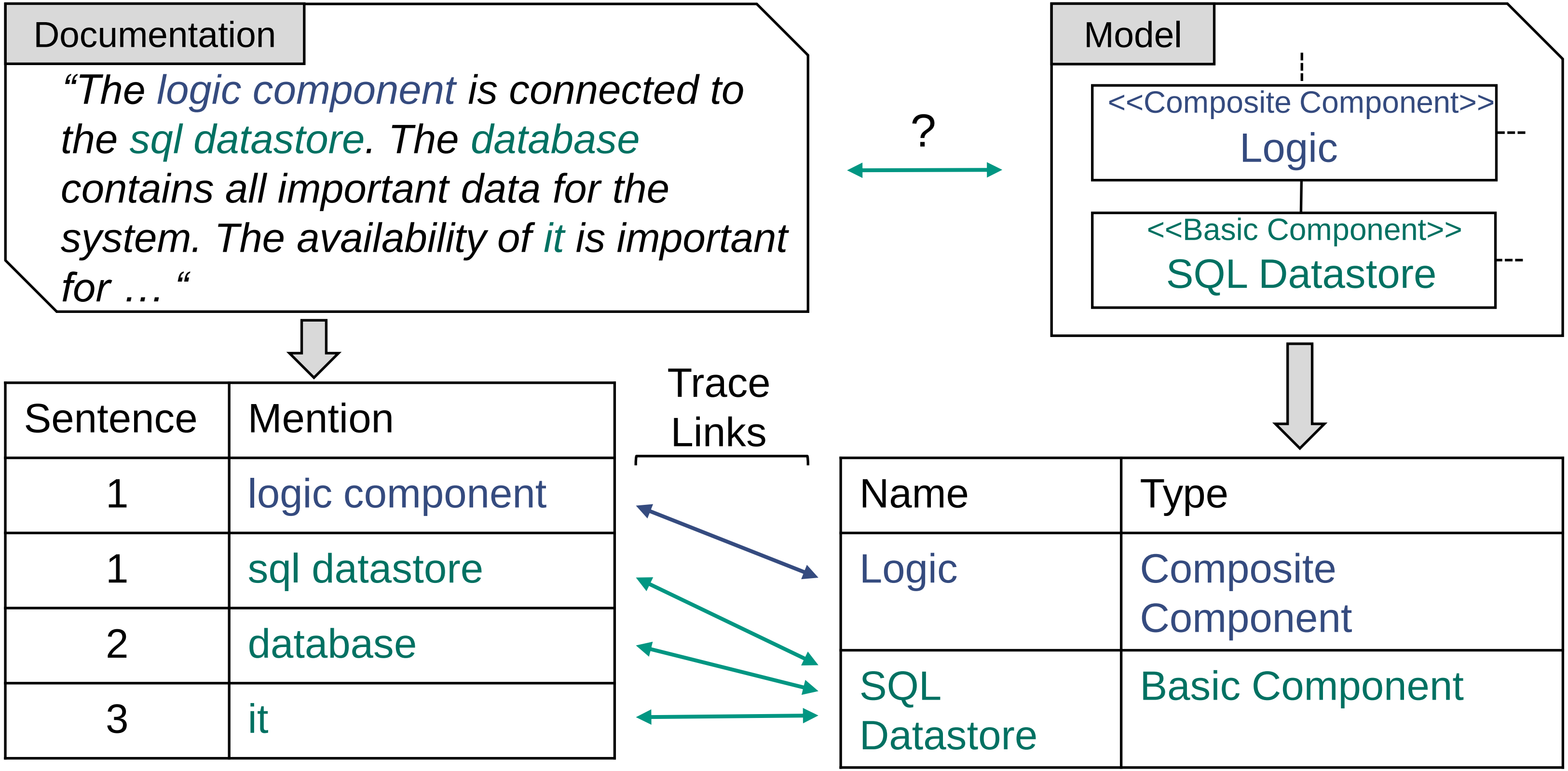
Trace Link Recovery for Software Architecture Documentation

Jan Keim, Sophie Schulz, Dominik Fuchß, Claudius Kocher, Janek Speit,
and Anne Koziolek



Documentation

“The logic component is connected to the sql datastore. The database contains all important data for the system. The availability of it is important for … “

?

