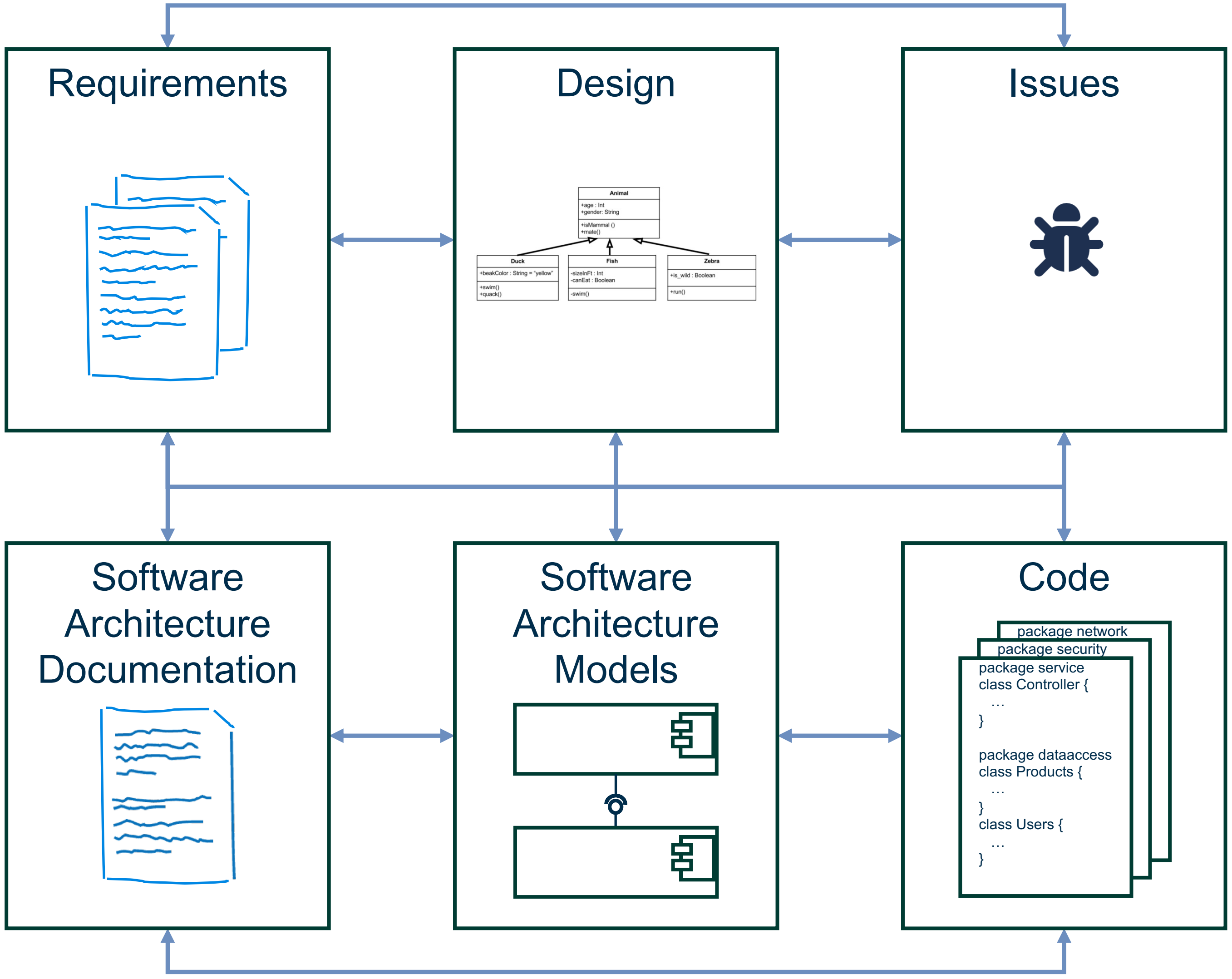
LiSSA: Toward Generic Traceability Link Recovery through Retrieval-Augmented Generation

Dominik Fuchß, Tobias Hey, Jan Keim, Haoyu Liu, Niklas Ewald, Tobias Thirolf, Anne Koziolek, KASTEL – Institute of Information Security and Dependability

What makes Trace Links important?



```
package network
package security
package service
class Controller {
   …
}

package dataaccess
class Products {
   …
}
class Users {
   …
}
```

Trace links are evidently useful for

- Software Maintenance
- Bug Localization
- Change Impact An.
- System Security
- …

