UNIVERSITÄT
LEIPZIG

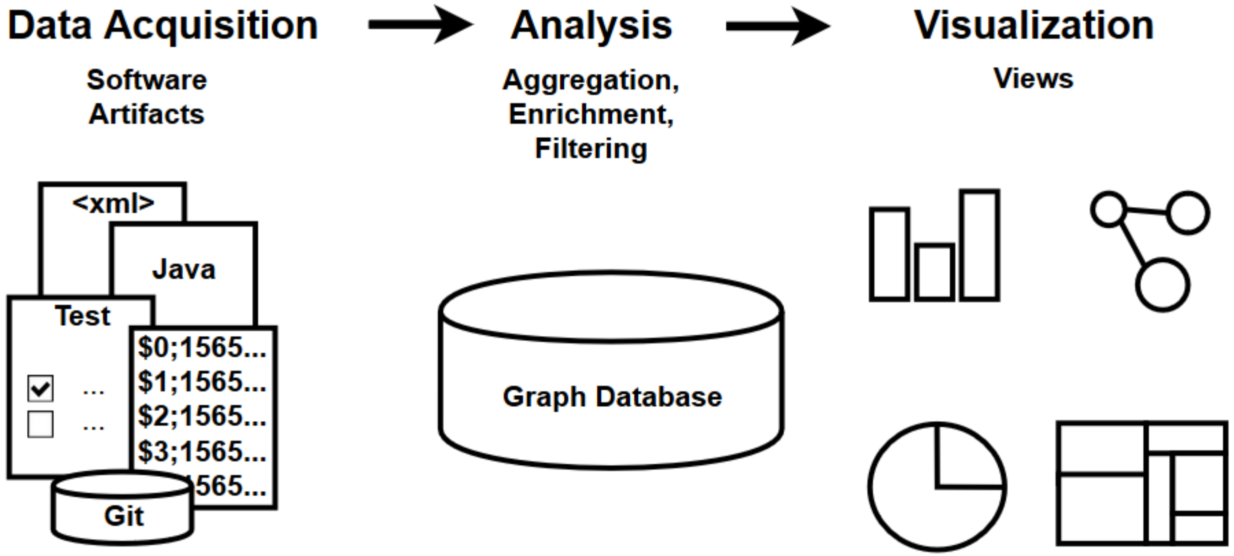
Symposium on Software Performance 2019

Graph-Based Analysis and Visualization of Software Traces

Würzburg, November 5, 2019

Richard Müller and Matteo Fischer

WHY GRAPHS?



– Software data naturally map to a multivariate, compound, attributed, and time-dependent graph

[Diehl and Telea 2014, Müller et al. 2018]

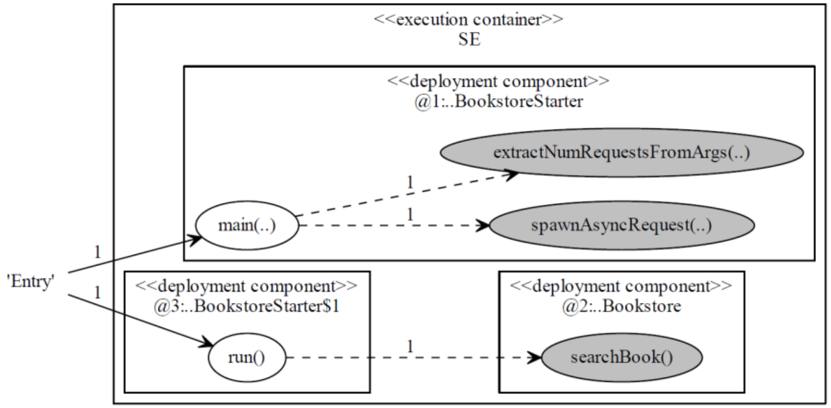
KIEKER

- Framework to monitor, analyze, and visualize software behavior
- Supports event-based and state-based monitoring
- Usable with Java, .NET, COBOL, and Visual Basic 6
- Provides tools
  - to inspect and analyze traces
  - to visualize them as UML sequence diagrams, markov chains, dependency graphs, and trace timing diagrams
- Output writers save traces to the file system or in a relational database

