34th Annual INCOSE international symposium

hybrid event

Dublin, Ireland
July 2 - 6, 2024

Impact Analysis of using Natural Language Processing and Large Language Model on Automated Correction of Systems Engineering Requirements

Arthur Oliveira
Feature Systems Engineer
aoliv355@ford.com

Automotive Feature Systems

Actions

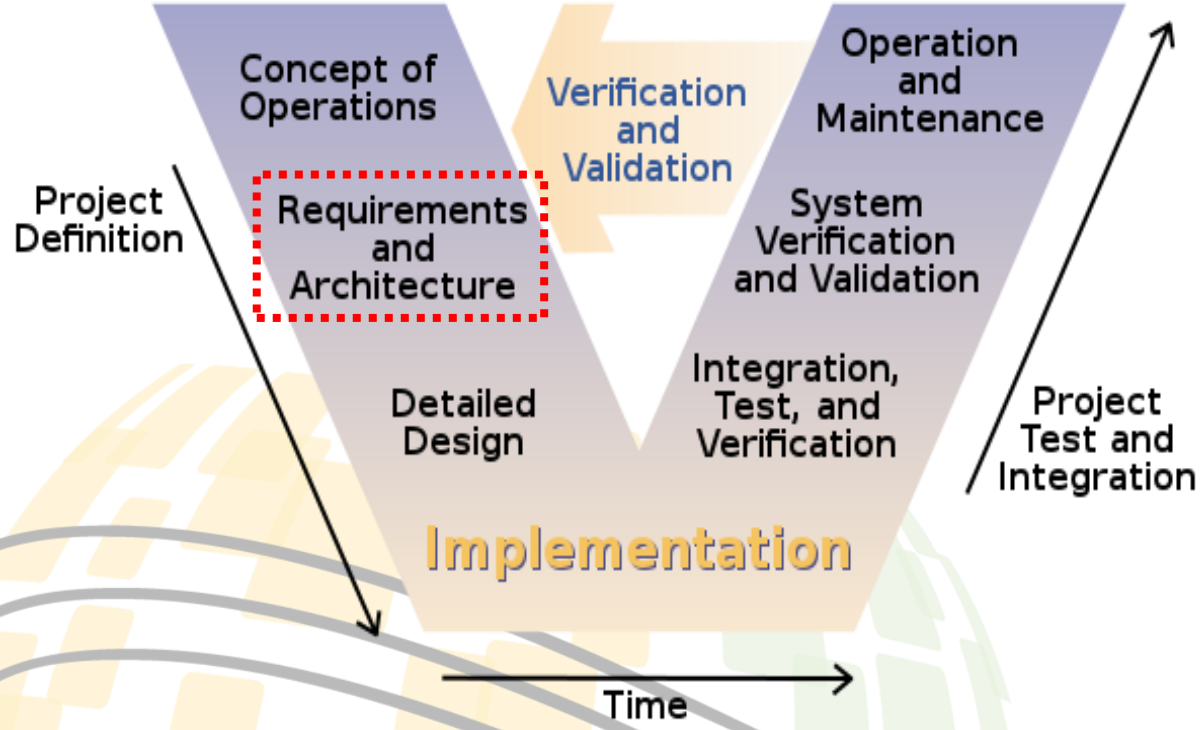
Feature

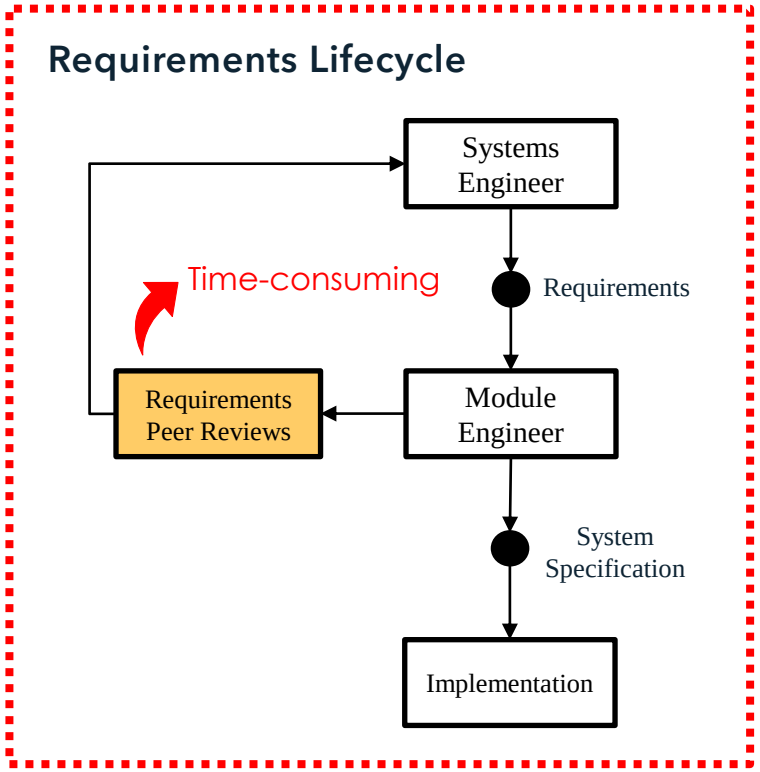
Electronic Systems

Functions

(functionalities)