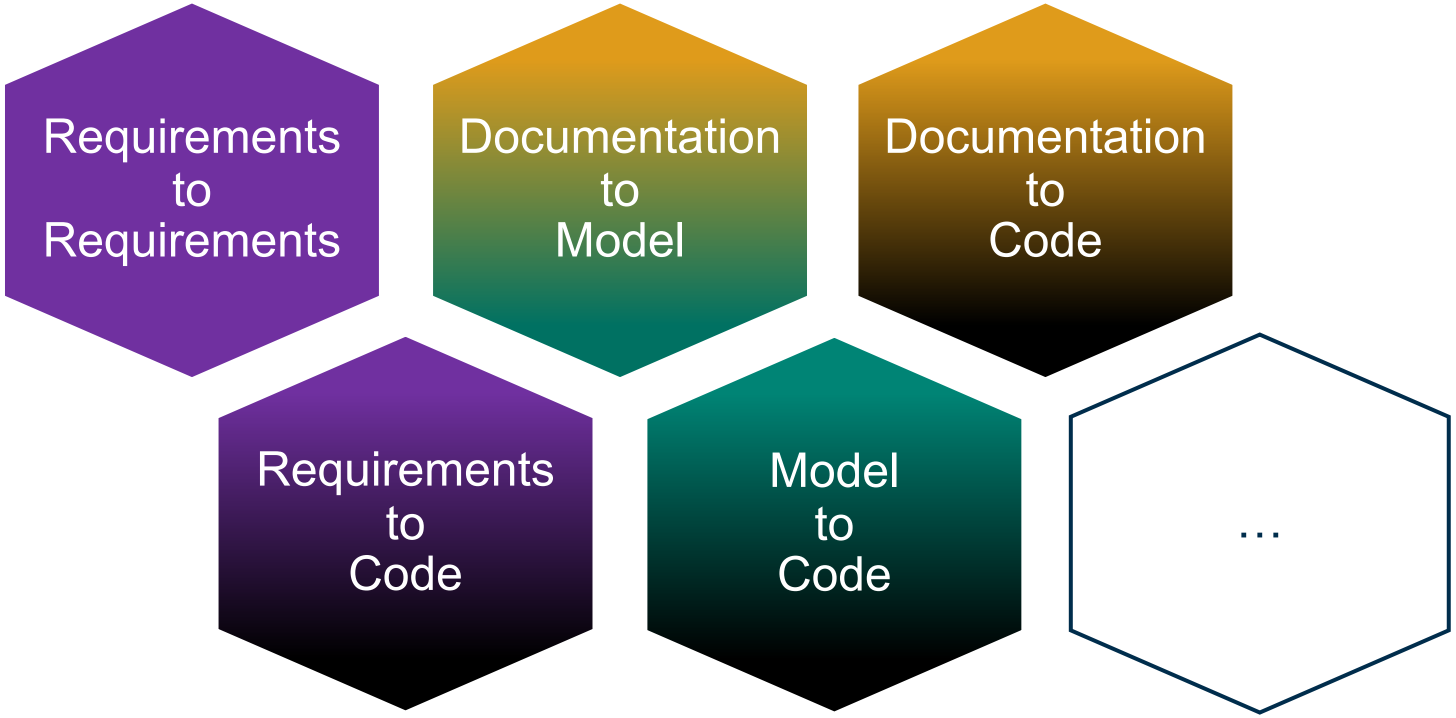
□ Artifact

↔ Relation

What makes Trace Links important?

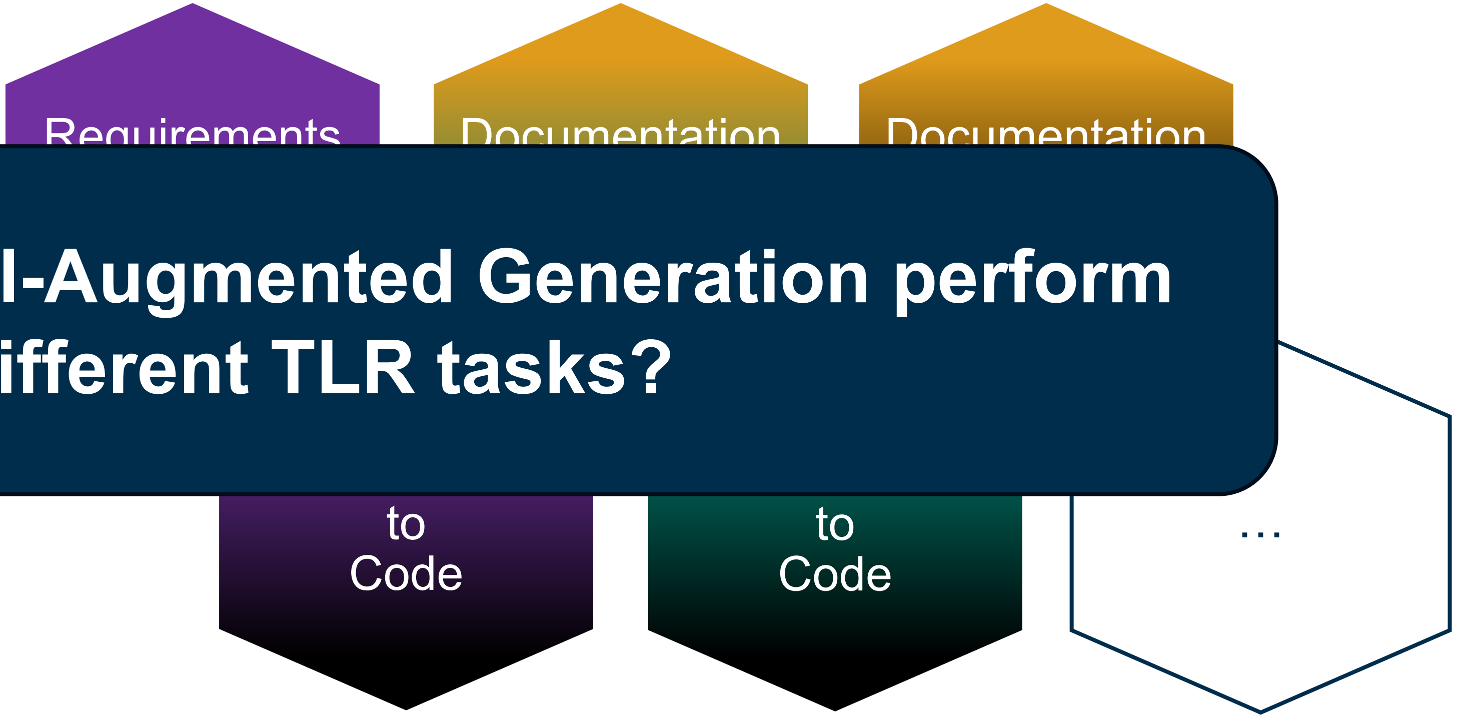
- Multitude of different artifacts
- Typical TLR tasks:
  - Requirements to Code
  - Documentation to Code
  - …

