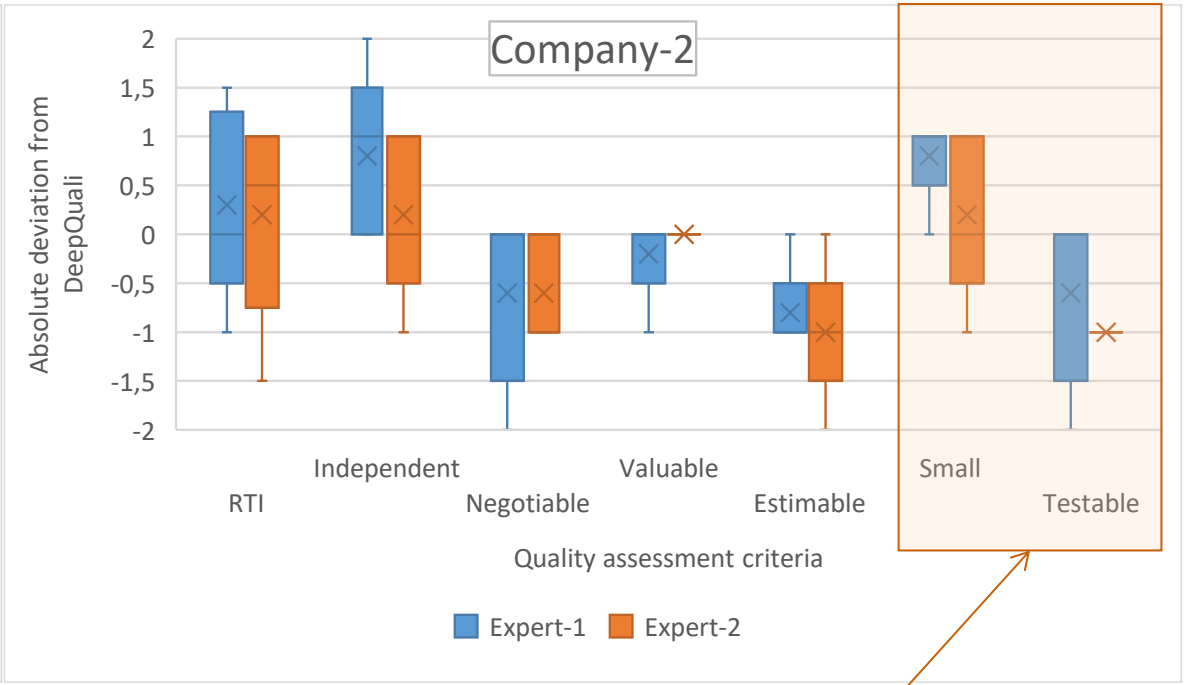
- Two small-size enterprises (SMEs)
- Agile software development
- Four human experts

Fraunhofer IESE

DeepQuali Evaluation Results: Consistency of the quality assessments

Experts are more critical on the *Testable* and less critical on the *Small* criteria