[van Hoorn, Waller, and Hasselbring 2012; Waller 2014; http://kieker-monitoring.net]

BUT…

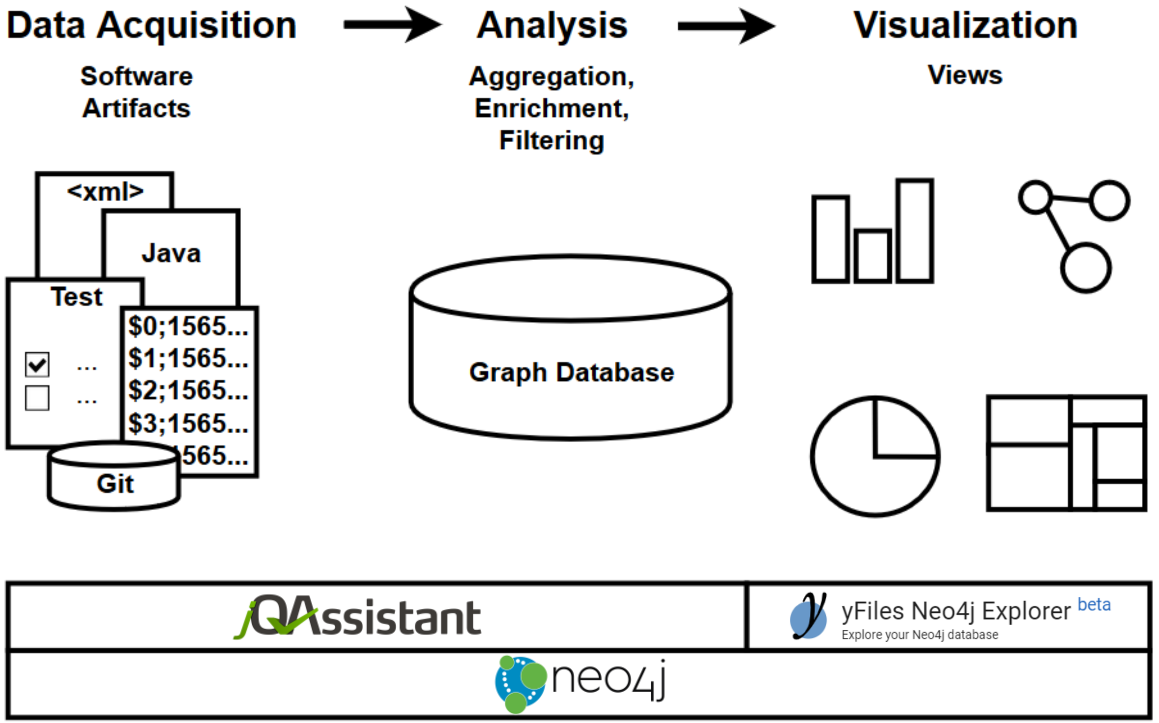
- There is no output writer for a graph database
- The visualizations produced by the Kieker tools are static images, for example,
  - Deployment operation dependency graph of Bookstore example



CONTRIBUTION

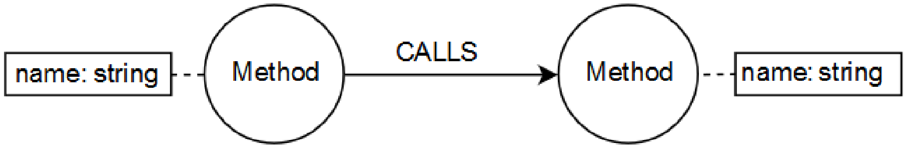
- jQAssistant plugin that scans event-based Kieker traces and stores them as a graph in a Neo4j database
- The plugin supports application performance monitoring and architecture discovery
- It complements existing Kieker tools
  - Analysis
    - Inspect and analyze traces with the graph query language Cypher
  - Visualization
    - Use interactive visualizations of call and dependency graphs