- Many specialized approaches



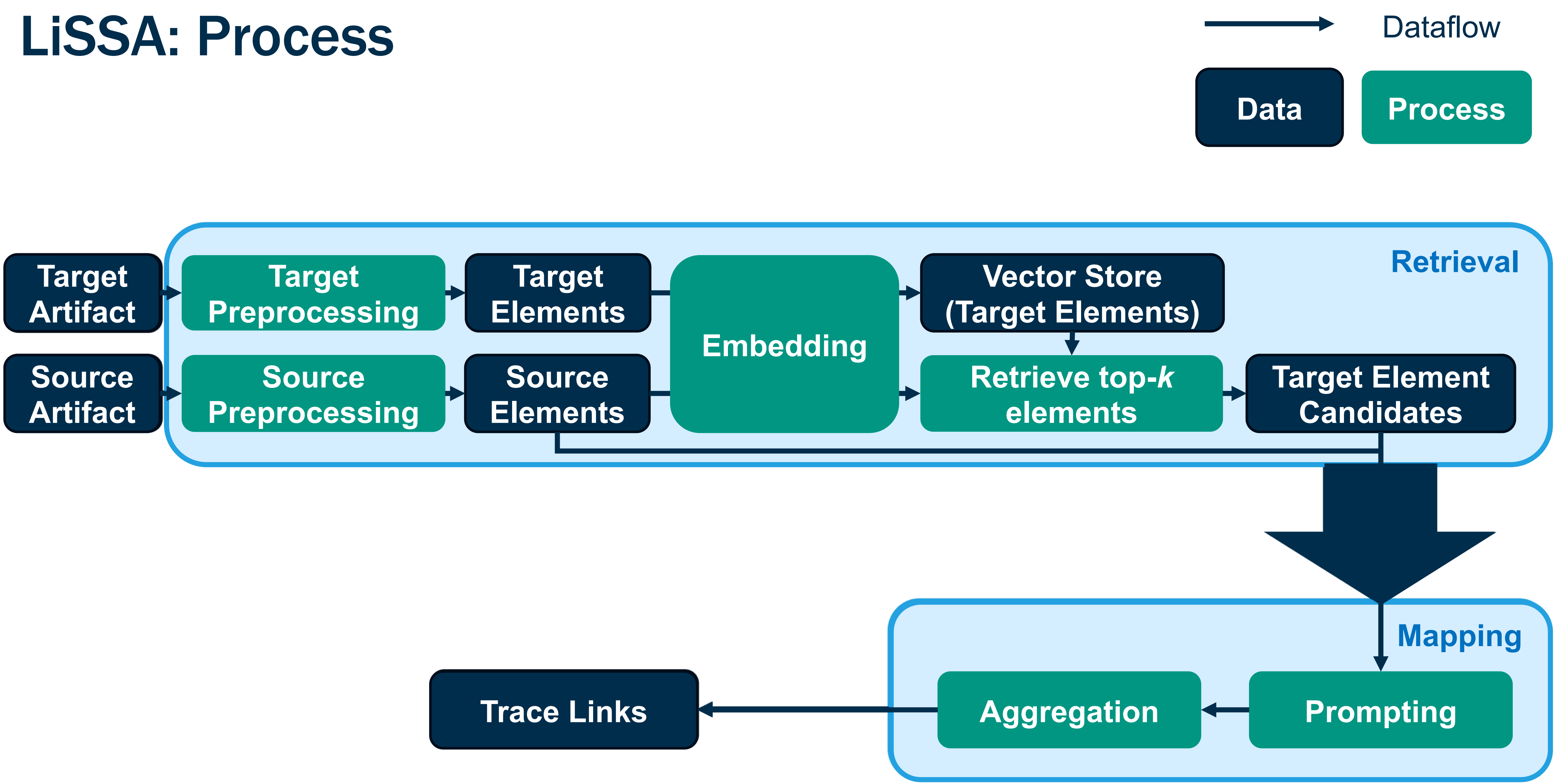
What makes Trace Links important?