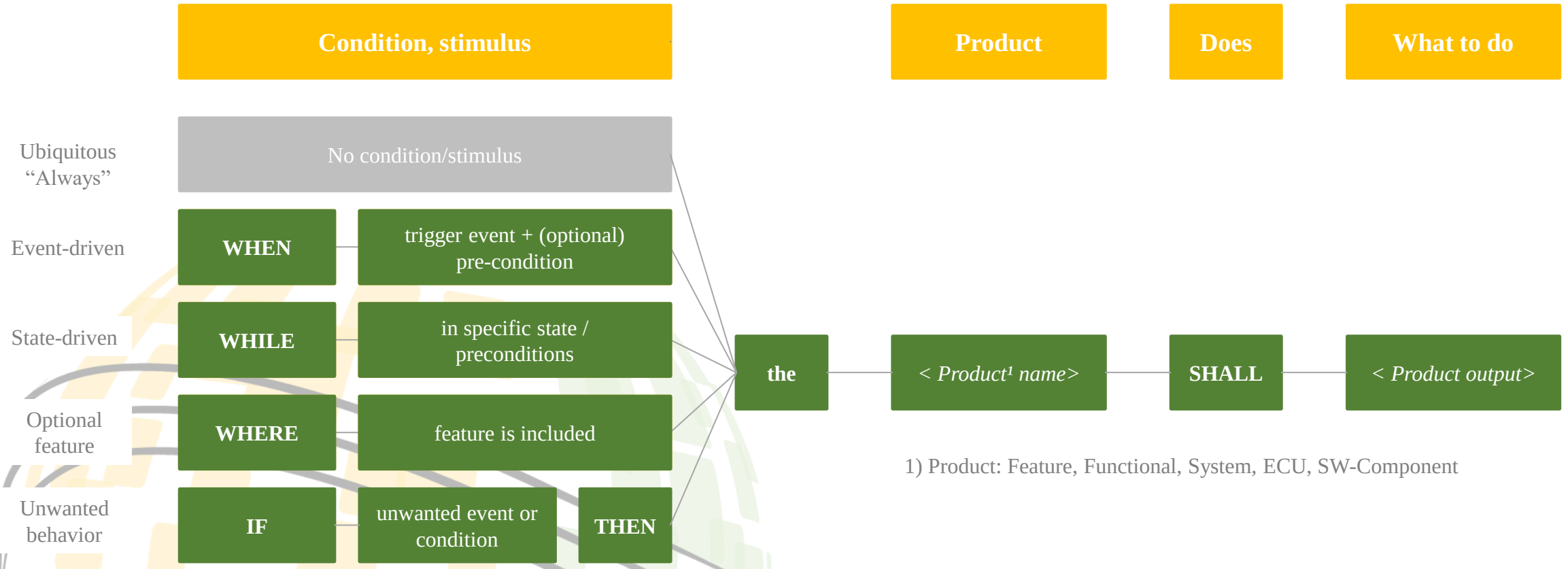
Problem Statement





A significant amount of software issues originates from deficient and ambiguous requirements analysis. These issues become more challenging to solve in the advanced phases of product development.

Easy Approach to Requirements Syntax (EARS)



Example (Event-driven): **When** the call button is pressed, **the** *feature* "*ABC*" **shall** illuminate the direction light.

Requirements Feedback through EARS rules

Example

"*When the button is pressed, the feature **SHALL**.*"

Unclear ✗
Requirement

+ EARS: After the term "shall" the requirement shall state an action.

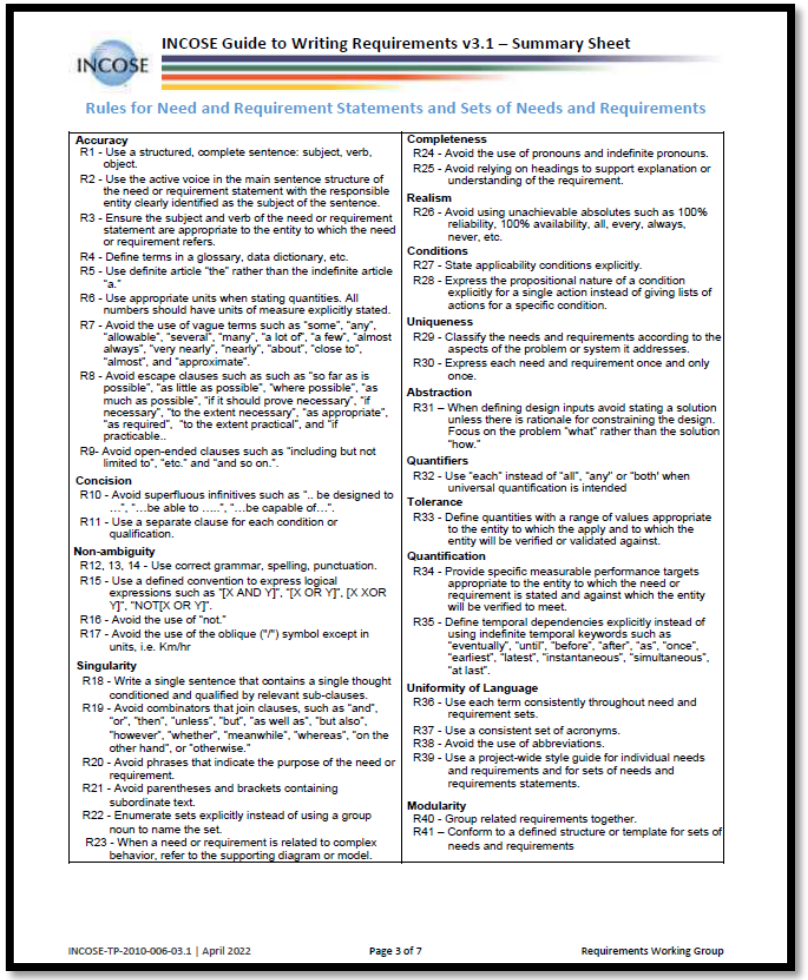
Clear Requirement ✓

"*When the button is pressed, the feature **SHALL** illuminate the way.*"

INCOSE Guide to Writing Requirements

- ✓ The Guide to Writing Requirements is an INCOSE Technical Product prepared by INCOSE.
  - 15 Rule Whitin 41 Rules of Requirement Writing Best Practices in this book, this project is covering 15 rules of them.