Model

<<Composite Component>>
Logic

<<Basic Component>>
SQL Datastore

Trace Links

Sentence	Mention
1	logic component
1	sql datastore
2	database
3	it

Name	Type
Logic	Composite Component
SQL Datastore	Basic Component

Existing approaches

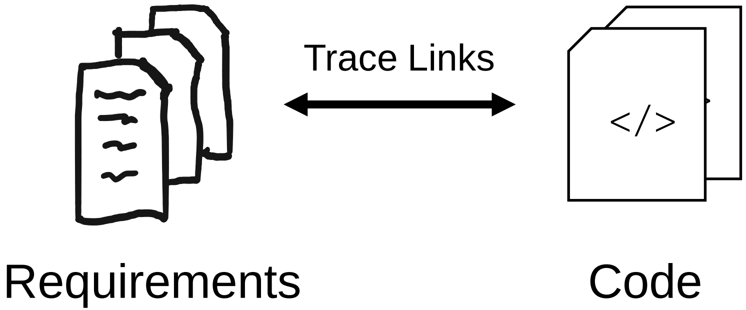
IR-based

[Rodriguez20]

NLP-based

[Zhang16]

[Guo17] [Moran20]



Trace Links

Requirements

Code

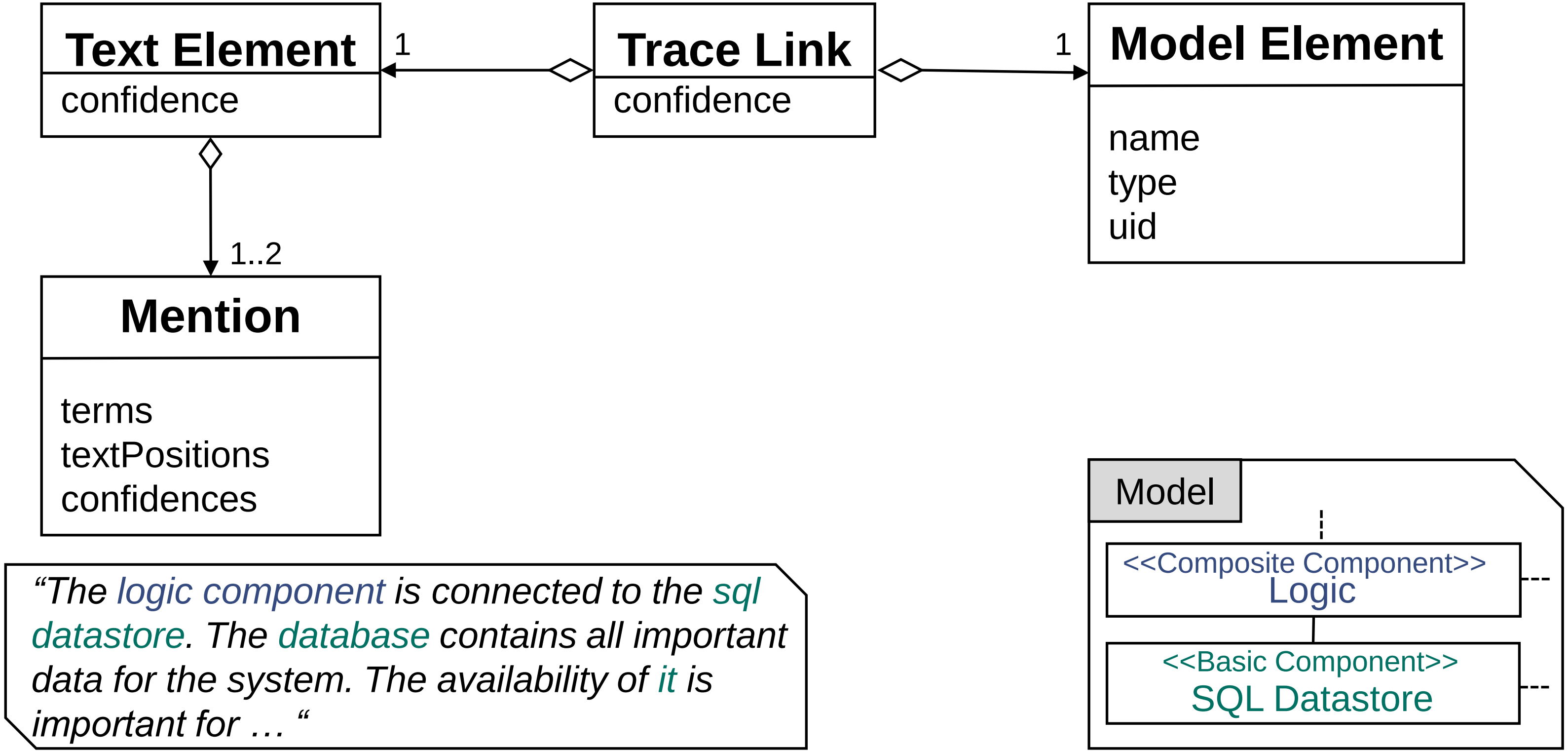
Research Question and Contributions