TECHNICAL BACKGROUND



NEO4J

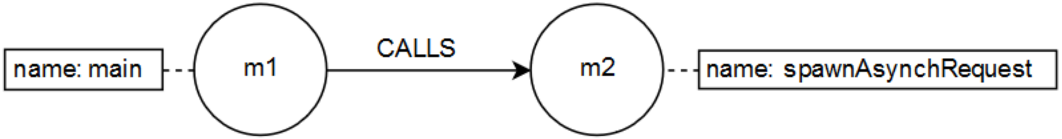
- Native graph database to store, manage, and query large amounts of connected data
- Models graph data with a labeled property graph
  - Labels are used to classify nodes
  - Relationships connect nodes, have a type, and can have a direction
  - Properties are attributes of nodes and relationships and stored as key-value pairs



[Needham and Hodler 2019; https://neo4j.com]

CYPHER

- Graph query language of Neo4j
- Matches given patterns in the graph using a visual, ASCII art-based syntax
  - `( )` node
  - `-[ ]->` directed relationship

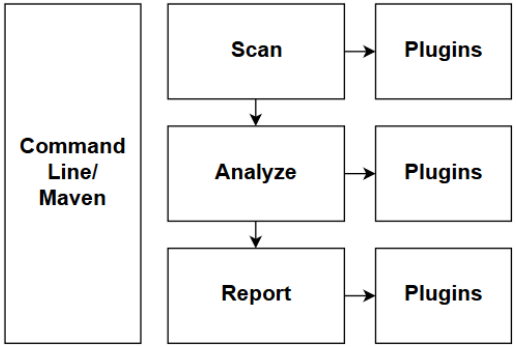


```
MATCH
     (m1:Method)-[CALLS]->(m2:Method)
RETURN
     m1.name, m2.name
```

[Francis et al. 2018; https://www.opencypher.org]

JQASSISTANT

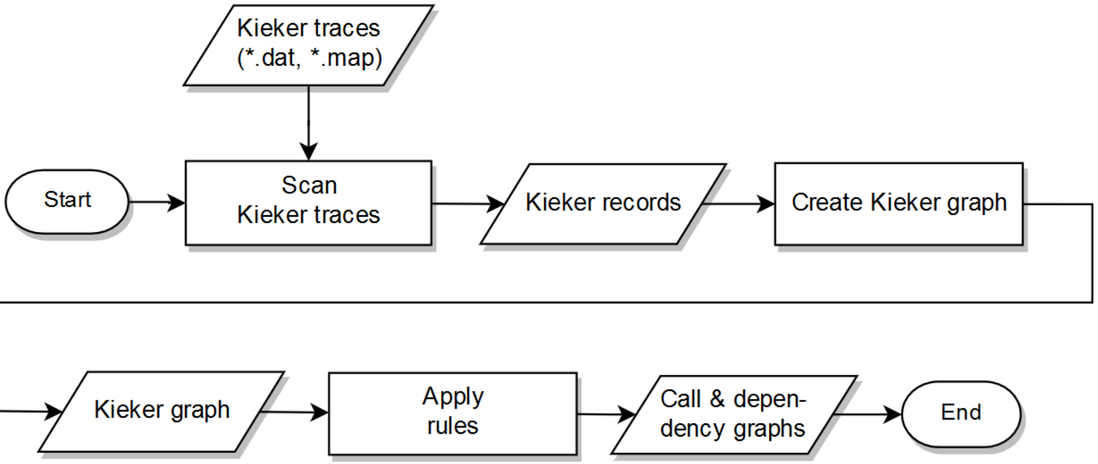
- Scans software artifacts and stores them in a Neo4j graph database
- Analyzes and modifies the graph data with rules
  - Constraints to identify violations
  - Concepts to aggregate, enrich, and filter
- Create reports
- Can be executed with Maven or from the command line
- Extendable through plugins, for example, Java, Jira, GitHub-Issues, JaCoCo scanner



[https://jqassistant.org; https://softvis-research.github.io/jqassistant-plugins]