INCOSE-TP-2010-006-03.1 | April 2022 Page 3 of 7 Requirements Working Group



INCOSE Guide to Writing Requirements v3.1 – Summary Sheet

Rules for Need and Requirement Statements and Sets of Needs and Requirements

Accuracy
R1 - Use a structured, complete sentence: subject, verb, object.
R2 - Use the active voice in the main sentence structure of the need or requirement statement with the responsible entity clearly identified as the subject of the sentence.
R3 - Ensure the subject and verb of the need or requirement statement are appropriate to the entity to which the need or requirement refers.
R4 - Define terms in a glossary, data dictionary, etc.
R5 - Use definite article "the" rather than the indefinite article "a."
R6 - Use appropriate units when stating quantities. All numbers should have units of measure explicitly stated.
R7 - Avoid the use of vague terms such as "some", "any", "allowable", "several", "many", "a lot of", "a few", "almost always", "very nearly", "nearly", "about", "close to", "almost", and "approximate".
R8 - Avoid escape clauses such as such as "so far as is possible", "as little as possible", "where possible", "as much as possible", "if it should prove necessary", "if necessary", "to the extent necessary", "as appropriate", "as required", "to the extent practical", and "if practicable..
R9- Avoid open-ended clauses such as "including but not limited to", "etc." and "and so on.".

Concision
R10 - Avoid superfluous infinitives such as ".. be designed to ….", "…be able to ……", "….be capable of….".
R11 - Use a separate clause for each condition or qualification.

Non-ambiguity
R12, 13, 14 - Use correct grammar, spelling, punctuation.
R15 - Use a defined convention to express logical expressions such as "[X AND Y]", "[X OR Y]", [X XOR Y]", "NOT[X OR Y]".
R16 - Avoid the use of "not."
R17 - Avoid the use of the oblique ("/") symbol except in units, i.e. Km/hr

Singularity
R18 - Write a single sentence that contains a single thought conditioned and qualified by relevant sub-clauses.
R19 - Avoid combinators that join clauses, such as "and", "or", "then", "unless", "but", "as well as", "but also", "however", "whether", "meanwhile", "whereas", "on the other hand", or "otherwise."
R20 - Avoid phrases that indicate the purpose of the need or requirement.
R21 - Avoid parentheses and brackets containing subordinate text.
R22 - Enumerate sets explicitly instead of using a group noun to name the set.
R23 - When a need or requirement is related to complex behavior, refer to the supporting diagram or model.

Completeness
R24 - Avoid the use of pronouns and indefinite pronouns.
R25 - Avoid relying on headings to support explanation or understanding of the requirement.

Realism
R26 - Avoid using unachievable absolutes such as 100% reliability, 100% availability, all, every, always, never, etc.

Conditions
R27 - State applicability conditions explicitly.
R28 - Express the propositional nature of a condition explicitly for a single action instead of giving lists of actions for a specific condition.

Uniqueness
R29 - Classify the needs and requirements according to the aspects of the problem or system it addresses.
R30 - Express each need and requirement once and only once.

Abstraction
R31 – When defining design inputs avoid stating a solution unless there is rationale for constraining the design. Focus on the problem "what" rather than the solution "how."

Quantifiers
R32 - Use "each" instead of "all", "any" or "both' when universal quantification is intended

Tolerance
R33 - Define quantities with a range of values appropriate to the entity to which the apply and to which the entity will be verified or validated against.

Quantification
R34 - Provide specific measurable performance targets appropriate to the entity to which the need or requirement is stated and against which the entity will be verified to meet.
R35 - Define temporal dependencies explicitly instead of using indefinite temporal keywords such as "eventually", "until", "before", "after", "as", "once", "earliest", "latest", "instantaneous", "simultaneous", "at last".

Uniformity of Language
R36 - Use each term consistently throughout need and requirement sets.
R37 - Use a consistent set of acronyms.
R38 - Avoid the use of abbreviations.
R39 - Use a project-wide style guide for individual needs and requirements and for sets of needs and requirements statements.

Modularity
R40 - Group related requirements together.
R41 – Conform to a defined structure or template for sets of needs and requirements

INCOSE-TP-2010-006-03.1 | April 2022 Page 3 of 7 Requirements Working Group

Requirements Feedback through INCOSE rules

Example

👎 *"When **any** button is pressed , the feature shall illuminate the way."*

+ INCOSE R7: Avoid the use of vague terms such as "any".

Clear Requirement ✓

👍 *"When the **light button** is pressed, the feature shall illuminate the way."*

Natural Language Processing (NLP)

The **spaCy** library in Python was used with a pre-trained NLP model to load tokenizer and tagger in English.

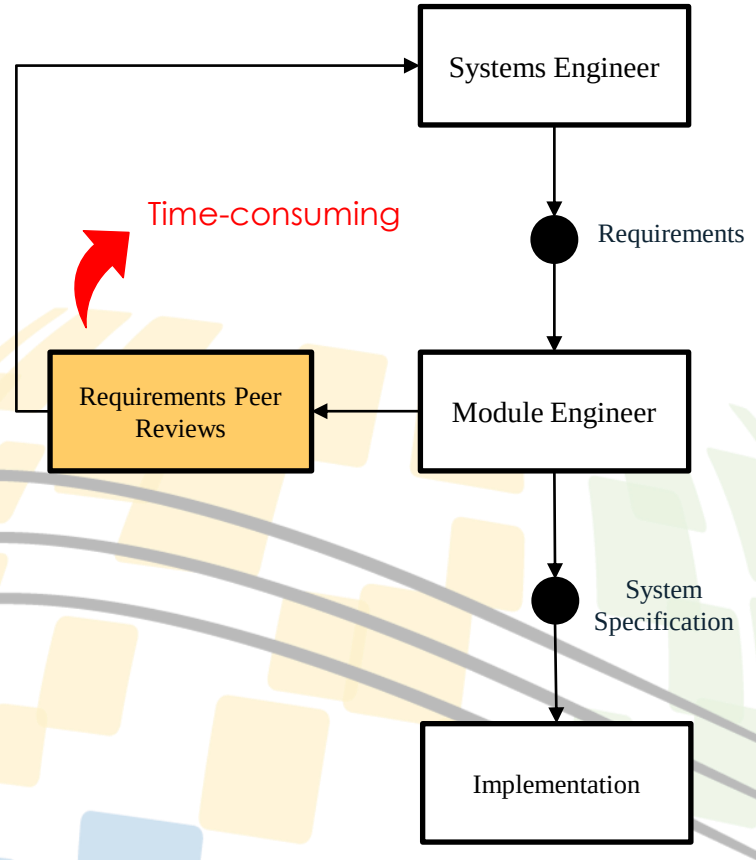
"When the button is pressed, the feature shall illuminate the way."