- How accurately can we identify trace links between textual informal SAD and formal models with our approach?

- Contributions
  - Extendable framework SWATTR (SoftWare Architecture Text Trace link Recovery) for creating such trace links
  - Data sets for evaluation
  - (Re-) Implementations of baseline approaches for comparison
  - Code, evaluation results, and reproduction packages online

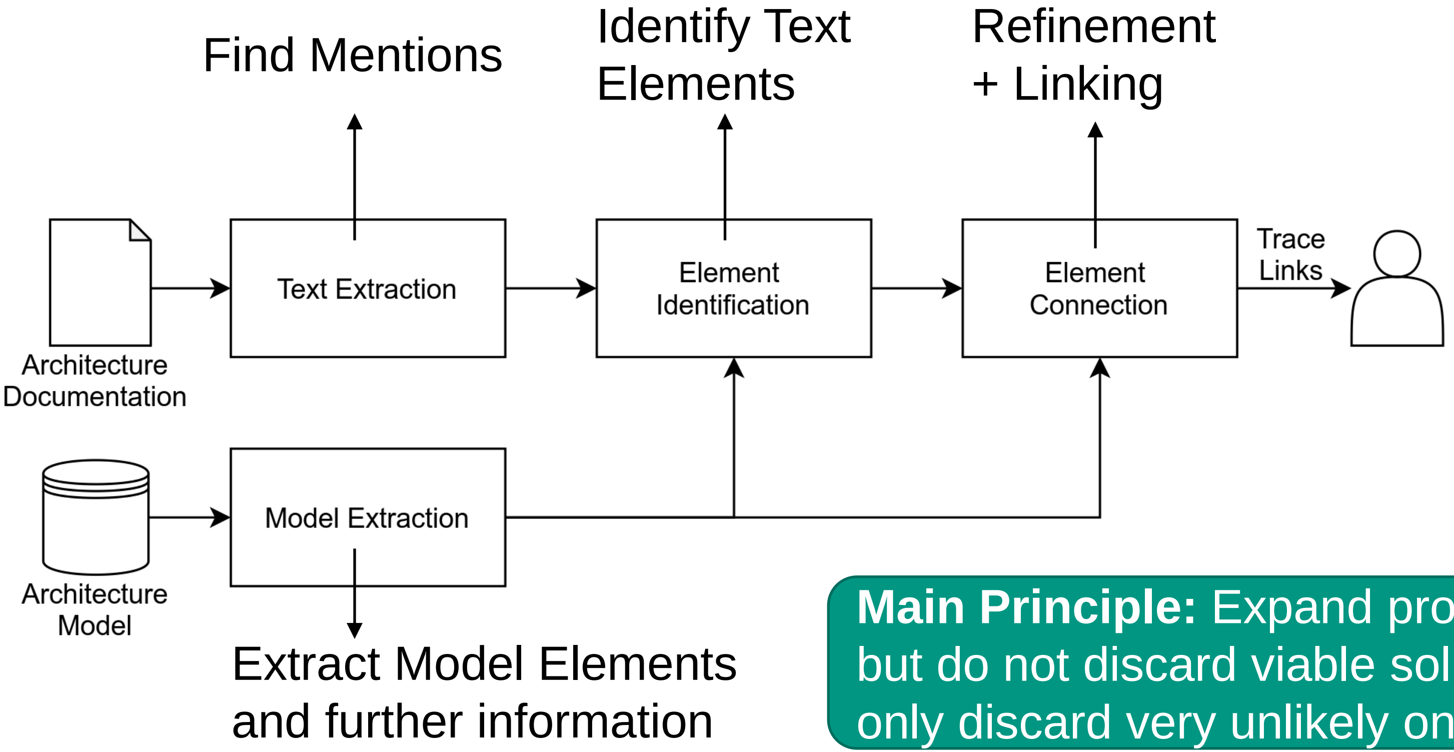
Artifacts at https://zenodo.org/record/4767470