- Multitude of different artifacts
- Typical TLR tasks:
  - Requirements to Code
  - Doc
  - …
- Many



How does Retrieval-Augmented Generation perform for different TLR tasks?

LiSSA: Process



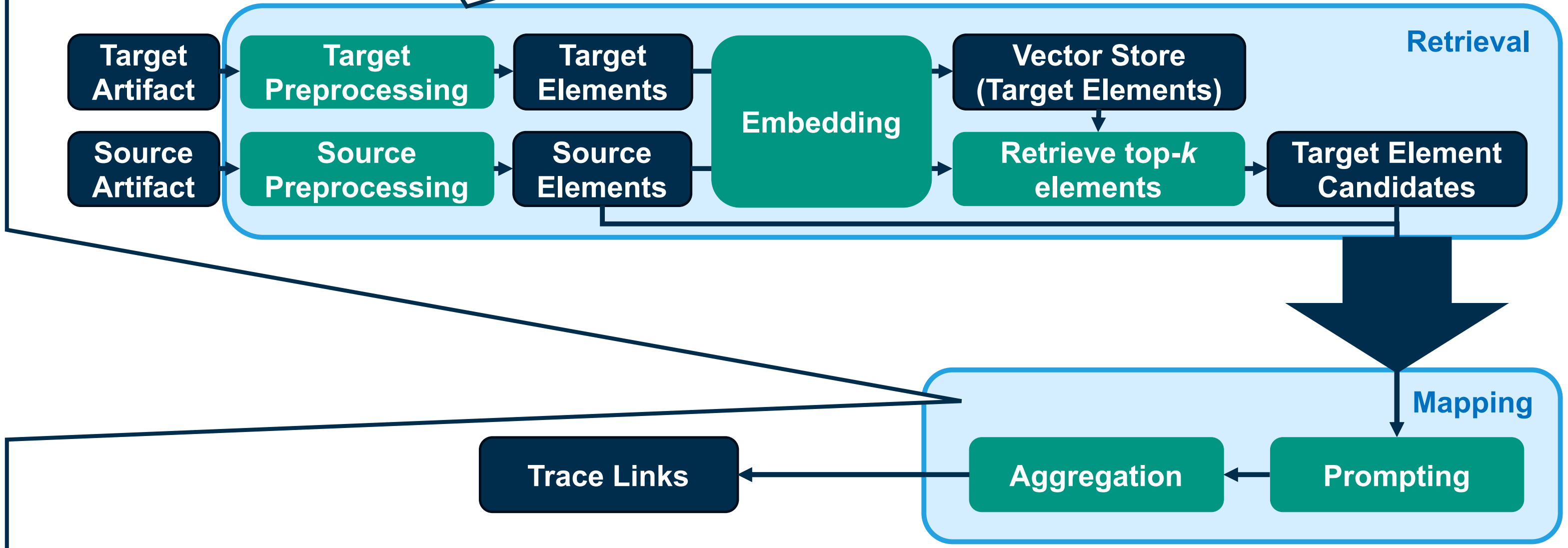
Evaluation

Preprocessing: Extract *elements* from *artifacts*

- No preprocessing
- Code chunking (fixed size)
- Code method splitting
- Model element extraction
- Sentence splitting

Prompting: Classify whether elements belong to each other

- No prompt:
  Always classify as ”trace link” → IR baseline
- KISS:
  Simple Yes/No-classification task (zero shot)
- Chain-of-thought:
  Zero shot prompt + request for reasoning