0	1	2	3	4	5	6	7	8	9	10	11	12
When	the	button	is	pressed	,	the	feature	shall	illuminate	the	way	.
SCONJ	DET	NOUN	AUX	VERB		DET	NOUN	AUX	VERB	DET	NOUN	

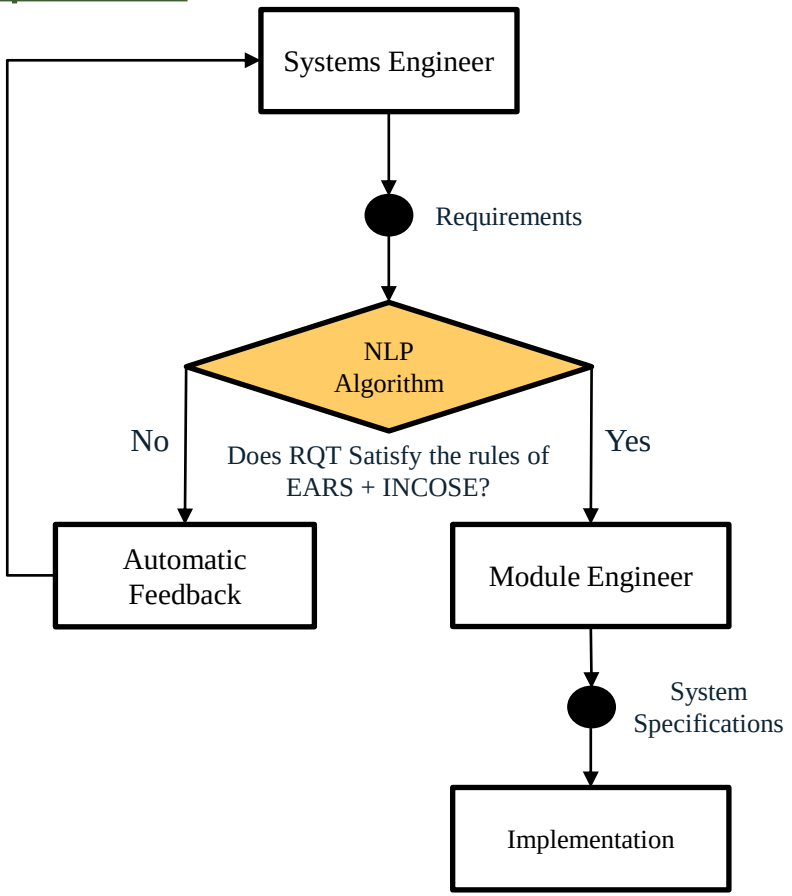
https://spacy.io/

Automatic Requirements Feedback through NLP

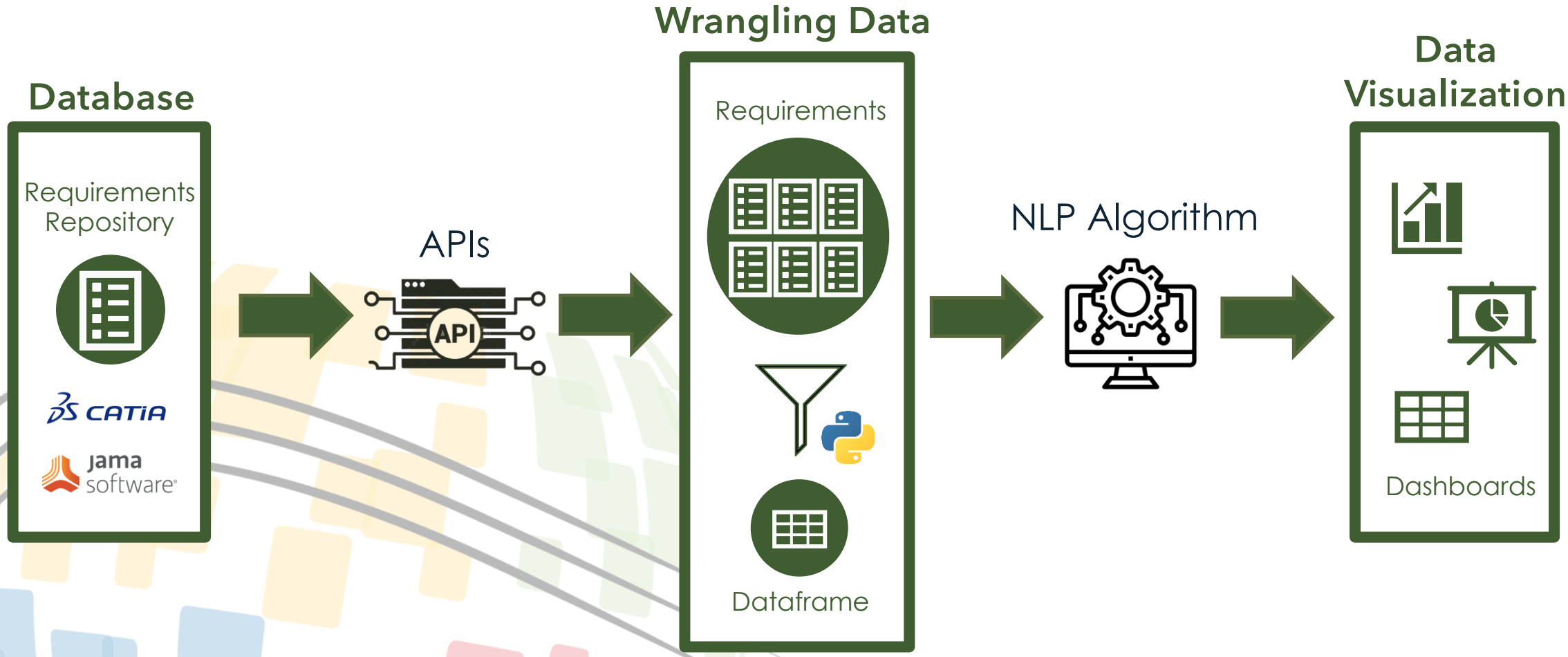
Old Approach



New Approach



Verifying Requirements Methodology



Large Language Model (LLM)

Large Language Model (LLM) is a type of artificial intelligence technology trained on vast datasets, such as ChatGPT and Google Gemini .

LLMs can be trained to narrow down the answer based on *context* and *content*.

Context: *"You are a Systems Engineering tool, and your job is to help write good requirements based on INCOSE rules and EARS template."*