Our Trace Link Model



“The logic component is connected to the sql datastore. The database contains all important data for the system. The availability of it is important for … “

SWATTR Overview



Find Mentions

Identify Text Elements

Refinement + Linking

Architecture Documentation

Text Extraction

Element Identification

Element Connection

Trace Links

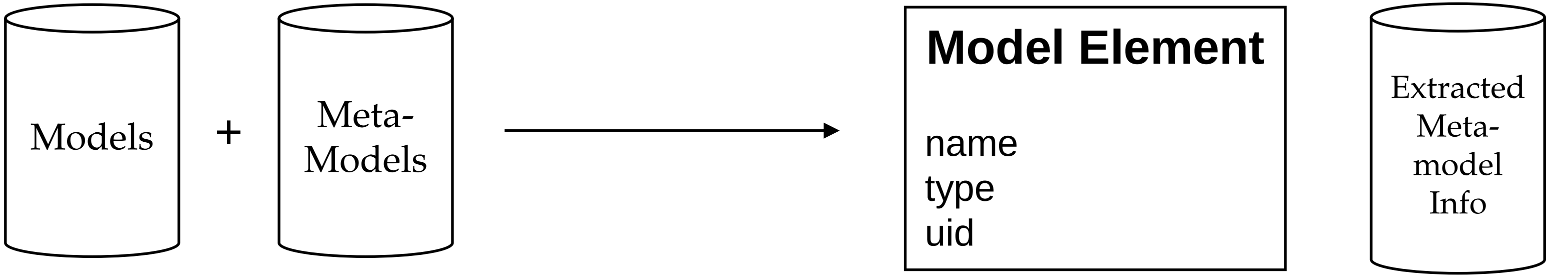
Architecture Model

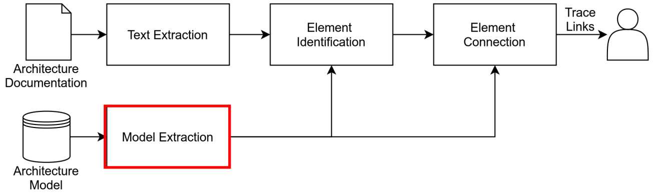
Model Extraction

Extract Model Elements and further information

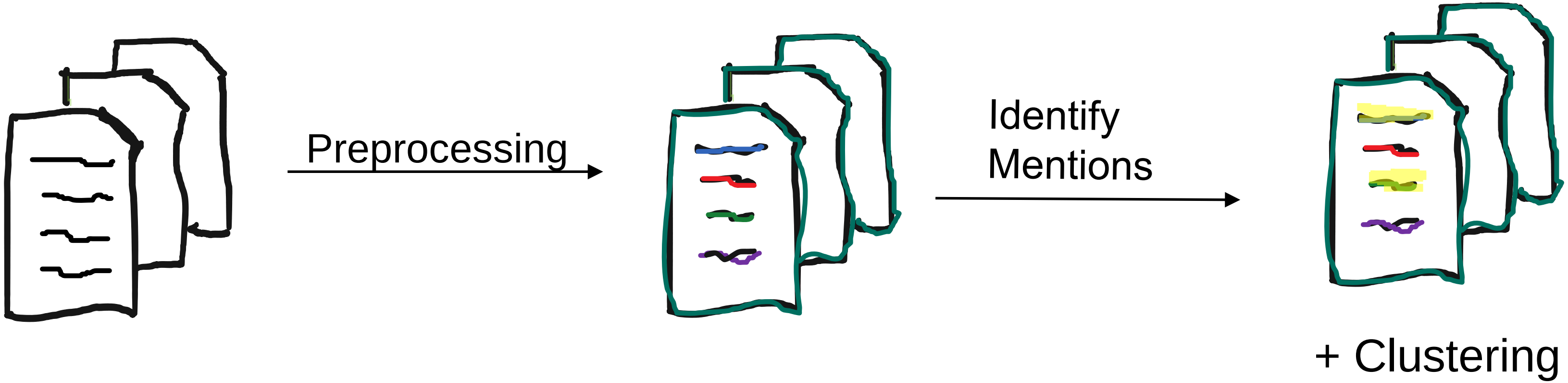
Main Principle: Expand promising results, but do not discard viable solutions too early; only discard very unlikely ones