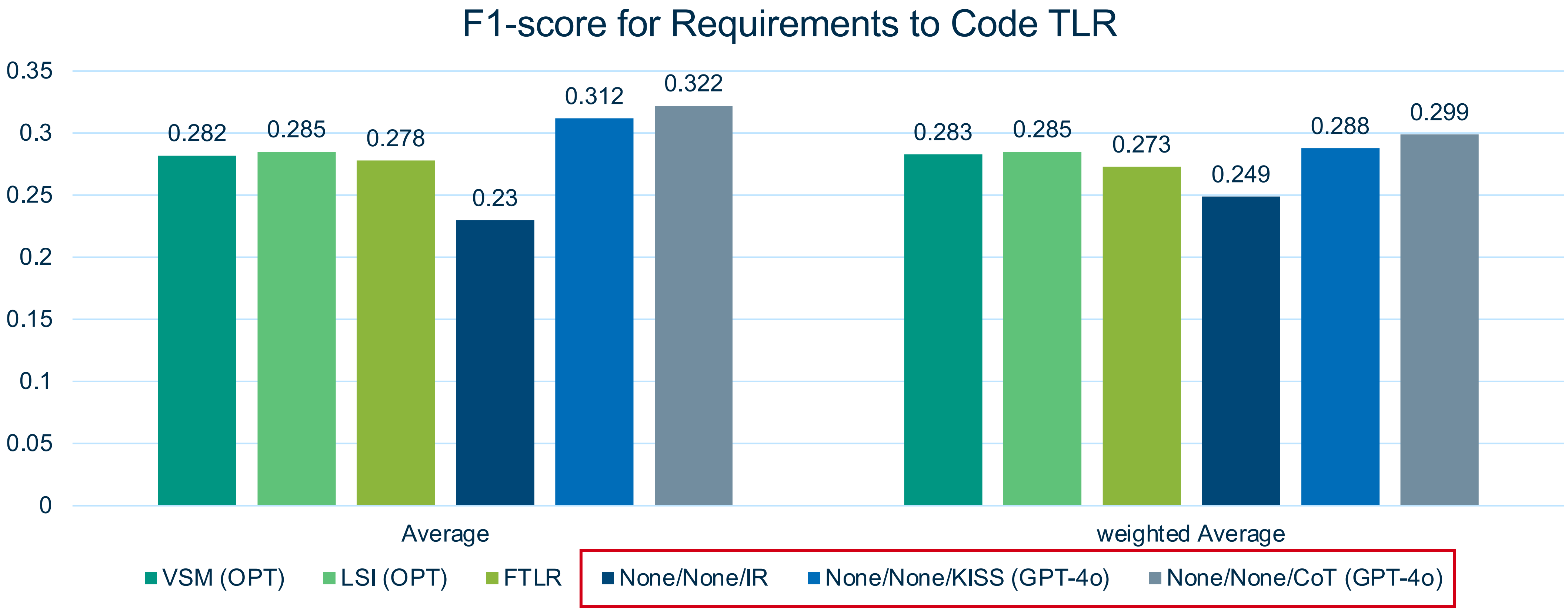
Evaluation

Requirements to Code TLR

Documentation to Code TLR

Documentation to Model TLR

Evaluation: Requirements to Code

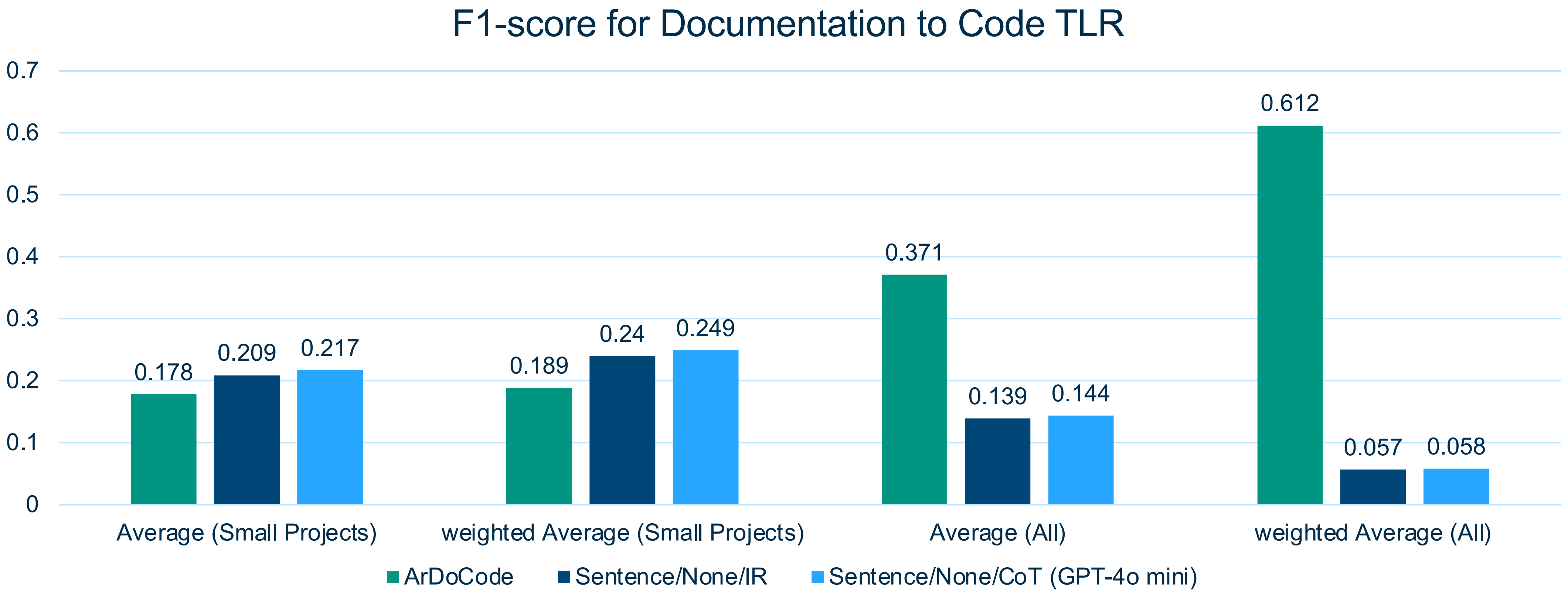


Requirements to Code TLR

Documentation to Code TLR

Documentation to Model TLR

Evaluation: Documentation to Code

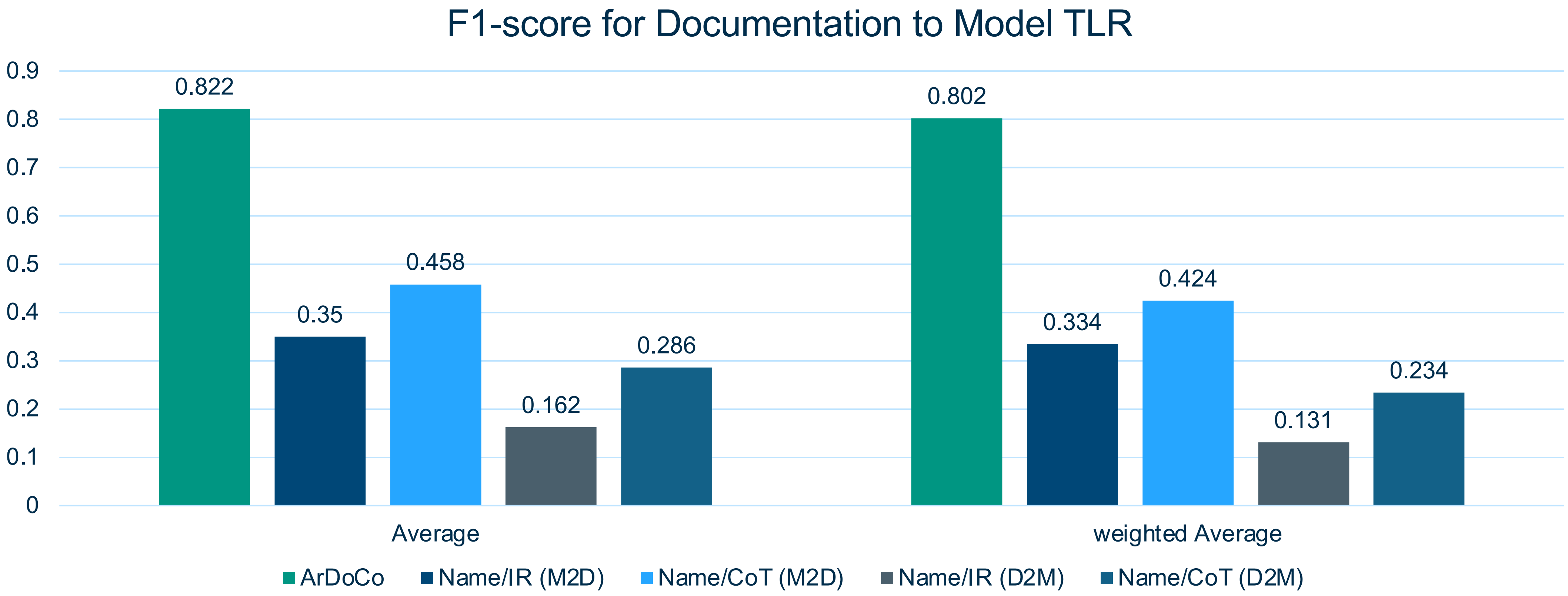


Requirements to Code TLR

Documentation to Code TLR

Documentation to Model TLR

Evaluation: Documentation to Model



Requirements to Code TLR

Documentation to Code TLR

Documentation to Model TLR

Conclusion

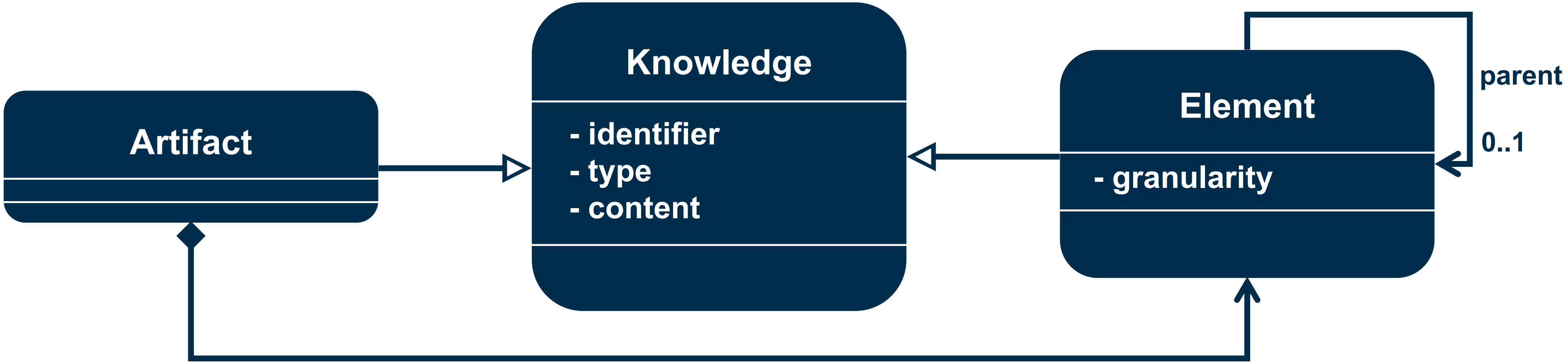
- We presented LiSSA, a generic TLR framework that uses RAG
- In the evaluation,