Content: *"Based on the suggestion: "Avoid the use of vague terms such as any", fix the requirement: "When any button is pressed, the feature shall illuminate the way"."*

Automated Correction through LLM

Example

1 Suggestion

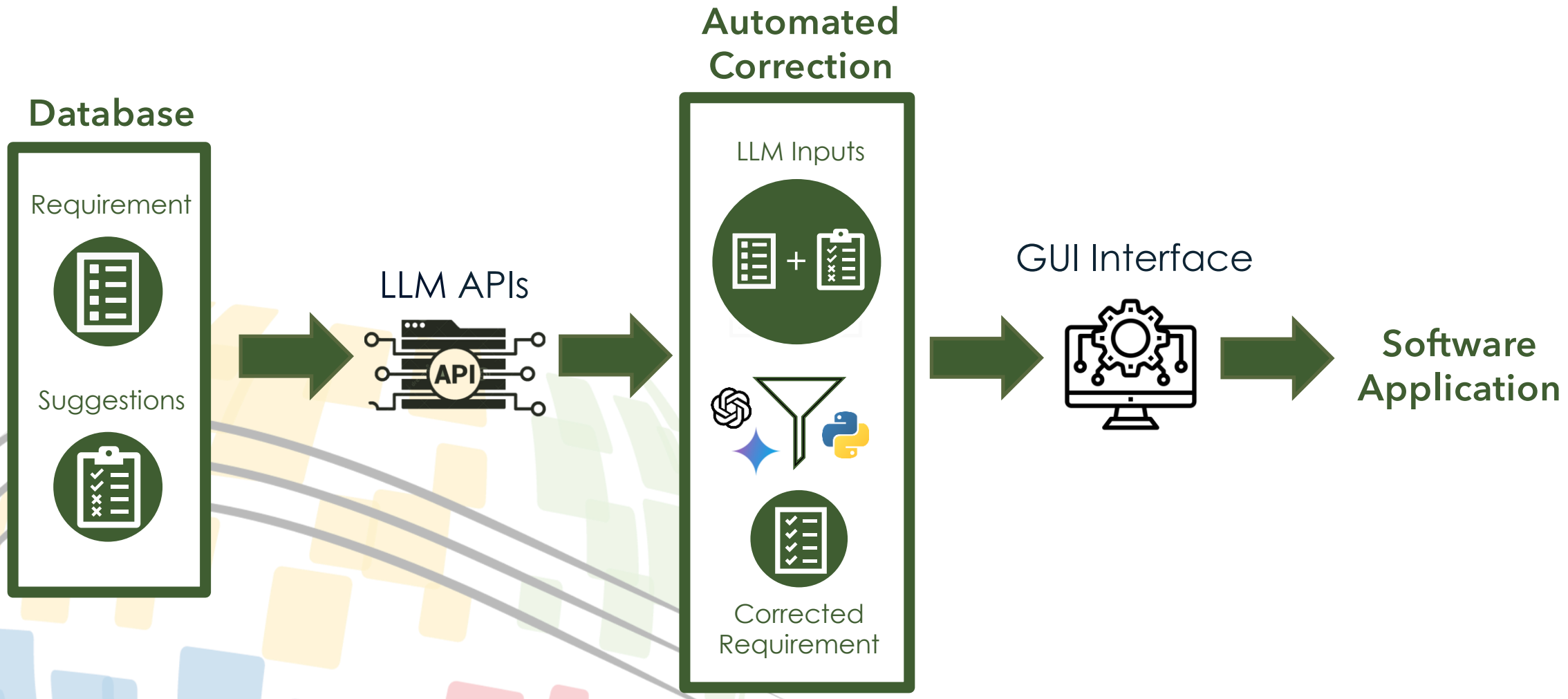
Input: *"Based on the suggestion(s): **"The word "shall" must appear at least once",** fix the requirement: **"When the light button is pressed, the feature illuminate the way"**."*

2 Requirement

3 Generated Correction

Output: "*When the light button is pressed, the feature **shall** illuminate the way.*"

Automated Correction Methodology



LLM Hallucination and Unfaithful Reasoning

As the number of suggestions increases, so does the difficulty of these premises being met by the LLM algorithm, which results in situations of "*hallucination*" and "*unfaithful reasoning*".