Extracting Model Information



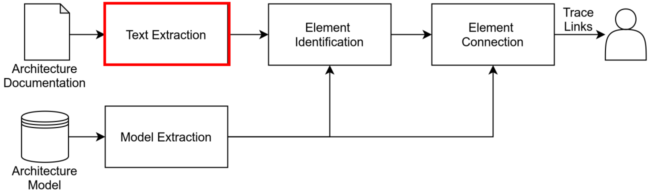


Extracting Text Information

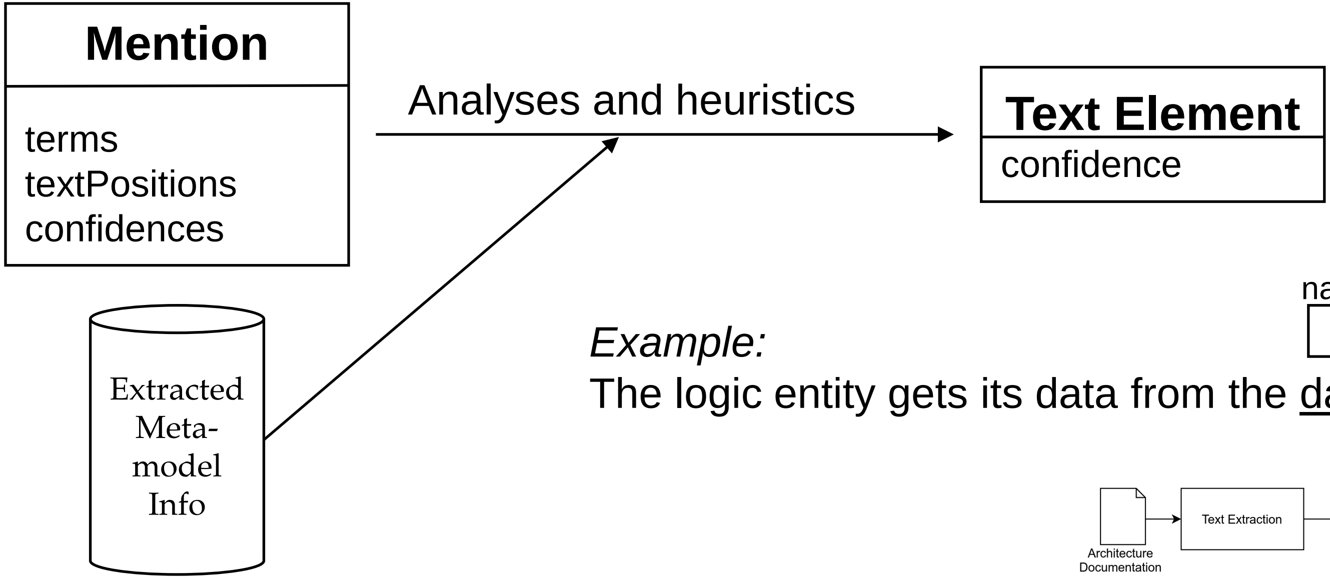


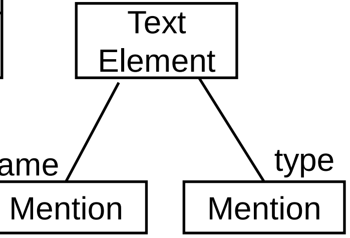
Analyses and Heuristics that contribute:

- Nouns and Plural Nouns
- Incoming/Outgoing Dependencies
- Pattern search
- Separators and Compound Terms