  - Our approach can significantly outperform state-of-the-art for requirement to code TLR – avg. F1: 0.278 (FTLR) vs. 0.322 (GPT-4o + CoT)
  - Chain-of-thought (CoT) prompting was on average more effective than simple classification prompting
  - Artifact-to-artifact TLR was (on average) better than fine-grained mappings
- Outlook:
  - Inter-requirements TLR with LiSSA (see https://ardoco.de/c/refsq25)
  - Documentation to code TLR (see https://ardoco.de/c/icsa25)
  - Definition of different kinds of “Trace Links”
  - Revisit fine-grained mappings + advanced aggregation strategies
  - Automatic prompt engineering

ardoco.de/c/icse25

Backup

LiSSA: Concepts

- Similar treatment of similar artifacts
  - Code-like artifacts (e.g., source code, test code)
  - Natural language artifacts (e.g., requirements, documentation, issues)
  - Structural model artifacts (e.g., UML component models)



Research Questions

- RQ1: Performance of RAG-based TLR compared to SotA
  - Significantly outperforming for requirements to code TLR
  - Documentation to Code: Better performance on smaller projects (less TLs)
  - Documentation to Architecture: No outperformance

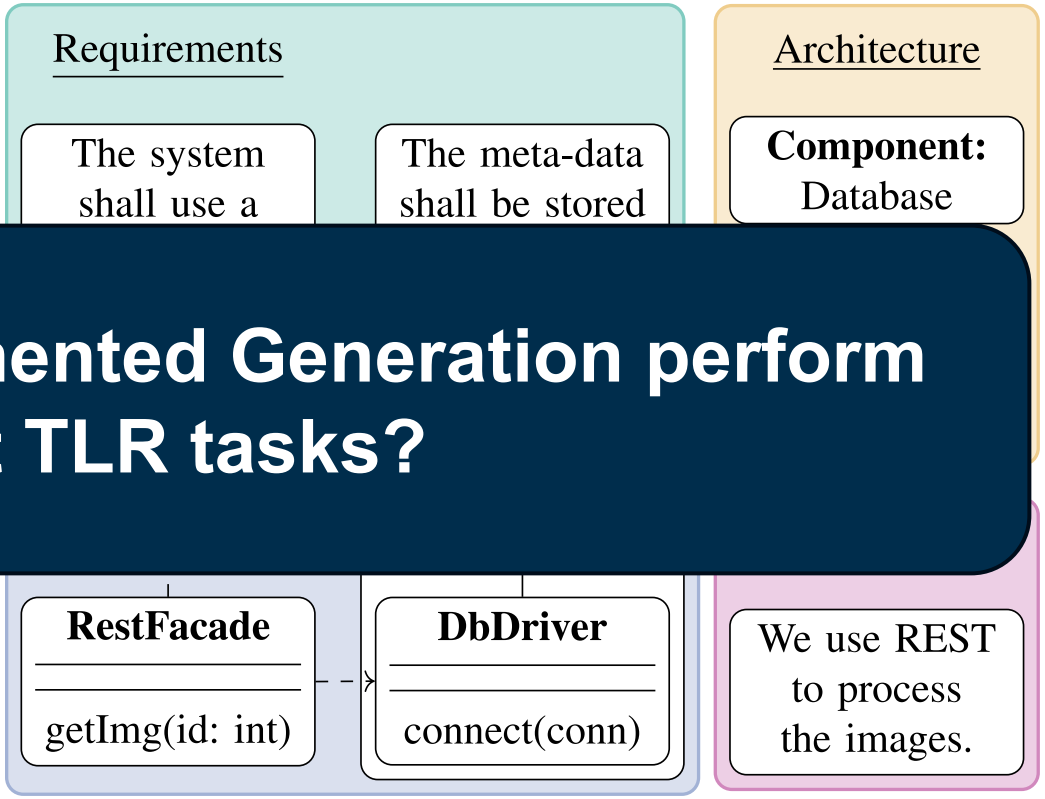
- RQ2: Effectiveness of CoT prompting
  - CoT performs better than simple classification prompts

- RQ3: Preprocessing Techniques
  - On average, not benefitial to split artifacts
  - However, on some projects this helps a lot

- RQ4: Effects of classification step compared to IR-only
  - Classification improves TLR

What makes Trace Links important?