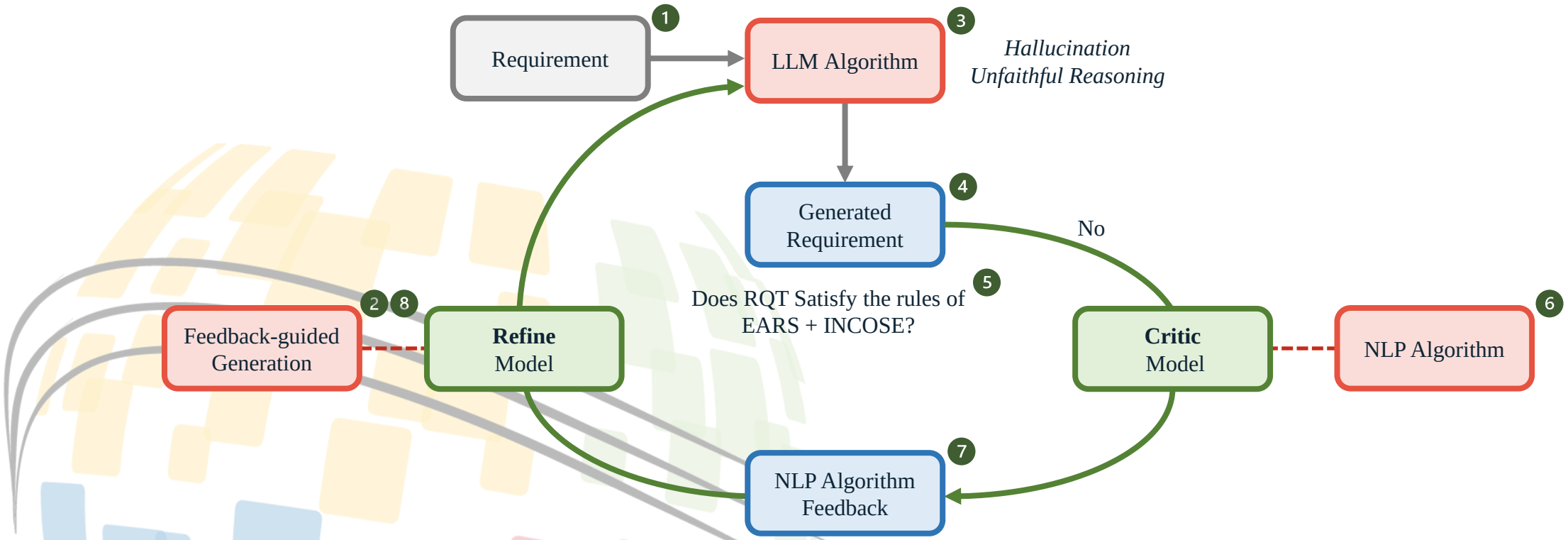
Input: *"Based on the suggestion(s): **"Suggestion 1, Suggestion 2, Suggestion 3, etc."**, fix the requirement: **"Requirement"**."*

Output: ***Requirement*** *with fixed* ***Suggestion 1*** *and* ***Suggestion 2****.*

✗ Suggestion 3 was not fixed.

Critical and Refining Methodology

LLM and NLP techniques can be used to achieve the project success in a cycle of refinement of the requirement correction.



LLM Response Losing Information

We verified that complex requirements **lose information** after being refined by the LLM.

Example

1 Information lost

Requirement Input: *"**Module X is waiting for response** and while the vehicle is on, when the light button is pressed, the feature shall turn on the light."*

Requirement Generated by LLM: *"While the vehicle is on, when the light button is pressed, the feature shall turn on the light."*

Pros: Concise requirement and following EARS and INCOSE ✓

Cons: Module X precondition was lost (waiting for response) ✗

Using BLEU score to get confidence in LLM response

The Bilingual Evaluation Understudy (BLEU) is a metric commonly used in NLP for evaluating Machine-generated sentences similarity to reference sentences.

Example

Requirement Input:
*"**Module X is waiting for response** and while the vehicle is on, when the light button is pressed, the feature shall turn on the light."*

Requirement Generated by LLM:
"While the vehicle is on, when the light button is pressed, the feature shall turn on the light."

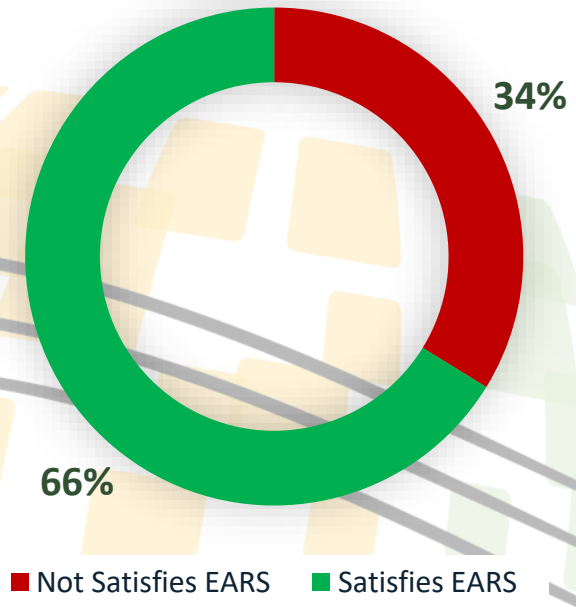
BLEU Score: 0.1142 ✗ - It varies from 0 (Different) to 1 (Similar).

Case Study - Automotive Feature

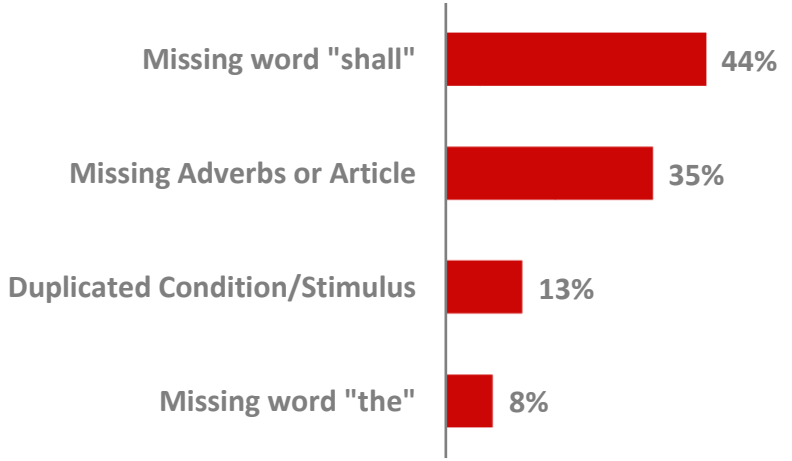
Total of 148 Feature Requirements

Requirements following EARS Template

Easy Approach to Requirements Syntax (EARS)