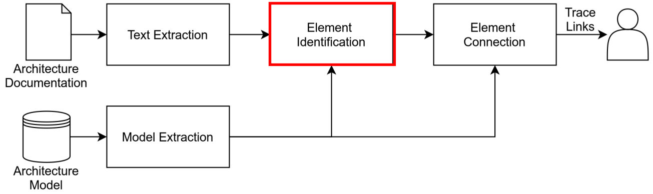


Identifying (Text) Elements

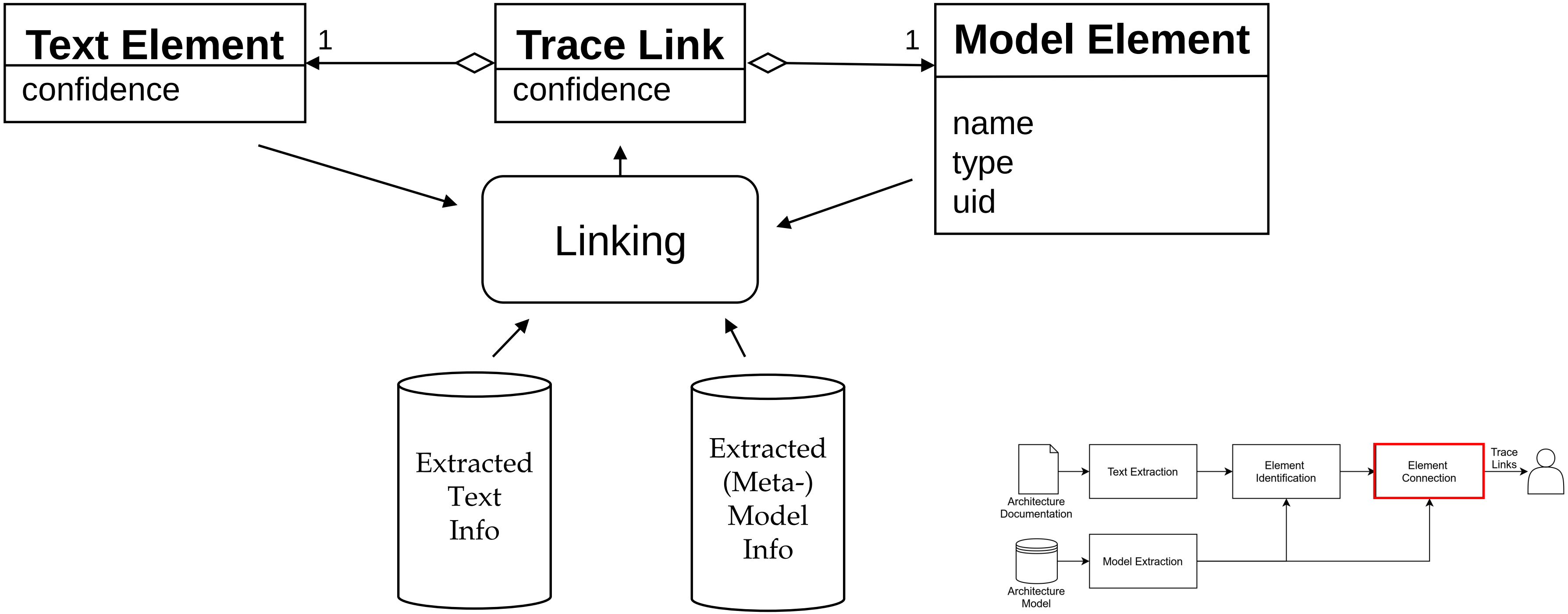




Example:
The logic entity gets its data from the <u>database</u> <u>component</u>



Connect Elements & Create Trace Links



Evaluation Setup

- Metrics: Precision, Recall, F1-Score
- Recall is more important than precision
- Comparison with two approaches

Measure	Acceptable	Good	Excellent
Recall	.60 - .69	.70 - .79	.80 - 1.0
Precision	.20 - .29	.30 - .49	.50 - 1.0

[Hayes06]

	Mediastore	TeaStore	Teammates
No. Sentences	37	43	198
No. Trace Links	25	25	80
No. Model Elements	14	13	8
Max TL per Sentence	2	2	7
Sentences Without TL	13 (35%)	22 (51%)	131 (66%)

Evaluation Results

	Mediastore			TeaStore			Teammates			Average			Weighted Avg.		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SWATTR	.47	**.60**	.53	**.63**	**.88**	**.73**	**.69**	**.89**	**.78**	**.60**	**.79**	**.68**	**.64**	**.83**	**.72**
Rodr.&C.	.07	.32	.12	.10	.20	.13	.10	.15	.12	.09	.22	.13	.10	.19	.13
Zhang et al	**.76**	.52	**.62**	.35	.28	.31	.49	.30	.37	.53	.37	.44	.52	.34	.41

Identified Weaknesses

- Wrongly identified terms when terms have close naming to model elements
  - E.g. *image* and *image components*
- Model elements have similar naming
  - E.g. *MediaManagement* and *UserManagement*
- Short terms and abbreviations cause problems
  - E.g. *GAE* instead of *Google App Engine (GAE) Database*

→ Dependent on configurations, needs configuration per project based on identified properties