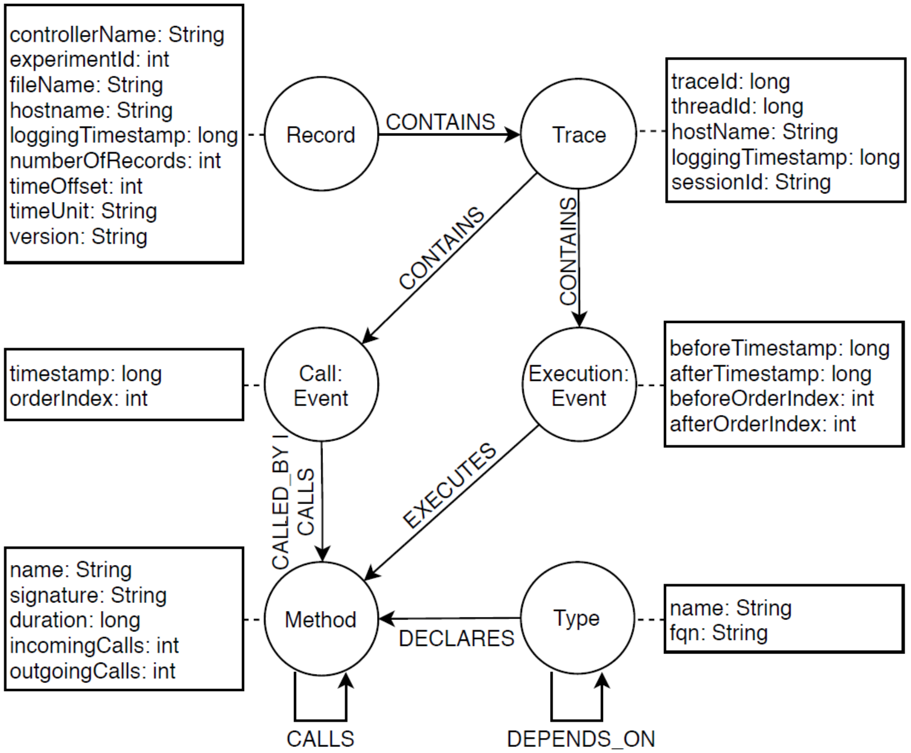
KIEKER PLUGIN

- Plugin for jQAssistant to scan and analyze event-based software traces
- Published on GitHub under GPL-3.0



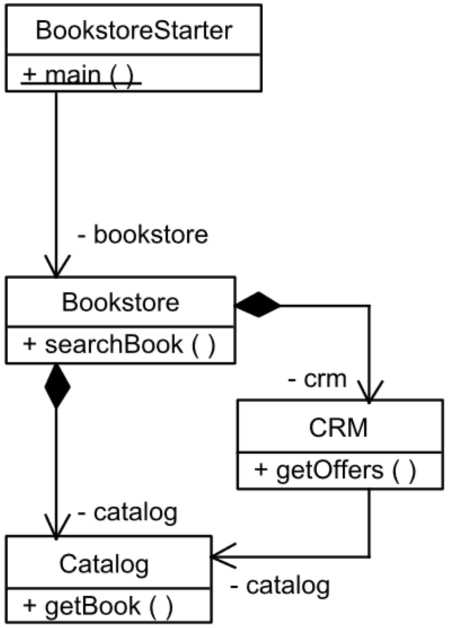
[https://github.com/softvis-research/jqa-kieker-plugin]