Breakdown of EARS errors



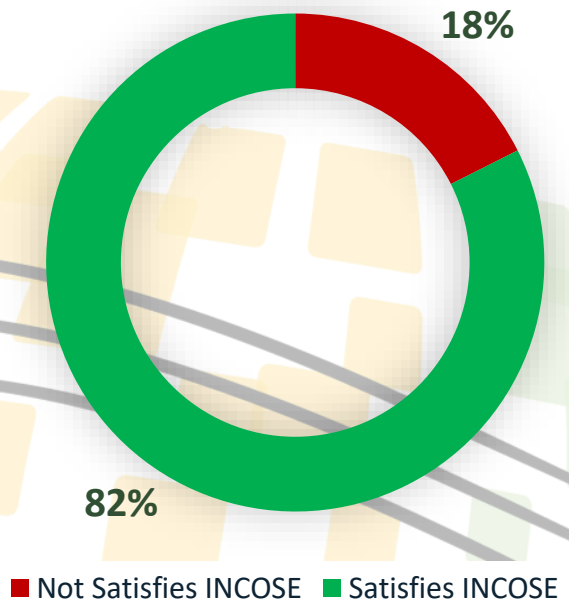
Descriptions:
#1: "shall" must appear exactly once.
#2: Requirement should begin with "when", "while", "where", "if" or "the".
#3: Condition/Stimulus such "when" should not appear more than once.
#4: "the" must appear at least once.

Case Study - Automotive Feature

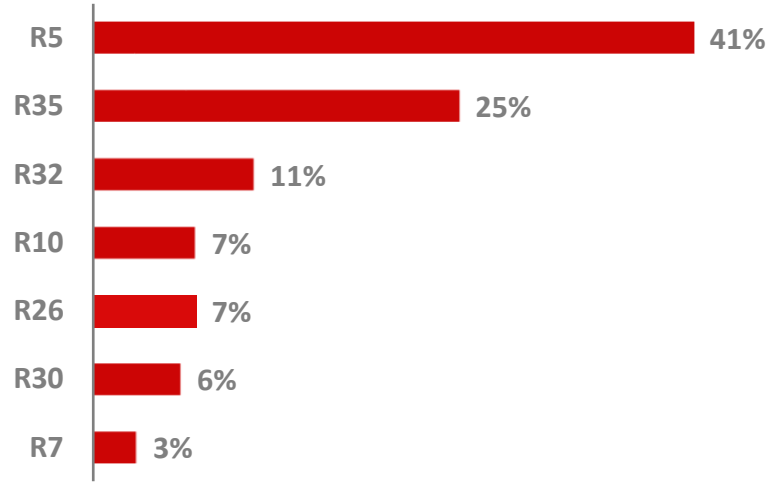
Total of 148 Feature Requirements

Requirements following INCOSE Rules

International Council on Systems Engineering (INCOSE)



Breakdown of INCOSE errors

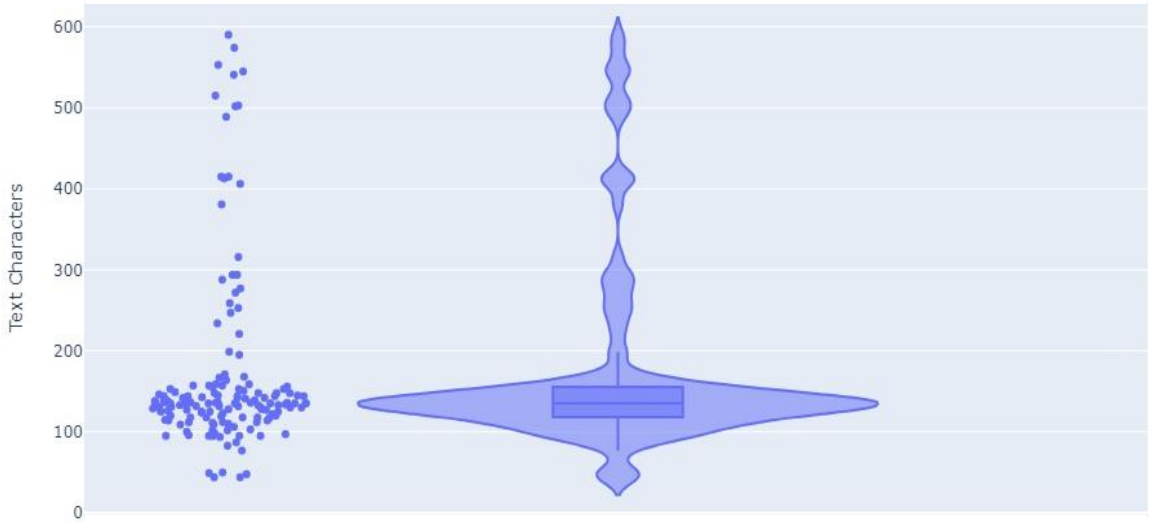


Descriptions:
#R5: Use definite article rather than indefinite.
#R7: Avoid the use of vague terms.
#R10 : Avoid superfluous infinitives.
#R26: Avoid using unachievable absolutes.
#R30: Express each need and requirement once and only once.
#R32: Use "each" instead of "both" when universal quantification is intended.
#R35: Define temporal dependencies explicitly instead of using indefinite temporal keywords.