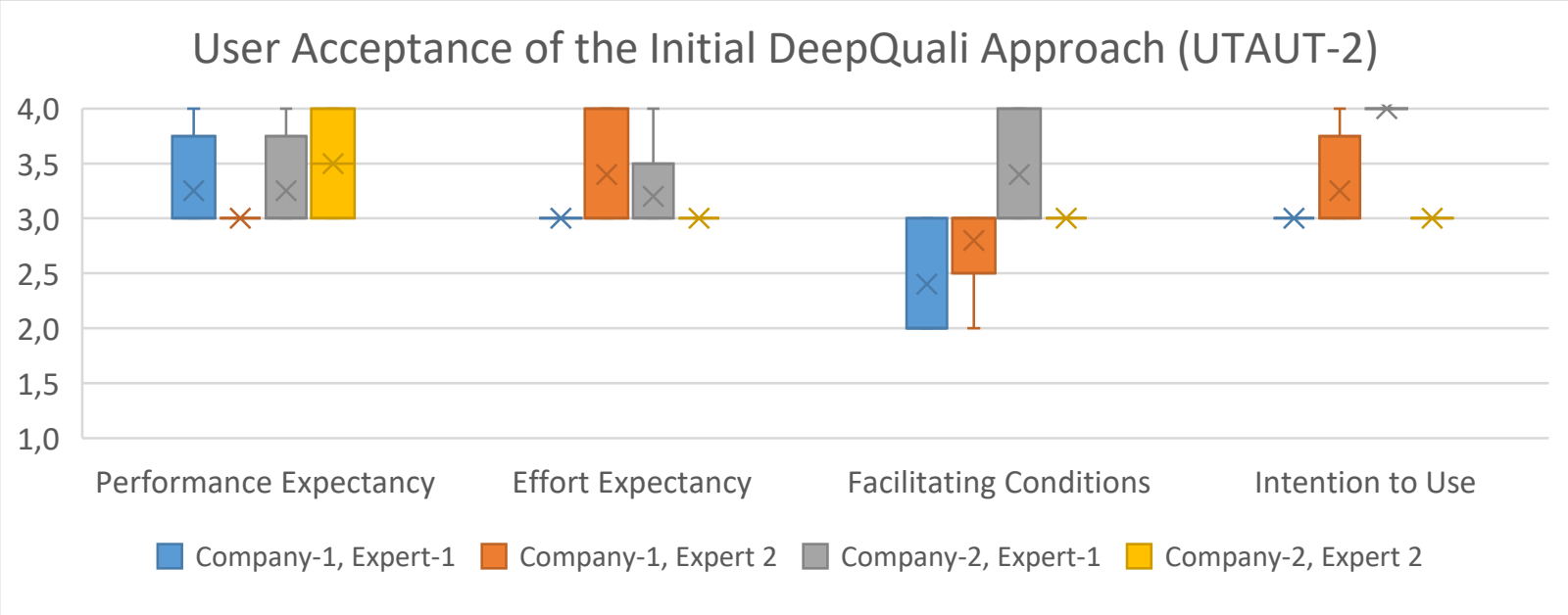
General observations

- Expert E1 wit higher experience was principally more critical than AI
- Experts consistently more critical than AI on the *Testable* criterion
- Experts consistently less critical than AI on the *Small* criterion

Explicit acceptance criteria in a user story improved its testability but at the same time increased its size.

DeepQuali Evaluation Results: Users' Acceptance

Experts appreciate the tool's practical usefulness but doubt its integrability into their work environment



Survey based on UTAUT

- **Performance expectancy**: The degree to which a person believes that using the technology will help improve their job or task performance.
- **Effort expectancy**: The perceived ease of using the technology.
- **Facilitating conditions**: The extent to which a person believes that organizational and technical resources exist to support technology use.
- **Intention to use**: The person's intention or willingness to use the technology.

UTAUT: Unified Theory of Acceptance and Use of Technology

DeepQuali Evaluation Results: Users' Feedback

Organization-specific standards may differ from 'average' baseline of an LLM

- High agreement between experts and LLM on INVEST criteria.
- Highest deviation in assessments on NET: *Negotiable*, *Estimable*, and *Testable* criteria.
  - Experts are more critical in their assessments than LLM.
  - More experienced experts are more critical than those with less experience.
- Experts rated consistently higher than LLM only on *Small*.
  - Potential reasons: The user stories represented a typical ("standard") size, the experts were familiar with. LLM expected smaller user stories.
- Human experts found the specific deficits identified by the LLM, along with additional explanations, very helpful for understanding the quality of user stories.

Lessons Learned

Proper framing, rather than brute force, improves the quality of LLM outcomes

- Provide LLM with comprehensive context information
  - Development and operation context of the software system, e.g., application domain and platform
  - Internal quality standards (e.g., regarding the *Size*), which may deviate from the common ones

- Provide relevant information to LLM as both general specification and specific examples.
  - Example: Specification of expected structure and size of user stories as well as typical examples of small-, medium- and large-size user stories.

- Force LLM to explain its outcomes and its reasoning
  - (!) It improves the assessment outcomes (quality rating closer to experts' assessments)
  - It improves understanding and trustworthiness of the outcomes

Thank you!

Questions, Remarks?

Contact

Dr. Adam Trendowicz
Fraunhofer IESE
Fraunhofer-Platz 1
67663 Kaiserslautern

DeepQuali Project Partner

Fraunhofer IESE

QA|WARE SOFTWARE ENGINEERING

LET'S DEV

Parts of this work were funded by the Federal Ministry of Research, Technology and Space (BMFTR) under grant no. 01IS23016D.