KIEKER GRAPH SCHEMA



APPLICATION EXAMPLE

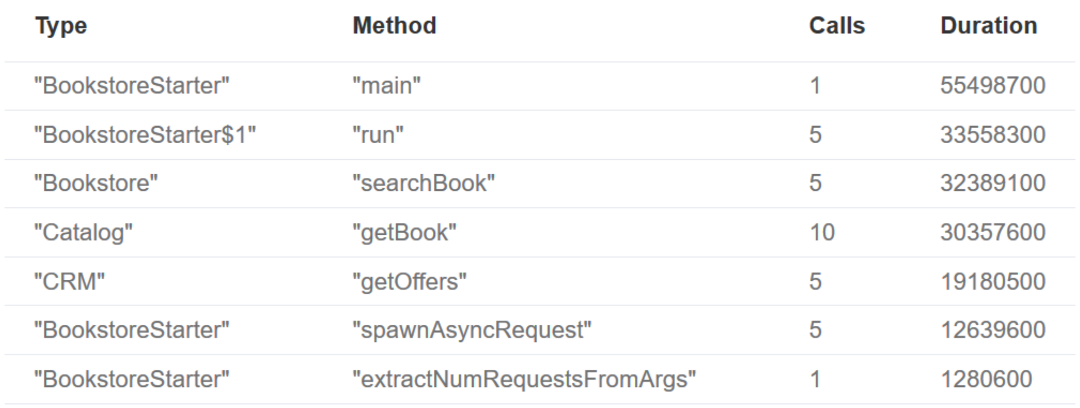
- Instrumented the Bookstore example from the Kieker user guide with AspectJ and activated aspects `OperationExecution` and `OperationCall`
- Scanned the monitored traces with the jQAssistant command line tool using the Kieker plugin



[http://kieker-monitoring.net/documentation]

ANALYSIS

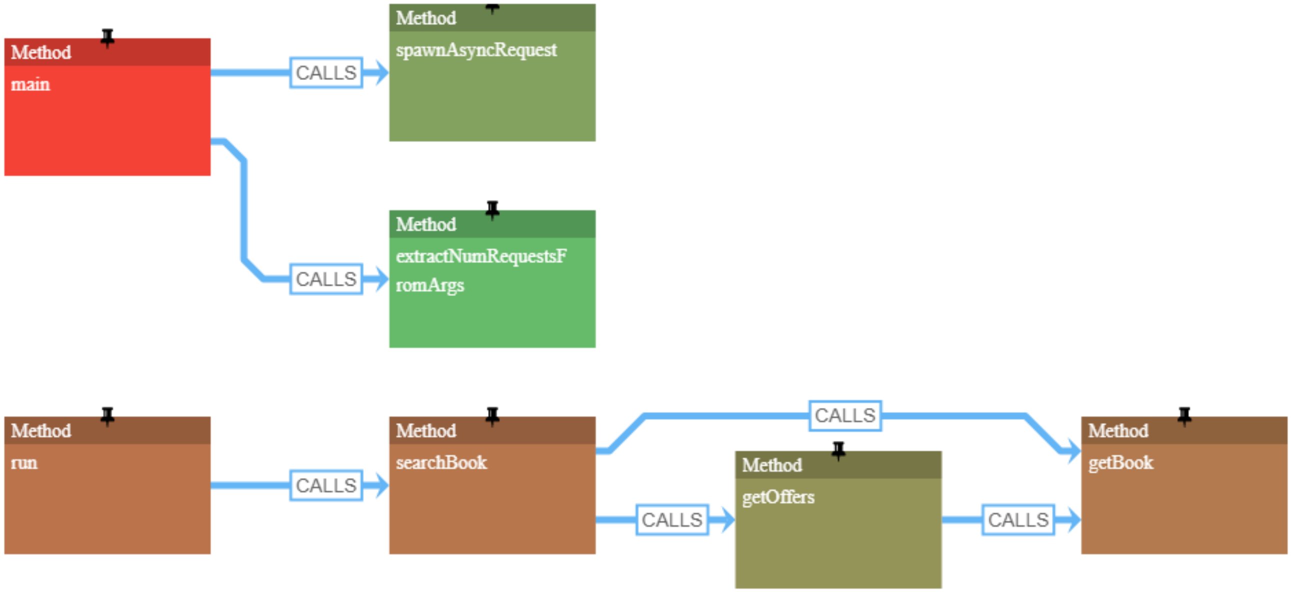
```
MATCH
     (t:Type)-[:DECLARES]->(m:Method)
WHERE
     t.fqn STARTS WITH "kieker"
RETURN
     t.name as Type, m.name AS Method, m.incomingCalls AS
     Calls, m.duration AS Duration ORDER BY Duration DESC
```