- Multitude of different artifacts
- Typical TLR tasks:
  - Requirements to Code
  - Doc
  - Re
  - …
- Many



How does Retrieval-Augmented Generation perform for different TLR tasks?

Evaluation: Requirements to Code (Datasets)

Dataset	Domain	NL	Programming	Requirements	Code	TLs
SMOS	Education	IT	Java	67	100	1044
eTour	Tourism	EN	Java	58	116	308
iTrust	Healthcare	EN	Java	131	226	286
Dronology (RE)	Aerospace	EN	Java, Python	99	423	602
Dronology (DD)	Aerospace	EN	Java, Python	211	423	740

Evaluation: Requirements to Code (GPT-4o, F_1-score)

Approach	SMOS	eTour	iTrust	Dronology (RE)	Dronology (DD)	Average	Weighted Average
VSM_{OPT}	0.422	0.483	0.217	0.158	0.131	0.282	0.283
LSI_{OPT}	0.422	0.453	0.253	0.162	0.135	0.285	0.285
FTLR	0.389	0.474	0.222	0.172	0.140	0.278	0.273
None/None/IR	0.366	0.342	0.105	0.196	0.144	0.230	0.249
None/None/KISS	0.285	0.493	0.290	0.260	0.229	0.312	0.288
None/None/CoT	0.294	0.526	0.276	0.273	0.241	0.322	0.299

Prompts

Prompt: KISS

Question: Here are two parts of software development artifacts.

{source_type}: '''{source_content}'''
{target_type}: '''{target_content}'''

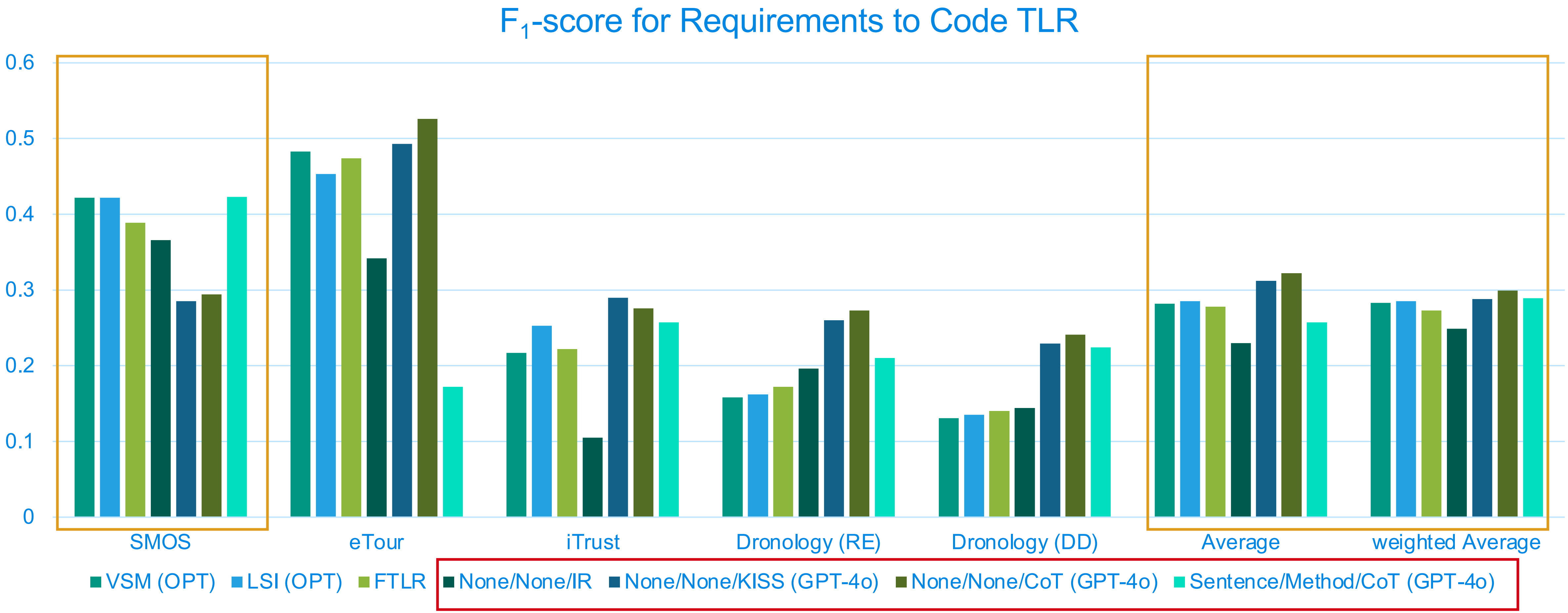
Are they related?
Answer with 'yes' or 'no'.

Prompt: CoT

Below are two artifacts from the same software system. Is there a traceability link between (1) and (2)? Give your reasoning and then answer with 'yes' or 'no' enclosed in <trace></trace>.

(1) {source_type}: '''{source_content}'''
(2) {target_type}: '''{target_content}'''

Evaluation: Requirements to Code



F$_1$-score for Requirements to Code TLR