Discussion, Conclusion & Future Work

- Results are good to excellent and outperform baseline approaches

- Limitation: linking is reliant on word similarity
  - Assumption that naming should be consistent

- Future Improvements and other work
  - Improve linking, less reliance on word similarity
  - Take relations into account
  - Combine results of different approaches to improve results
  - Find inconsistencies

Artifacts at
https://zenodo.org/record/4767470

References

- **[Hayes06]** Hayes, J.H., Dekhtyar, A., Sundaram, S.K.: Advancing candidate link generation for requirements tracing: The study of methods. IEEE Transactions on Software Engineering 32(1), 4 (2006)
- **[Rodriguez20]** Rodriguez, D.V., Carver, D.L.: Multi-objective information retrieval-based NSGA-II optimization for requirements traceability recovery. In: 2020 IEEE EIT. pp.271-280. https://doi.org/10.1109/EIT48999.2020.9208233, ISSN: 2154-0373
- **[Zhang16]** Zhang, Y., Wan, C., Jin, B.: An empirical study on recovering requirement-to-code links. In: 17th IEEE/ACIS SNPD. pp. 121-126 (2016)
- **[Guo17]** Guo, J., Cheng, J., Cleland-Huang, J.: Semantically enhanced software traceability using deep learning techniques. In: 2017 IEEE/ACM 39th ICSE. pp. 3-14 (2017)
- **[Moran20]** Molenaar, S., Spijkman, T., Dalpiaz, F., Brinkkemper, S.: Explicit alignment of requirements and architecture in agile development. In: REFSQ 2020. pp. 169-185. Lecture Notes in Computer Science, Springer International Publishing (2020)