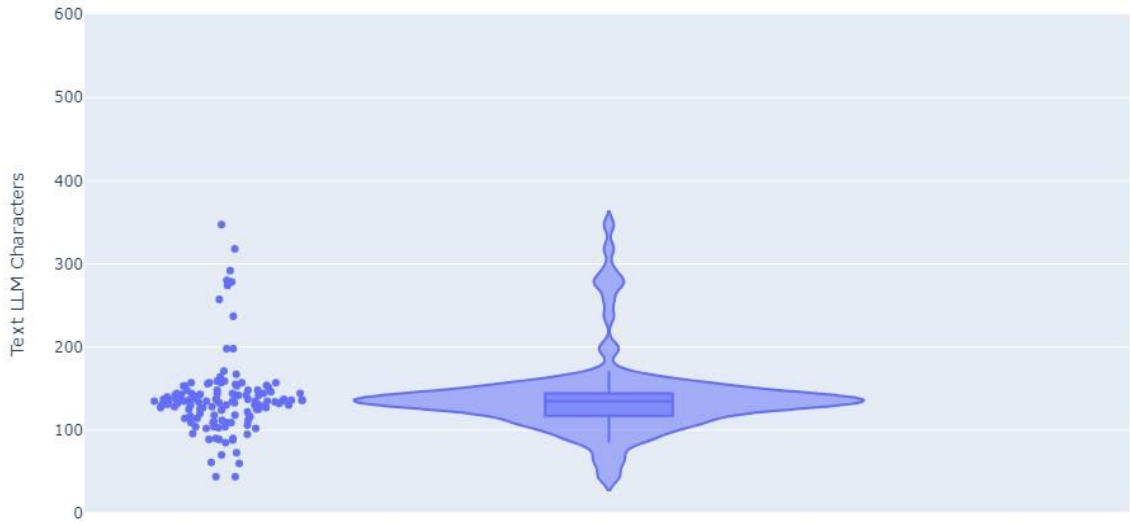
Case Study - Automotive Feature

Number of Characters

Requirements Created by Engineer



Requirements Created by LLM

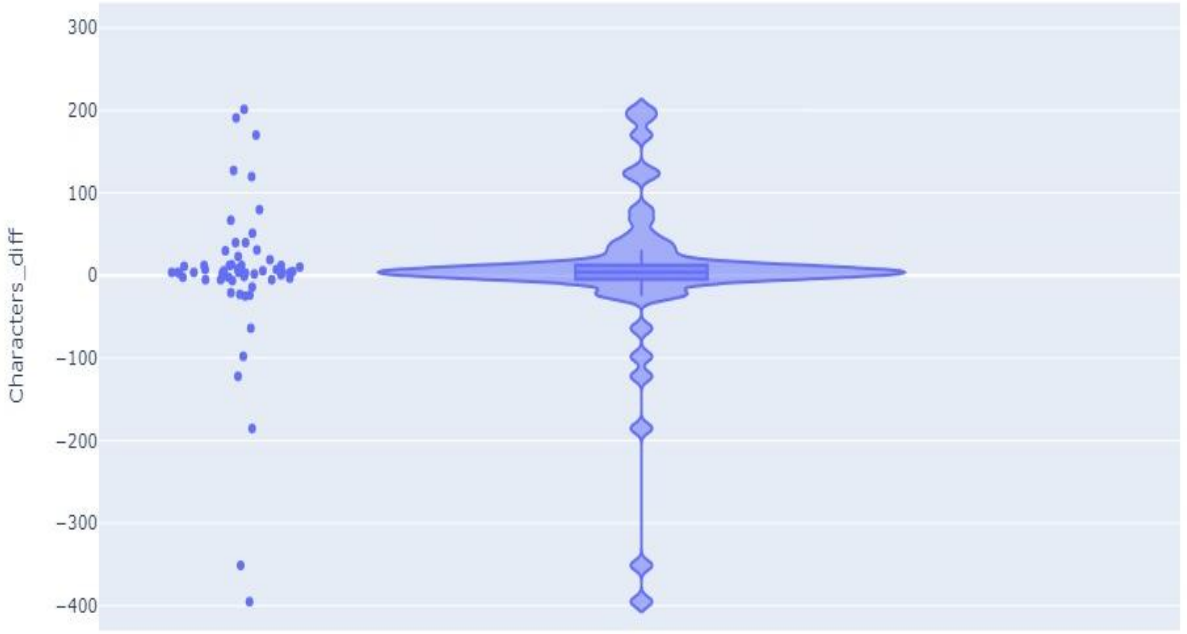


The LLM has a natural tendency to reduce the amount characters in requirement to comply with EARS and INCOSE standards, trying to make it more concise or atomic.

Case Study - Automotive Feature

Number of Characters / BLEU Score

BLEU Score (All requirements generated)
0.4461
BLEU Score (Requirements with difference up to 5 characters)
0.5915



The BLEU score increases when the difference between the characters of the engineer-created and LLM-created requirements decreases.

Calibrating the Refine Model

The BLEU score was used as a threshold in conditions to verify the quality of the LLM response.

Condition 1: *If: BLEU score < 0.6 and Sentences in requirement > 2.*

Condition 2: *If BLEU score < 0.6 and Difference of characters > 50.*

Condition 3: *If Sentences in requirement > 2 and Difference of characters > 50.*

Results # Case Study - Automotive Feature

Among 56 of 148 requirements that were corrected by the LLM algorithm, 6 lost information or were not clear enough, losing the message that should be transmitted in that requirement.

- *The performance of the requirements reworked by the LLM was around **89.3%**.*
- *The requirements created by the LLM are **100% comply with EARS and INCOSE rules**.*

The application of LLM for requirements correction has shown a significant contribution, but it is highly recommended that Systems Engineers carefully review their requirements throughout the product development process to prevent the loss of information.

Impact Analysis of using Natural Language Processing and Large Language Model on Automated Correction of Systems Engineering Requirements

Q&A

Thank you!

Arthur Oliveira
Feature Systems Engineer
aoliv355@ford.com