Type	Method	Calls	Duration
"BookstoreStarter"	"main"	1	55498700
"BookstoreStarter$1"	"run"	5	33558300
"Bookstore"	"searchBook"	5	32389100
"Catalog"	"getBook"	10	30357600
"CRM"	"getOffers"	5	19180500
"BookstoreStarter"	"spawnAsyncRequest"	5	12639600
"BookstoreStarter"	"extractNumRequestsFromArgs"	1	1280600

CALL GRAPH

yFiles Neo4j Explorer beta
Explore your Neo4j database

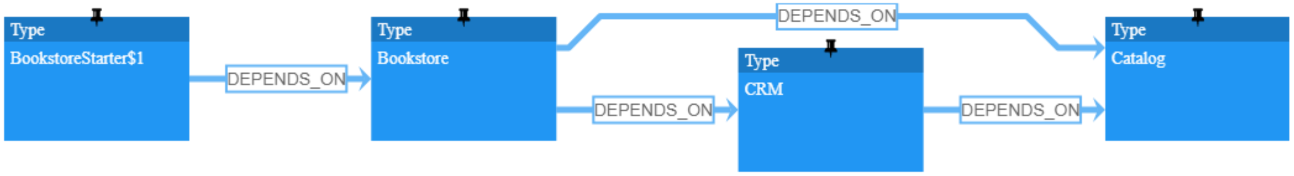


The property `duration` of each `Method` node is mapped to a color gradient from green (short) to red (long)

[https://www.yworks.com/neo4j-explorer]

DEPENDENCY GRAPH

yFiles Neo4j Explorer beta
Explore your Neo4j database



[https://www.yworks.com/neo4j-explorer]

CONCLUSION

- Presented a jQAssistant plugin that scans event-based software traces and stores them as a graph in a Neo4j database
- Illustrated feasibility and usefulness with the Bookstore example
  - Analysis with an example Cypher query for aggregated method calls
  - Visualization of the call and dependency graphs in the yFiles Neo4j explorer

FUTURE WORK

- Extend the plugin to scan further record types, for example, state-based records
- The plugin can be used as a blueprint to contribute a Kieker writer for graph databases

REFERENCES

- S. Diehl and A. C. Telea. “Multivariate Graphs in Software Engineering“. In: Multivar. Netw. Vis. Dagstuhl Semin. #13201 Dagstuhl Castle, Ger. May 12-17, 2013 Revis. Discuss. Ed. by A. Kerren, H. C. Purchase, and M. O. Ward. Vol. 8380. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014. Chap. 2, pp. 13-36.
- N. Francis et al. “Cypher: An Evolving Query Language for Property Graphs”. In: ACM SIG-MOD Int. Conf. Manag. Data. 2018, p. 13.
- A. van Hoorn, J. Waller, and W. Hasselbring. “Kieker: A framework for application performance monitoring and dynamic software analysis”. In: 3rd ACM/SPEC Int. Conf. Perform. Eng. (ICPE 2012). ACM, 2012, pp. 247-248.
- R. Müller et al. “Towards an Open Source Stack to Create a Unfied Data Source for Software Analysis and Visualization”. In: Proc. 6th IEEE Work. Conf. Softw. Vis. Madrid, Spain: IEEE, 2018.
- M. Needham and A. E. Hodler. Graph Algorithms -Practical Examples in Apache Spark & Neo4j. 1st ed. O'Reilly, 2019.
- J. Waller. Performance Benchmarking of Application Monitoring Frameworks. Kiel Computer Science Series 2014/5. Department of Computer Science, Kiel University, 2014.

UNIVERSITÄT LEIPZIG

THANK YOU.

Richard Müller and Matteo Fischer
Information Systems Institute, Software Engineering Department

rmueller@wifa.uni-leipzig.de
@rimllr
https://github.com/softvis-research
http://softvis.wifa.uni-leipzig.de